



102 PROGRAMMES

BASIC
GFA

ATARI ST

PSA

PAR
JACQUES
DECONCHAT

PSA

102 PROGRAMMES POUR
ATARI ST
EN BASIC GFA

Connaissez-vous la collection Programmes chez PSI ?

Amiga 500, 1000 et 2000

- 102 programmes Amiga - Jacques Deconchat
- Super Jeux Amiga - Jean-François Sehan

Amstrad CPC

- 102 programmes Amstrad - Jacques Deconchat
- Super Jeux Amstrad - Jean-François Sehan
- Amstrad en famille - Jean-François Sehan

Atari ST

- 102 programmes Atari , Basic GFA - Jacques Deconchat
- Super Jeux Atari , Basic GFA - Jean-François Sehan

PC et compatibles

- 102 programmes PC et compatibles - Jacques Deconchat
- Super Jeux PC et compatibles - Jean-François Sehan

Thomson

- 102 programmes MO6, TO8 et TO9+ - Jacques Deconchat
- Superjeux MO6, TO8 et TO9+ - Jean-François Sehan
- MO6, TO8 et TO9+ en famille - Jean-François Sehan

La loi du 11 mars 1957 n'autorisant, aux termes des alinéas 2 et 3 de l'article 41, d'une part, que les « copies ou reproductions strictement réservées à l'usage privé du copiste et non destinées à une utilisation collective », et, d'autre part, que les analyses et les courtes citations dans un but d'exemple et d'illustration, « toute représentation ou reproduction intégrale, ou partielle, faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause, est illicite » (alinéa 1^{er} de l'article 40).

Cette représentation ou reproduction, par quelque procédé que ce soit, constituerait donc une contrefaçon sanctionnée par les articles 425 et suivants du Code Pénal.

© *Éditions P.S.I. est une filiale de Nathan Éditeur*

GROUPE
DE
LA CITE

6-10, boulevard Jourdan
75014 PARIS
1988

ISBN : 2-86595-483-8

**JACQUES
DECONCHAT**

**102 PROGRAMMES POUR
ATARI ST
EN BASIC GFA**


**ÉDITIONS P.S.I.
1988**

Atari ST est une marque déposée d' Atari Corp
Basic GFA est une marque déposée par GFA Systemtechnik

SOMMAIRE

Avant-propos	9
Organisation de l'ouvrage	11

Niveau 1

Instructions élémentaires	14
Allumettes à plusieurs	22
Les allumettes	24
Le nombre mystérieux	26
Bataille navale	28
Roulette	30
Couloirs	32
Chaud, froid	34
Devine un chiffre	36
Le jeu de Wytthof	38
Golf	40
Pair-impair	42
Jackpot	44
Calendrier perpétuel	46
Le jeu du 21	48
Solfège	50
Le jeu de la fève	52
Rebonds	54
Jeu de l'oie aléatoire	56

Niveau 2

Dimensionnement d'un tableau – Recherche d'un élément dans un tableau et dans une chaîne – L'instruction DATA – Manipulation des chaînes	61
Le jeu de saute-mouton	64
La course au 20	66
Le jeu de Marienbad	68
Dé truqué	70
Espion	72
Jour de la semaine	74
421	76
Compose	78
Tour de cartes	80
Pendu inversé	82
Musique	84
Lettres	86
Machinamot	88
Tirage d'un dé truqué	90

Loto	92
Le jeu de la vie	94
Un taquin de calcul	97
Belote	100
Hanoï	102
Calculatrice	104

Niveau 3

Graphisme et jeux	108
Visions	110
Mathieu	112
Plus grand, plus petit	114
La bombe	116
Mots	118
Idem	120
Mastermind	122
Nicomaque	124
Le pendu	126
Accrochez les wagons	128
Le loup et les agneaux	130
Le lézard d'Alice	132
Reines sur l'échiquier	134
Un cube	136
Le jeu du missile	138
Chasse au sous-marin	140
Ordre	142
Bridge	144
A travers	146
Evasion	148
Permutations	150
Répétitions	152

Niveau 4

Techniques d'animation – Objets graphiques mobiles – Fenêtres et menus	157
Adresse	160
Zombies	162
Diago	164
Le singe	166
Affichage publicitaire	168
Chapeaux	170
Clavier	172
Dessin de dés	174
Smokey	176
Captures dans l'espace	178
Danger spatial	180

Course de voitures	182
Obstacles	184
Poker	186
Course de chevaux	188
Sous-marin	190
Horloge 1	192
Echiquier	194
L'électricien fou	196
L'indésirable	198

Niveau 5

Utilisation de la souris – Fonctions particulières	203
Horloge 2	204
Anagrammes	206
Cibles	208
Gouttes	210
Combat aérien	212
Anomalie	214
Le jeu du 15	216
Décomaque	218
Mur de briques	220
Plusieurs en un	222
Othello	224
Tic tac toe	226
Dames	228
Parcours du cavalier	230
Quatre à quatre	232
Dessin de cercles	234
Taquin de lettres	236
Œufs	238
Création de formes	240
Ardoise magique	242
Fais-moi un dessin	244
Echecs	246

Avant-propos

Vous trouverez dans ce livre plus de 100 programmes (essentiellement des jeux) rédigés en BASIC GFA (version 2.0) pour les ordinateurs Atari de la série ST (520 ST, 1040 ST et Méga St).

Pour utiliser ces programmes, il suffit de choisir un mode d'écran (320*200 en 16 couleurs, ou 640*200 en 4 couleurs, ou écran monochrome). Tous les programmes sont prévus pour fonctionner dans les trois modes. On lance le Basic GFA en cliquant deux fois sur l'icône correspondante, puis on tape le listing du programme.

On notera que chaque ligne de programme est numérotée, sur les listings de l'ouvrage. Le Basic GFA ne comporte pas de numéros de ligne. **Ces numéros ne doivent en aucun cas être tapés.** Ils ont été ajoutés, lors de l'impression, pour faciliter les commentaires sur les programmes : les ajouter au listing empêcherait les programmes de fonctionner correctement.

Il est prudent, après avoir introduit le texte d'un programme, de le sauver sur disquette avant son exécution : cette façon d'agir évitera d'avoir à retaper le programme en entier, en cas d'erreur.

En Basic GFA, les noms des variables et les mots réservés commencent automatiquement par une majuscule : on pourra donc frapper les programmes indifféremment en minuscules ou en majuscules, sans risque d'erreur.

Si, lors d'une tentative d'exécution, on rencontre une erreur de type "mémoire pleine", il sera prudent de redémarrer l'ordinateur, avant de relancer ce programme : il se peut, en effet, que la mémoire soit encombrée de résidus de variables inutiles, en provenance des programmes que l'on a essayés auparavant. Pour des raisons analogues, il sera préférable de faire un RESET (matériel ou logiciel, en tapant par exemple en mode direct VOID(FRE(O)) avant de lancer l'exécution d'un nouveau programme. Cette méthode est notamment conseillée si des problèmes se manifestent avec les affichages (caractères trop grands, remplissage de tout l'écran avec de la couleur, ...).

Les programmes proposés sont très courts, et l'ordre dans lequel ils sont présentés n'est bien évidemment pas arbitraire : il s'agit d'initier le lecteur, à travers la réalisation et l'utilisation de programmes de jeux de bonne qualité, à la connaissance et à la maîtrise progressive de cet outil extraordinaire qu'est un ordinateur individuel.

La brièveté des programmes proposés permettra de les taper facilement, sans risque d'erreur. Vous pourrez bien sûr les utiliser comme base pour la réalisation de jeux plus performants.

Les programmes sont abondamment commentés pour vous aider à comprendre leur fonctionnement, et les pages de présentation, au début de chaque chapitre, vous permettront de retrouver facilement les caractéristiques propres des instructions Basic qu'ils utilisent.

Vous trouverez, à la fin de chaque texte de présentation, quelques suggestions pour des améliorations possibles, ou des modifications, qui pourraient rendre le jeu proposé plus attrayant, plus performant ou plus complet.

Nota bene : les programmes ont été écrits et vérifiés pour la version 2 du Basic GFA. L'auteur ne saurait être tenu pour responsable dans le cas où une évolution future de ce Basic serait incompatible avec les textes proposés.

Rappelons que tous les programmes ont été prévus pour fonctionner dans les trois modes graphiques disponibles : moniteur noir et blanc, mode couleur moyenne résolution, mode couleur basse résolution.

Les numéros de ligne servent uniquement à commenter les listings, et ne doivent en aucun cas être écrits dans les programmes.

Organisation de l'ouvrage

Les jeux sont classés par ordre de difficulté croissante, suivant 5 niveaux :

- **Le niveau 1** permet l'étude des instructions élémentaires du langage Basic : **LET** (optionnel ici), **PRINT**, **INPUT**, **IF ... THEN**, **FOR ... NEXT**, **DO ... LOOP**, **REPEAT ... UNTIL**, **WHILE ... WEND**, **GOTO**, **COSUB**, **ON ... GOTO**, **ON ... GOSUB**.
- **Au niveau 2**, on utilisera les instructions de confection et de manipulation des éléments d'un tableau, et les instructions permettant le traitement des chaînes de caractères : **DIM**, **R(I)**, **DATA**, **MID\$**, **RIGHT\$**, **LEFT\$**.
- **Au niveau 3**, on s'intéresse plus particulièrement au graphisme : **PLOT**, **LINE**, **BOX**, **PBOX**, **RBOX**, **CIRCLE**, **PCIRCLE**, **TEXT**, etc., ainsi qu'aux procédures avec passage de paramètres.
- **Le niveau 4** est essentiellement consacré à une instruction permettant la réalisation de jeux interactifs : il s'agit de saisir au vol un caractère frappé au clavier, pour modifier le comportement du programme sans interrompre son déroulement. C'est l'instruction **INKEY\$**. On y découvrira également l'utilisation des **SPRITE**, la gestion des fenêtres d'alerte (**ALERT**) et des menus déroulants (**MENU**).
- Enfin, **le niveau 5** est entièrement dédié à l'utilisation de la souris (**MOUSE**), avec des programmes assez complexes pouvant réutiliser toutes les instructions découvertes dans les précédents chapitres.

Chaque jeu est présenté d'une façon très systématique, par une courte description suivie d'une étude détaillée de chacune des lignes importantes du programme (les lignes sont spécialement numérotées, lors de l'impression des programmes, pour faciliter les commentaires). Après quoi, les indications nécessaires au bon fonctionnement du jeu sont données, ainsi que quelques éléments pouvant conduire à des extensions ou à des améliorations du jeu proposé.

Les listings des programmes sont imprimés à partir des originaux de l'auteur, sur son imprimante personnelle.

Pour simplifier, la touche d'entrée, marquée par une flèche sur le clavier (↵), est appelée **RET** dans les commentaires.

Il est toujours possible de quitter un programme en cours d'utilisation avant sa fin normale, en tapant la combinaison **CSA** (elle est désignée ainsi dans les textes) : cela signifie qu'il faut taper simultanément les trois touches **Control Shift et Alternate**, pour sortir du programme. Ces trois touches sont situées en bas et à gauche, sur le clavier de l'Atari ST, et sont donc très facilement accessibles.

Niveau 1

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Allumettes à plusieurs	Jeu de réflexion et de stratégie, pour plusieurs joueurs.
Les allumettes	Jeu de réflexion, contre l'ordinateur.
Le nombre mystérieux	Jeu de déduction pour un joueur. Plusieurs niveaux.
Bataille navale	Un seul navire à trouver, mais quelques particularités.
Roulette	Une simulation élégante du jeu très connu.
Couloirs	Un jeu de logique, à découvrir.
Chaud, froid	Logique et hasard, pour un joueur. Plusieurs niveaux.
Devine un chiffre	Un jeu très facile, pour un ou plusieurs joueurs.
Le jeu de Wytthof	Un jeu de réflexion assez simple, pour deux joueurs.
Golf	Un parcours de 18 trous, pour un seul joueur.
Pair-impair	Un jeu de casino en trois temps, pour un ou plusieurs joueurs.
Jackpot	Jeu de hasard, pour un joueur.
Calendrier perpétuel	Confection d'un calendrier, mois par mois.
Le jeu du 21	Jeu de hasard. L'ordinateur tient la banque.
Solfège	Apprenez le solfège, à l'aide de votre ordinateur.
Le jeu de la fève	Un jeu de réflexion, pour deux joueurs.
Rebonds	Un jeu très visuel, pour un joueur.
Jeu de l'oie aléatoire	Un très vieux jeu, revu et corrigé par l'ordinateur, pour deux joueurs.

Instructions élémentaires

Nous ne retiendrons ici que les instructions fondamentales, et les instructions de structure ; beaucoup d'autres commandes, ou fonctions particulières, seront utilisées à l'occasion. Par exemple :

- *RND pour obtenir un nombre aléatoire entre 0 et 1.*
- *RANDOM (N) pour obtenir un entier entre 0 (compris) et N (non compris).*
- *INT pour prendre la partie entière d'un résultat.*
- *REM pour placer un commentaire.*
- *END pour terminer un programme.*
- *etc.*

Les instructions de programmation, utilisées dès le niveau 1, et présentes dans pratiquement tous les programmes sont :

LET (optionnelle) ; **PRINT** ; **PRINT AT** ; **INPUT** ; **FOR ... NEXT** ; **DO ... LOOP** ; **WHILE ... WEND** ; **REPEAT ... UNTIL** ; **IF ... THEN** ; **GOTO** ; **GOSUB** ; **ON ... GOTO** ; **ON ... GOSUB**.

L'INSTRUCTION LET OU INSTRUCTION D'AFECTATION

C'est l'instruction par excellence des langages de programmation évolués, puisque son rôle essentiel est de permettre, pour le programmeur, une gestion transparente de la mémoire. En pratique, cela signifie, que pour ranger une quantité en mémoire, par exemple 12, il n'est pas nécessaire de savoir dans quelle case de la mémoire on va la déposer : il suffit de lui affecter un nom autorisé, et c'est par ce nom que la quantité sera par la suite retrouvée et manipulée.

- **Si la quantité est numérique**, le nom sera une simple lettre de l'alphabet, ou un ensemble de plusieurs chiffres ou lettres, commençant obligatoirement par une lettre.
- **Si la quantité est alphanumérique**, la même règle s'applique pour le nom, qui devra cependant être suivi du symbole \$, et la variable devra être écrite entre guillemets.

Exemple : LET A1 = 12 ; la variable numérique A1 prend la valeur 12
LET NOM \$ = "RENOIR" ; la variable alphanumérique
NOM\$ prend la valeur RENOIR.

En GFA Basic, comme dans la plupart des Basic, le **LET** est devenu optionnel ; aucune ambiguïté n'étant possible, on écrira donc :

A1 = 12
NOM\$ = "RENOIR"
etc.

L'INSTRUCTION PRINT OU INSTRUCTION DE SORTIE (AFFICHAGE ECRAN)

C'est une instruction de dialogue, qui permet à la machine d'afficher sur l'écran un texte, le résultat d'un calcul, ou une variable. C'est le seul moyen que possède l'ordinateur pour faire connaître ses résultats. Ainsi, **PRINT A1** provoque l'affichage de 12 (si l'on a auparavant fait **A1 = 12**), **PRINT A\$** provoque l'affichage de BONJOUR (si l'on a fait **A\$ = "BONJOUR"**). Utilisé après un effacement d'écran, le **PRINT** provoque l'affichage sur la première ligne, en haut, à gauche.

Un certain nombre de paramètres peuvent être utilisés pour obtenir sur l'écran un affichage convenable :

- **PRINT TAB(N)** provoque sur la ligne d'impression courante un affichage commençant à la *N*ème colonne à partir de la gauche (de 1 à 40 colonnes, ou de 1 à 80 colonnes sont disponibles, selon le mode d'écran choisi).
- Un **point-virgule (;)** en fin de **PRINT** permet soit d'enchaîner une autre impression sans répéter l'instruction **PRINT**, soit d'indiquer que la prochaine impression sera, en l'absence d'indication contraire, effectuée exactement à la suite de l'impression précédente.
- La **virgule (,)** provoque une tabulation horizontale automatique prédéfinie.

Lors de la frappe, l'instruction **PRINT** pourra être abrégée en tapant ? (ou **P**). Il est à noter que le GFA Basic permet d'écrire en abrégé les instructions les plus courantes. On pourra consulter le manuel avec profit, à ce sujet.

L'instruction PRINT AT (X,Y) ;

Le GFA Basic dispose d'une instruction particulière, d'un emploi très souple, qui permet de positionner le curseur sur l'écran, en vue de la prochaine écriture, ou d'une saisie. Il s'agit de l'instruction **PRINT AT (X,Y) ;**

X correspond à une tabulation horizontale (numéro de colonne : de 1 à 40, et de 1 à 80, selon le mode d'écran), et Y à une tabulation verticale (numéro de ligne : de 1 à 21, variable selon la taille des caractères).

Ainsi, pour écrire le mot BONJOUR à la 5^e colonne de la 8^e ligne, on tapera :

PRINT AT(5,8) ; "BONJOUR"

L'INSTRUCTION INPUT OU INSTRUCTION D'ENTREE (CLAVIER)

Cette instruction, qui provoque un arrêt et une attente de l'appareil, permettra d'entrer un nombre, ou un texte, qui sera affecté à la variable suivant l'ordre **INPUT**. Ceci est fait en mode conversationnel, c'est-à-dire au fur et à mesure des besoins ; en effet, lorsque le GFA Basic rencontre une instruction **INPUT**, il s'arrête et vous rend la main. On peut alors lui entrer une (ou plusieurs) information de nature convenable (numérique ou alphanumérique) et lui signaler que cette entrée est terminée à l'aide de la touche **RETURN**, ce qui lui rendra la main (le programme continue alors à s'exécuter).

Si l'information entrée n'est pas de nature convenable (vous tapez, par exemple, un texte sur un **INPUT X**, qui attend un nombre), l'ordinateur émet un léger BIP caractéristique, et se remet en attente.

Il est possible d'afficher des messages avant l'entrée de la valeur de la variable, et de placer le curseur à un endroit précis de l'écran en utilisant **PRINT AT(X,Y) ;**

On pourra, par exemple, écrire :

PRINT AT(5,8) ;
INPUT "VOTRE NOM, S'IL VOUS PLAÎT" ; NOM\$

On pourra aussi, dans de nombreux cas, utiliser une instruction **INPUT\$(N)**, qui permet d'attendre une entrée jusqu'à ce que le nombre N de caractères demandés ait été frappé. Ainsi **R\$=INPUT\$(1)** permet l'attente d'un seul caractère saisi dans R\$. Il n'est pas nécessaire dans ce cas de signaler la fin de l'entrée en tapant **RETURN**.

LES INSTRUCTIONS DE STRUCTURATION DU BASIC GFA

L'instruction FOR ... NEXT, ou boucle avec compteur incorporé

On est très souvent conduit, en programmation, à vouloir exécuter un nombre entier de fois une certaine série d'instructions (par exemple pour faire avancer un mobile de la colonne 6 à la colonne 20, etc.). Tous les Basic disposent pour faire cela d'une instruction dite de boucle, la variable de boucle prenant ses valeurs dans les entiers ou les décimaux, et le pas de

cette variable (c'est-à-dire la quantité dont elle augmente ou diminue à chaque passage) pouvant prendre ses valeurs dans les entiers ou les décimaux.

Ainsi, si l'on écrit :

FOR I = 0 TO 20 STEP 2

...

... **instructions**

...

NEXT I

Les instructions contenues dans la boucle seront exécutées pour $I = 0$, puis pour $I = 2$, etc. et ceci jusqu'à $I = 20$, soit 11 fois. La variable I peut être utilisée pour faire un calcul, mais ne devra pas être modifiée dans la boucle. Il est prévu une facilité permettant la sortie avant la fin de la boucle, sous la forme d'une instruction **EXIT IF** (test de sortie).

Si le pas de la boucle (**STEP**) n'est pas précisé, il sera automatiquement pris égal à 1.

Il est possible d'emboîter plusieurs boucles dont les variables portent des noms différents.

Une boucle inconditionnelle : DO ... LOOP

L'instruction **DO ... LOOP** permet de résoudre un problème particulier, en évitant de recourir à la tristement célèbre instruction **GOTO** des anciens Basic (qui reste cependant disponible ici) : il s'agit de réaliser des boucles sans fin, avec la possibilité éventuelle de s'échapper par un test placé n'importe où dans la boucle. Ce type de boucle pourra être d'un emploi très commode lorsqu'il s'agit de répéter indéfiniment une action dont il est difficile de prévoir si elle doit ou non s'arrêter à un moment donné.

Il est possible de sortir à tout moment avec un test **EXIT IF...**

Exemple

N = 0

DO

N=N+RANDOM(5)

EXIT IF N>45

N=N-RANDOM(3)

PRINT N

LOOP

Les boucles avec test en début de boucle : WHILE ... WEND

La boucle **FOR ... NEXT** souffre d'un excès de rigidité, puisque le compteur de boucle (en général une variable entière) ne peut progresser que d'une quantité constante, définie au début (de 1 en 1, ou de 2 en 2, etc.).

La structure **WHILE ... WEND** est en général plus souple : le test permettant de sortir de la boucle est placé en début de boucle (par exemple **WHILE a<>0**) et il est possible de tester plusieurs variables (**WHILE a<>0 OR b=1**). De plus la boucle ne sera pas parcourue si le test se révèle négatif au départ.

On peut également sortir dans le corps de la boucle, avec **EXIT IF...**

Les boucles avec test en fin de boucle : REPEAT ... UNTIL

La structure **REPEAT ... UNTIL** permet la même souplesse que la structure **WHILE ... WEND**, mais dans ce cas, le test permettant de quitter la boucle est effectué en fin de boucle. Il est possible de sortir à n'importe quel endroit avec **EXIT IF ...**

LES STRUCTURES DE TEST EVOLUEES : LE BLOC IF ... THEN ... ELSE ... ENDIF

La plupart des Basic disposent d'une structure de test sur une seule ligne, sous la forme : **IF condition THEN instruction 1 ELSE instruction 2.**

Lorsque l'ensemble des instructions à exécuter est complexe, et ne peut s'écrire sur une seule ligne, le programmeur est alors obligé d'utiliser des renvois, sous la forme **GOTO** ou **GOSUB**, qui alourdissent le programme et rendent sa lecture très difficile. Le Basic GFA résoud ce problème d'une façon très élégante, en proposant d'emblée une structure de test très évoluée : le bloc **IF ... ENDIF**.

Le format d'écriture de ce type de test se présente ainsi :

IF condition THEN

...

... **instructions**

...

ELSE

...

... **instructions**

...

ENDIF

L'utilisation du **ELSE** est évidemment facultative, et il est possible d'emboîter plusieurs structures de ce type, ce qui donne une très grande souplesse de programmation.

LE RENVOI INCONDITIONNEL : GOTO

Le Basic GFA n'utilise pas de numéros de ligne. Ces lignes sont donc exécutées les unes après les autres, dans l'ordre où elles sont présentes dans le texte du programme. Il peut être nécessaire, au cours d'un programme, de ne pas suivre cet ordre. On peut alors utiliser une instruction **GOTO**, qui détourne le fonctionnement du programme de son cours normal, en enchaînant l'exécution sur les lignes commençant à l'étiquette désignée après l'ordre **GOTO**.

L'étiquette est un nom quelconque, choisi par le programmeur de la manière la plus explicite possible, et se termine impérativement par deux points.

Exemple

...

GOTO essai le programme se branche sur la ligne commençant par
 essai

...

...

...

essai :

...

L'instruction **GOTO** pourra être également utilisée sous une forme plus élaborée, permettant de renvoyer à une ligne dûment étiquetée en tenant compte de la valeur d'une variable numérique : c'est l'instruction **ON ... GOTO ...**

On pourra par exemple écrire :

ON X GOTO début, milieu, fin.

Le renvoi sera fait à l'étiquette début si X vaut 1, à l'étiquette milieu si X vaut 2, et à l'étiquette fin si X vaut 3.

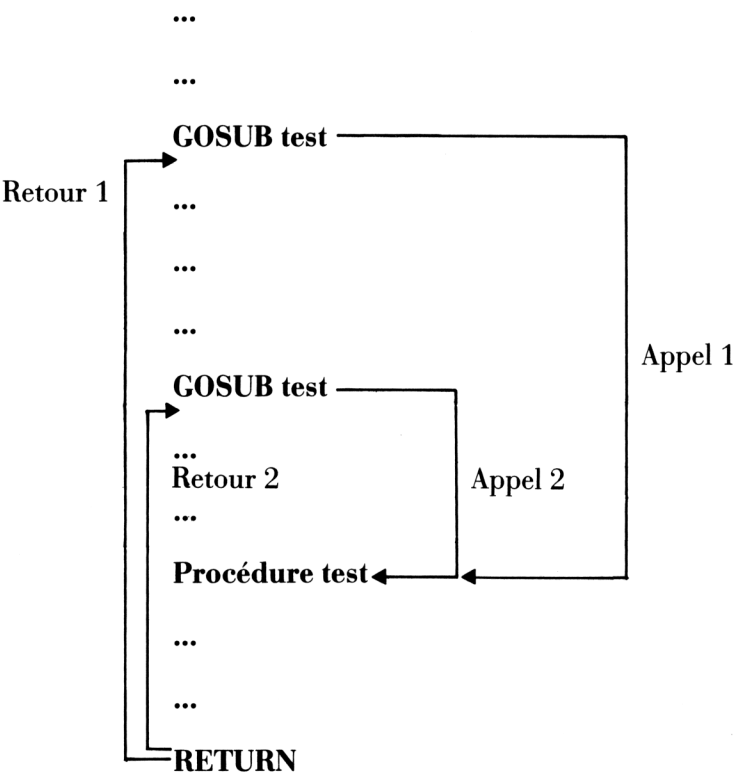
L'INSTRUCTION GOSUB OU INSTRUCTION D'APPEL DE SOUS-PROGRAMME

Lorsqu'une même suite d'instructions doit être exécutée plusieurs fois, et appelée à partir de plusieurs endroits du programme, on évitera de la réécrire plusieurs fois. On la rédige une fois pour toutes, et on la place en dehors du programme proprement dit (par exemple en fin de programme). L'appel de ce sous-programme se fait par une instruction **GOSUB étiquette**, mais ici, l'étiquette n'est pas écrite, comme pour **GOTO**, directement dans le programme. Il s'agit en fait d'une procédure, c'est-à-dire d'un programme autonome, ayant ses propres variables, et dans lequel le programme appelant ne peut rentrer que s'il y a appel.

La procédure doit impérativement se terminer par une instruction **RETURN**, qui provoque le retour dans le programme principal, à la ligne suivant celle qui avait provoqué le détournement.

Voici le schéma théorique de fonctionnement :

Programme principal



Un sous-programme peut lui-même appeler un autre sous-programme, et ainsi de suite.

On peut également appeler un sous-programme en tenant compte d'une condition préalable, avec la forme **ON ... GOSUB...**

Par exemple :

ON X GOSUB premier, deuxième, troisième

On part exécuter la procédure premier si X vaut 1, la procédure deuxième si X vaut 2, et la procédure troisième si X vaut 3.

Allumettes à plusieurs

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : logique (plusieurs joueurs).

Difficulté du jeu : assez difficile.

LE JEU :

Dans ce jeu pour plusieurs joueurs, le meneur de jeu devra, en début de partie, choisir un nombre d'allumettes, et indiquer au ST le nombre de joueurs.

A chaque coup, le joueur qui a le trait est autorisé à retirer du tas d'allumettes restantes jusqu'au double de ce que vient de retirer le précédent joueur ; celui qui retire la dernière allumette a gagné (ou perdu, si l'on modifie légèrement les règles).

LE PROGRAMME :

Les lignes 4 et 5 permettent les différentes entrées.

Les lignes 12 à 16 réalisent les impressions.

La ligne 19 contrôle le coup de chaque joueur (par rapport au coup précédent).

La ligne 10 gère les tours (remise à 1 du numéro du joueur).

La ligne 9 réalise le test de fin, et la ligne 24 affiche le numéro du joueur gagnant (avec la règle prévue initialement).

POUR JOUER :

Après avoir lancé le programme par RUN, on entre le nombre initial d'allumettes, suivi de RET, puis le nombre de joueurs (et RET).

Le premier joueur ne pourra enlever qu'une allumette, le deuxième pourra en enlever 1 ou 2 ; le suivant 1, 2 ou 1, 2, 3 ou 4 selon le coup précédent, et ainsi de suite ; le retrait maximal autorisé est à chaque fois affiché par le ST.

Lorsque la dernière allumette a été prise, on obtient l'affichage LE JOUEUR... A GAGNE, et la partie est terminée.

On arrête le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut convenir que celui qui prend la dernière allumette est éliminé du jeu, et recommencer la partie entre les joueurs restants qui seront ainsi éliminés les uns après les autres.

On peut aussi imaginer une représentation graphique des tas d'allumettes, tirer au hasard le nombre à enlever en début de partie, etc.

```

1  Rem ALLUMETTES A PLUSIEURS
2  Fullw 2
3  Clearw 2
4  Input "NOMBRE TOTAL D'ALLUMETTES :";N
5  Input "NOMBRE DE JOUEURS      :";J
6  F=1
7  K=0
8  Clearw 2
9  While N<>0
10     K=K+1+K*(K>=J)
11     Print
12     Print "TOUR DU JOUEUR ";K
13     Sound 1,8,4,K,4
14     Sound 1,0,0,0
15     Print "TOTAL RESTANT :";N
16     Print "MAXIMUM AUTORISE :";F
17     Repeat
18         Input "VOTRE COUP :";C
19     Until C>0 And C=Int(C) And C<=F And C<=N
20     N=N-C
21     F=2*K
22 Wend
23 Print
24 Print "LE JOUEUR ";K;" GAGNE"
25 R$=Input$(1)

```

Les allumettes

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : contre le ST.

Difficulté du jeu : moyenne.

LE JEU :

Le joueur devra en début de partie choisir une certaine quantité d'allumettes, ainsi que la prise maximale autorisée. Il décide ensuite s'il commence, ou s'il laisse commencer le ST. Chacun prend à son tour autant d'allumettes qu'il le désire, dans la limite du nombre maximal autorisé. Celui qui prend la dernière allumette est considéré comme perdant.

LE PROGRAMME :

Le jeu du joueur commence en ligne 26.

Celui du ST commence en ligne 15.

La ligne 28 contrôle la légalité du coup du joueur.

Les lignes 23 et 29 gèrent le changement de joueur.

Le coup du ST est calculé dans une boucle, de la ligne 16 à la ligne 20.

Le résultat est affiché par la ligne 24.

Si la partie est terminée, la ligne 7 le détecte et on continue en 34 pour le message de fin.

On arrête le programme en tapant la combinaison CSA.

POUR JOUER :

On commence l'exécution par RUN, et, à l'affichage du message NOMBRE TOTAL, on entre le nombre d'allumettes, suivi de RET. Puis on entre la prise maximale autorisée, suivie de RET. Si le joueur veut commencer, il frappe ensuite 0 ; sinon il frappe 1, puis RET.

Après chaque coup, le nombre d'allumettes restantes est affiché, et le ST affiche VOUS RETIREZ lorsque c'est au joueur de prendre.

Celui qui prend la dernière allumette a perdu.

EXTENSIONS POSSIBLES :

On pourra bien sûr envisager un affichage graphique couleur plus spectaculaire, imaginer laisser le hasard choisir le nombre d'allumettes et la prise maximale, etc.

```

1  Rem LES ALLUMETTES
2  Fullw 2
3  Clearw 2
4  Input "NOMBRE TOTAL D'ALLUMETTES :";N
5  Input "PRISE MAXIMUM PAR COUP :";I
6  Input "QUI COMMENCE (VOUS:0 - MOI:1) :";D
7  While N<>0
8      For J=1 To N
9          Print "!";
10         Sound 1,8,4,4,1
11     Next J
12     Sound 1,0,0,0,0
13     Print
14     If D=1 Then
15         S=I+D
16         Do
17             T=(N-S)/(I+D)
18             Exit If T=Int(T)
19             S=S-D
20         Loop
21         P=Abs(S-D)
22         P=P-(P=0)
23         D=0
24         Print "J'EN PRENDS ";P
25     Else
26         Repeat
27             Input "VOUS ENLEVEZ :";P
28             Until P=Int(P) And P>=1 And P<=I And P<=N
29             D=1
30         Endif
31         N=N-P
32         Print "IL RESTE ";N;" ALLUMETTES"
33     Wend
34     Print
35     If D=1 Then
36         Print "J'AI GAGNE..."
37     Else
38         Print "BRAVO"
39     Endif

```

Le nombre mystérieux

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de déduction pour un joueur.

Difficulté du jeu : facile.

LE JEU :

On demande à l'ordinateur de fabriquer un nombre de n chiffres (avec n compris entre 1 et 7).

On devra ensuite essayer de le découvrir, avec le moins d'essais possible, en recevant les seuls renseignements TROP GRAND ou TROP PETIT.

LE PROGRAMME :

La ligne 7 permet de fabriquer le nombre à deviner.

Les entrées sont faites par les lignes 12 et 13.

La ligne 14 teste si le résultat est exact.

La ligne 17 signale un essai trop petit, et la ligne 21 un essai trop grand.

Le compteur d'essais est en ligne 24.

La ligne 28 s'occupe de l'affichage du résultat.

POUR JOUER :

On lancera le programme par RUN.

On obtient un premier affichage NOMBRE DE CHIFFRES auquel on répond par un entier entre 1 et 7, suivi de RET.

On obtient alors l'affichage ESSAI N° 1, suivi de la question VOTRE NOMBRE.

On entre un nombre, et l'on presse RET.

On obtient alors une réponse TROP PETIT, ou TROP GRAND, ou encore, si l'on a trouvé juste, un affichage de VOUS AVEZ TROUVE... EN... ESSAIS. Si le résultat n'est pas exact, on revient à ESSAI N° 2, et VOTRE NOMBRE.

On peut arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra imaginer de faire deviner au joueur autre chose qu'un chiffre (par exemple, un mot, en utilisant l'ordre alphabétique, ou encore autre chose...).

```

1  Rem NOMBRE MYSTERIEUX
2  Fullw 2
3  Clearw 2
4  Repeat
5      Input "NOMBRE DE CHIFFRES :";N
6  Until N<=7 And N>0 And N=Int(N)
7  X=Random(10^N)+1
8  J=1
9  Clearw 2
10 Do
11     Deftext 1
12     Print "ESSAI NUMERO ";J
13     Input "VOTRE NOMBRE :";M
14     Exit If M=X
15     If M<X Then
16         Deftext 3
17         Print "TROP PETIT"
18         Sound 1,8,5,2,40
19     Else
20         Deftext 2
21         Print "TROP GRAND"
22         Sound 1,8,8,4,40
23     Endif
24     J=J+1
25     Sound 1,0
26 Loop
27 Print
28 Print "VOUS AVEZ TROUVE ";X;" EN ";J;" ESSAIS"
29 Sound 1,8,10,4,25
30 Sound 1,0

```


Bataille navale

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : réflexion.

Difficulté du jeu : facile.

LE JEU :

Une bataille navale a priori assez simple, avec un seul navire caché au hasard dans une grille de 10 cases sur 10 cases.

A chaque coup, on devra tirer en donnant les coordonnées H (horizontale) et V (verticale) de la case visée, et le ST répond en affichant la distance séparant le navire invisible de la case visée. Mais, si l'on tire trop près, le navire risque de se recacher au hasard.

LE PROGRAMME :

Les boucles emboîtées I (8-14) et J (11-13) dessinent le terrain de jeu.

Les lignes 16 et 17 définissent l'emplacement du navire.

Les coordonnées du tir sont entrées aux lignes 22 et 24, et la ligne 26 affiche le coup sur l'écran. La distance est calculée en ligne 27, et la ligne 29 regarde si l'on a ou non fait mouche.

La ligne 35 intervient pour déplacer le navire si les coups sont rapprochés.

La ligne 36 permet l'affichage de la distance.

POUR JOUER :

Le terrain de jeu est affiché après avoir frappé RUN. A l'affichage de LIGNE, on répond en donnant le numéro de la ligne horizontale sur laquelle on désire tirer. On tape ensuite RET et, à la question COLONNE, on répond en donnant le numéro de la colonne verticale suivi de RET.

Si on a tiré trop près, on obtient l'affichage PFFFF... qui signifie que le navire ennemi a changé d'emplacement.

Si on a tiré sur la bonne case, on obtient l'affichage du nombre total de tirs effectués pour gagner.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Elles sont innombrables, puisqu'aussi bien la bataille navale est l'un des jeux les plus répandus qui soit. On peut introduire deux joueurs, jouer contre le ST, mettre plusieurs navires, des sous-marins, etc.

```

1  Rem BATAILLE NAVALE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";K
5  R1=1-(K>1)
6  Clearw 2
7  S=1
8  For I=0 To 9
9      Print At(3*I*R1+5,1);I
10     Print At(1,I+2);I;
11     For J=0 To 9
12         Print At(3*J*R1+5,I+2);". ";
13     Next J
14 Next I
15 Do
16     X=Random(10)
17     Y=Random(10)
18     Do
19         Deftext 1
20         Print At(1,14);Spc(40*R1)
21         Print At(1,14);"LIGNE :";
22         Input H
23         Print At(21,14);"COLONNE :";
24         Input V
25         Deftext 2
26         Print At(3*V*R1+5,H+2);S
27         D=Int(Sqr((X-H)*(X-H)+(Y-V)*(Y-V)))
28         Deftext 3
29         If D=0 Then
30             Print
31             Print "TROUVE EN ";S;" COUPS"
32         End
33     Endif
34     S=S+1
35     Exit If D=1
36     Print At(1,16);"DISTANCE  :";D;"          "
37 Loop
38 Print At(1,16);"PFFFF....          "
39 Loop

```

Roulette

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : hasard (plusieurs joueurs).

Difficulté du jeu : facile.

LE JEU :

Il est surtout spectaculaire par son aspect graphique : une boule, représentée par une étoile, se déplace sur un terrain de jeu comportant des cases numérotées de 1 à 36, ainsi que trois cases supplémentaires, le 00, une case P (comme pair) et une case I (comme impair).

La boule parcourt au moins une fois chaque case, puis s'arrête au deuxième tour ; on peut ensuite la relancer pour une autre partie.

LE PROGRAMME :

Les boucles emboîtées K (8-13) et I (9-12) dessinent le terrain de jeu, auquel la ligne 14 rajoute les trois symboles P, 00 et I.

Les boucles emboîtées K (17-45) et I (18-44) réalisent le déplacement de la boule, l'arrêt étant contrôlé par la ligne 32.

La boucle J (30-31) réalise une légère temporisation.

La ligne 46 tire la position d'arrêt.

Les lignes 35 à 41 gèrent les essais suivants.

POUR JOUER :

Il suffit de faire RUN, et de regarder.

Une fois la boule arrêtée, il suffira de frapper la touche 0 pour la relancer.

Taper une touche quelconque pour sortir du programme.

EXTENSIONS POSSIBLES :

On pourra essayer de se rapprocher davantage d'une véritable roulette, de différentes manières : mieux dessiner le terrain de jeu, introduire des couleurs, simuler le ralentissement de la boule, introduire différentes possibilités de pari, et assurer la gestion des gains correspondants, tenir les comptes de plusieurs joueurs, et annoncer les résultats sous une forme plus usuelle.

```

1  Rem ROULETTE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  Clearw 2
6  H=4+(R=1)
7  Deftext 1-2*(R<3)
8  For K=1 To 4
9      For I=1 To 9
10         S=2*I
11         Print At(H*I,4*K-2);(S And K=1)+(S+18 And K=2)
12         +(S-1 And K=3)+(S+17 And K=4)
13     Next I
14 Next K
15 Print At(1,4);"P";At(1,8);"00";At(1,12);"I"
16 Deftext 1-(R=2)-3*(R=1)
17 Do
18     For K=1 To 4
19         For I=1 To 9-(K<4)
20             If I=10 And K<4 Then
21                 F=4*K+1
22                 G=1
23             Else
24                 F=4*K-1
25                 G=H*I
26             Endif
27             X=X-1
28             Print At(G,F);"*";
29             Sound 1,12,5,7,5
30             Sound 1,0,0,0,0
31             For J=1 To 150
32                 Next J
33                 If X<>1 Then
34                     Print At(G,F);" "
35                 Else
36                     Deftext 1
37                     Print At(1,17);"UN AUTRE ESSAI (O/N) :";
38                     R%=Upper$(Input$(1))
39                     If R$="O" Then
40                         Run
41                     Else
42                         End
43                     Endif
44                 Endif
45             Next I
46         Next K
47         X=Random(38)
48     Loop

```

Couloirs

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion.

Difficulté du jeu : difficile.

LE JEU :

Il fait appel à des notions de logique élémentaire. Le joueur peut se présenter à l'entrée, soit du couloir numéro 1, soit du couloir numéro 2. Dans ces deux entrées sont disposés des tourniquets qui basculent à votre passage et vous laissent passer, une fois vers la droite, et la fois suivante vers la gauche. De plus, un tourniquet analogue a été disposé dans le couloir central. Pourrez-vous prévoir où (sortie A ou B) va s'effectuer votre prochaine sortie ?

LE PROGRAMME :

Les lignes 5 à 13 dessinent les couloirs (et les trois tourniquets).

La ligne 24 attend votre choix.

Le décodage du chemin est fait par le ST en utilisant trois variables logiques I, J et K, représentant les états des trois tourniquets (lignes 28 à 55).

POUR JOUER :

Après la frappe de RUN, le couloir est dessiné sur l'écran, les tourniquets étant visibles, et l'on devra taper 1 ou 2 en réponse à la question du ST.

Dans la version retenue ici, la réponse est alors immédiatement affichée, et le chemin suivi depuis le début du jeu, ainsi que les sorties correspondantes, sont affichés sur l'écran. Cela présente l'avantage de permettre à plusieurs joueurs d'exercer leur sagacité en essayant de prévoir, pour une entrée donnée, ce que va répondre le ST.

On arrête le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Elles seront très nombreuses ; il est très facile d'imaginer un couloir plus complexe avec d'autres entrées et d'autres sorties, et d'autres tourniquets. On peut imaginer un affichage graphique des résultats en dessinant, un bref instant, l'état des tourniquets après la réponse du joueur. On peut aussi ajouter un test des réponses (jeu à plusieurs), décompter les points de chaque joueur, etc.

```

1  Rem COULOIRS
2  Fullw 2
3  Clearw 2
4  Deftext 2
5  Print "=====
6  Print
7  Print "1  <  ===== A"
8  Print "      >"
9  Print "===== < =====
10 Print "      >"
11 Print "2  <  ===== B"
12 Print "      >"
13 Print "=====
14 I=0
15 J=0
16 K=0
17 R$=""
18 T$=""
19 Deftext 1
20 Do
21   Print At(1,11);"ENTREE 1 OU 2 :      "
22   Repeat
23     Print At(18,11);
24     Input C$
25     Until C$<>" " And C$<"3" And Len(C$)=1
26     R$=R$+C$
27     Print At(1,13);R$
28     If C$="1" Then
29       I=Not (I)
30       If I<>0 Then
31         Goto Changk
32       Else
33         S$="A"
34         Goto Fin
35       Endif
36     Endif
37     J=Not (J)
38     If J<>0 Then
39       Goto Changk
40     Endif
41     S$="B"
42   Fin:
43   T$=T$+S$
44   Print At(1,15);"SORTIE EN ";S$
45   Print
46   Print T$
47 Loop
48 Changk:
49 K=Not (K)
50 If K<>0 Then
51   S$="B"
52 Else
53   S$="A"
54 Endif
55 Goto Fin

```

Chaud, froid

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : solitaire.

Difficulté du jeu : moyenne.

LE JEU :

Un des plus vieux jeux du monde, toujours aussi pratiqué, mais cette fois, c'est l'ordinateur qui vous sert de partenaire : il va en effet cacher un objet à un certain emplacement d'une grille dont vous aurez vous-même choisi les dimensions. Pour le trouver, il ne vous donnera que quelques indications sous la forme TU BRULES, ou C'EST FROID, etc.

Vous devez réussir à découvrir l'objet caché, avec le moins d'essais possible (votre score sera affiché en fin de partie).

LE PROGRAMME :

Les lignes 5 et 6 définissent la position de l'objet.

La ligne 15 calcule la distance par rapport aux deux coordonnées que vous avez entrées aux lignes 12 et 13.

La ligne 16 regarde si vous avez gagné.

Les lignes 18 et 19 normalisent la distance (qui doit avoir un sens, quelles que soient les dimensions de la grille initiale).

Les lignes 25 à 44 effectuent l'affichage correspondant au résultat trouvé.

POUR JOUER :

On lance le programme par l'ordre RUN, puis on introduit les dimensions de la grille (carrée) et l'on frappe RET.

Dès l'affichage suivant (DONNEZ X), on entre la coordonnée X supposée, suivie de RET.

On obtient ensuite l'affichage DONNEZ Y, et l'on entre Y suivi de RET.

On obtient en réponse un texte, qui est variable selon que l'on sera plus ou moins loin de l'objet cherché (cela va de COMPLETEMENT GELE à BRULANT). Lorsque l'on a trouvé, on obtient l'affichage du nombre d'essais effectués.

Pour sortir du programme, il faudra taper la combinaison CSA.

EXTENSIONS POSSIBLES :

Si l'on maîtrise l'utilisation de la couleur, il sera intéressant de trouver un moyen plus visuel pour symboliser la distance, par exemple des pavés allant du rouge au bleu, aux emplacements essayés.

On peut aussi travailler dans un espace à trois dimensions, cacher plusieurs objets, etc.

```

1  Rem CHAUD, FROID
2  Fullw 2
3  Clearw 2
4  Input "DIMENSION DE LA GRILLE :";D
5  X=Random(D)+1
6  Y=Random(D)+1
7  Clearw 2
8  S=0
9  Do
10   Deftext 1
11   Sound 1,0
12   Input "DONNEZ X :";A
13   Input "DONNEZ Y :";B
14   S=S+1
15   T=Abs(X-A)+Abs(Y-B)
16   Exit If T=0
17   Print "VOUS ETES ";
18   N=Int(Sqr(D)/T)+1
19   N=N-(5-N)*(N>5)
20   Deftext N
21   If T>Sqr(D) Then
22     Print "COMPLETEMENT GELE"
23     Sound 1,8,4,2,25
24   Else
25     If N=1 Then
26       Print "A PEINE TIEDE"
27       Sound 1,8,2,3,25
28     Endif
29     If N=2 Then
30       Print "TIEDE"
31       Sound 1,8,12,3,25
32     Endif
33     If N=3 Then
34       Print "CHAUD"
35       Sound 1,8,3,4,25
36     Endif
37     If N=4 Then
38       Print "TRES CHAUD"
39       Sound 1,8,2,5,25
40     Endif
41     If N=5 Then
42       Print "BRULANT"
43       Sound 1,8,10,5,25
44     Endif
45   Endif
46 Loop
47 Print
48 Sound 1,8,4,6,25
49 Print "VOUS AVEZ TROUVE EN ";S;" ESSAIS"
50 Sound 1,0
51 R$=Input$(1)

```


Devine un chiffre

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : lancer de dés.

Difficulté du jeu : facile.

LE JEU :

C'est un jeu de hasard, de conception très simple, mais néanmoins assez passionnant : le joueur introduit sa mise, et parie sur un chiffre, de 1 à 6 ; trois dés sont alors lancés ; si le chiffre joué apparaît deux fois, il double sa mise ; s'il apparaît trois fois, il triple sa mise.

LE PROGRAMME :

Le tirage des dés est réalisé par un sous-programme placé en lignes 41 à 45.

L'affichage est réalisé par les lignes 17 et 19.

Les lignes 22 à 24 calculent le gain dans les diverses configurations possibles.

La totalisation est effectuée en ligne 31 pour les gains, et en ligne 27 pour les pertes.

La variable R\$, testée en ligne 39, détermine la poursuite éventuelle du jeu.

POUR JOUER :

Le lancement du programme par RUN provoque un premier affichage VOTRE MISE ; on répond par un nombre suivi de RET. On obtient alors un deuxième affichage DONNEZ UN CHIFFRE DE 1 A 6 ; on entre un chiffre suivi de RET ; les 3 dés, repérés par les lettres A, B, C s'affichent alors, ainsi qu'un message indiquant votre gain ou votre perte.

Puis une indication sur votre total actuel apparaît, ainsi que la question ON CONTINUE (O/N) ; taper O pour continuer, N pour arrêter.

On peut arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra facilement introduire des contrôles (mise minimale, coup proposé de 1 à 6, etc.). On pourra aussi dessiner des dés avec un graphisme plus élaboré, introduire la possibilité de jouer à plusieurs, et même faire participer le ST.

```

1  Rem DEVINE UN CHIFFRE
2  Fullw 2
3  G=0
4  Repeat
5      Clearw 2
6      Input " VOTRE MISE :";N
7      Print
8      Input "DONNEZ UN CHIFFRE DE 1 A 6 :";D
9      Gosub Tirage
10     A=X
11     Gosub Tirage
12     B=X
13     Gosub Tirage
14     Print
15     Print
16     Deftext 3
17     Print "          A          B          C"
18     Deftext 2
19     Print "          ";A;"          ";B;"          ";X
20     Deftext 1
21     T=0
22     T=T-N*(D=A)
23     T=T-N*(D=B)
24     T=T-N*(D=X)
25     If T=0 Then
26         Print "PERDU"
27         G=G-N
28     Else
29         Print
30         Print "VOUS GAGNEZ ";T
31         G=G+T
32     Endif
33     Print
34     Print
35     Print "TOTAL DE VOS GAINS :";G
36     Print
37     Print "ON CONTINUE ( O/N ) :";
38     R$=Upper$(Input$(1))
39 Until R$="N"
40 End
41 Procedure Tirage
42     X=Random(6)+1
43     Sound 1,10,2*X,4,20
44     Sound 1,0
45 Return

```

Le jeu de Wytthof

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion (deux joueurs).

Difficulté du jeu : moyenne.

LE JEU :

On dispose d'une grille carrée de 10 cases sur 10 cases. Un point apparaît, en dehors de la diagonale et les joueurs devront le déplacer à chaque tour, suivant seulement trois directions : droite, bas et diagonale, bas et droite. Le premier joueur qui arrive en (9,9) sera considéré comme gagnant, ou perdant, selon la convention faite au début.

LE PROGRAMME :

Les lignes 6 à 9 permettent de définir une position de départ, en dehors de la diagonale.

Les boucles emboîtées I (10-15) et J (12-14) dessinent le terrain de jeu.

La ligne 19 imprime la position du point mobile.

Le test de fin est effectué en ligne 27.

Le changement de joueur est assuré par la procédure change (51-53).

Les directions de déplacement disponibles, 1, 2 et 3, sont décodées par trois sous-programmes qui commencent aux lignes 32, 39 et 44.

POUR JOUER :

Taper RUN pour lancer l'exécution du programme. Le terrain de jeu s'affiche, ainsi que la position de départ.

Puis on obtient l'affichage JOUEUR 1 et en dessous la question DIRECTION (1, 2, 3), à laquelle on répond en tapant l'un de ces trois nombres. C'est ensuite au tour du joueur 2.

Le jeu se termine lorsque l'un des joueurs arrive en case (9, 9).

On arrête en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut tout d'abord améliorer les contrôles, améliorer ensuite l'affichage (graphisme, couleur), étendre le jeu à plusieurs joueurs avec un terrain agrandi, introduire un monstre qui va essayer de capturer un des joueurs, prévoir un programme qui permettrait au ST de jouer, etc.

```

1  Rem LE JEU DE WYTHHOF
2  Fullw 2
3  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
4  Clearw 2
5  K=1-0.5*(R<>1)
6  X=Random(5)
7  Repeat
8    Y=Random(5)
9  Until Y<>X
10 For I=0 To 9
11   Print At(2*K*I+4,2);I;At(1,I+3);I;
12   For J=0 To 9
13     Print At(2*K*J+3,I+3);" .";
14   Next J
15 Next I
16 H=1
17 Repeat
18   Deftext 1-H*(R=2)-(2*H-1)*(R=1)
19   Print At(2*K*X+3,Y+3);" *";
20   Deftext 1
21   Print At(1,13+3*K);"JOUEUR ";H;Spc(36)
22   Sound 1,10,4*H,5,8
23   Sound 1,0
24   Print At(10,13+3*K);"- DIRECTION (1,2,3) :";
25   D=Val(Input$(1))
26   On D Gosub Bas,Diago,Droite
27 Until X=9 And Y=9
28 Print At(2*K*X+3,Y+3);" *"
29 Print At(1,15);"C'est fini...."
30 R#=Input$(1)
31 End
32 Procedure Droite
33   X=X+1
34   If X>9 Then
35     X=9
36   Endif
37   Gosub Change
38 Return
39 Procedure Diago
40   Gosub Droite
41   Gosub Bas
42   Gosub Change
43 Return
44 Procedure Bas
45   Y=Y+1
46   If Y>9 Then
47     Y=9
48   Endif
49   Gosub Change
50 Return
51 Procedure Change
52   H=2-Abs(H-1)
53 Return

```

Golf

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : solitaire.

Difficulté du jeu : assez facile.

LE JEU :

Il s'agit d'essayer de réaliser, avec le moins de coups possible, un parcours de golf de 18 trous. Pour cela, il vous faut frapper la balle avec une force plus ou moins grande, jusqu'à obtenir sa chute dans le trou.

Mais attention, il peut y avoir un certain aléa sur chaque coup, et vous serez amené souvent à revenir en arrière (il faudra alors faire précéder l'indication de la force par un signe -).

LE PROGRAMME :

La ligne 10 permet de définir la position du trou, et la ligne 14 dessine le terrain de jeu.

Le déplacement à réaliser est calculé en ligne 22.

La boucle I (31-36) réalise le déplacement de la balle.

La ligne 37 regarde si le coup est, ou non, gagnant ; s'il l'est, on recommence jusqu'à ce que l'on ait réalisé 18 trous [boucle S (7-42) ; fin de boucle en 44, avec affichage de fin].

POUR JOUER :

On obtient l'affichage du terrain en frappant RUN ainsi que le dessin de la balle sur sa position de départ, et le texte : TROU 1. FORCE ; on doit alors répondre en donnant la force désirée (un nombre entre 1 et 300...).

Dès que l'on aura tapé RET, la balle recommence à se déplacer ; elle peut alors soit s'arrêter avant le trou (auquel cas on recommence), soit tomber dedans (on passe alors au trou suivant), soit le dépasser : on devra alors frapper avec une force négative.

Le jeu se termine avec l'affichage 18 TROUS EN ... COUPS.

On arrête en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera possible d'étendre le jeu pour faire participer plusieurs joueurs, d'améliorer la qualité graphique, d'utiliser une représentation spatiale du terrain (se référer à certains jeux d'arcade), d'avoir un terrain plus accidenté, des obstacles, de changer de club...

```

1  Rem GOLF
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  E=0
6  K=1-(R<>1)
7  For S=1 To 18
8      Clearw 2
9      X=0
10     T=Random(26*K)+6+3*K
11     Repeat
12         Inc E
13         Deftext 3
14         Print At(1,6);String$(38*K,240);
15         Repeat
16             Deftext 2
17             Print At(T+1,6);" ";At(X+1,5);"0";
18             Deftext 1
19             Print At(10,10);Spc(30);
20             Print At(1,10);"TROU ";S;"    FORCE :";
21             Input R
22             F=Int(R*0.16-(Abs(R)*Sqr(Rnd+0.1)^2)/1000)
23         Until F<>0
24         If F>35*K-X Then
25             F=35*K-X
26         Endif
27         If F<=-X Then
28             F=-X
29         Endif
30         Deftext 2
31         For I=Sgn(F) To F Step Sgn(F)
32             X=X+Sgn(F)
33             Print At(X-Sgn(F)+1,5);" ";At(X+1,5);"0";
34             Sound 1,8,4,4,1
35             Sound 1,0
36         Next I
37     Until X=T
38     Print At(X+1,5);" ";At(X+1,6);"0";
39     Sound 1,8,4,3,10
40     Pause 8
41     Sound 1,0
42 Next S
43 Deftext 1
44 Print At(1,14);"18 TROUS EN ";E;" COUPS"
45 R$=Input$(1)

```

Pair-impair

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : lancer de dés.

Difficulté du jeu : facile.

LE JEU :

C'est un jeu de hasard, pour un ou plusieurs joueurs. Le ST lance simultanément deux dés, non visibles et pose trois questions : quelle est la parité du total, quel est le total, et enfin donner la valeur de l'un des dés.

On peut ensuite soit poursuivre le jeu, soit l'interrompre à tout moment.

LE PROGRAMME :

Les dés sont tirés par un sous-programme placé aux lignes 46 à 50.

La ligne 9 tient le compte du nombre de parties, et les lignes 19, 27 et 32 comptabilisent les points marqués.

La parité du total est déterminée par les lignes 14 et 15 (la ligne 15 compare avec la réponse du joueur).

Les lignes 40 et 42 réalisent l'affichage de fin de partie : nombre de points, nombre de parties jouées.

La ligne 44 provoque l'arrêt si l'on répond N à la question ON CONTINUE.

POUR JOUER :

On lance le programme par l'instruction RUN.

Il suffit ensuite de répondre aux diverses questions posées, en suivant scrupuleusement les indications données et en tapant RET après chaque réponse.

Lorsque l'on a terminé une partie, on obtient l'affichage du total actuel et du nombre de parties sous la forme de, par exemple, 6 SUR 10.

On obtient alors l'affichage de la question ON CONTINUE (O/N) et l'on devra répondre N pour arrêter.

EXTENSIONS POSSIBLES :

Si l'on désire jouer à plusieurs, il sera intéressant de prévoir la possibilité de miser, et d'ajouter une gestion du score réalisé par chacun des joueurs.

On pourra aussi prévoir une amélioration sur le plan de la présentation (utilisation de la couleur, etc.).

```

1  Rem PAIR-IMPAIR
2  Fullw 2
3  T=0
4  R=0
5  Repeat
6      Gosub Tirage
7      A=X
8      Gosub Tirage
9      R=R+1
10     S=A+X
11     Clearw 2
12     Input "TOTAL PAIR (0) OU IMPAIR (1) :";N
13     Print
14     P=S/2-Int(S/2)
15     If N<>P And N-0.5<>P Then
16         Print "FAUX"
17     Else
18         Print "EXACT"
19         T=T+1
20     Endif
21     Print
22     Input "QUEL EST LE TOTAL PRESUME :";N
23     If N<>S Then
24         Print
25         Print "PERDU"
26     Else
27         T=T+2
28         Print
29         Input "BRAVO. DONNEZ UN DE :";N
30         Print
31         If N=A Or N=S-A Then
32             T=T+3
33             Print "BRAVO"
34         Else
35             Print "PERDU"
36         Endif
37     Endif
38     Pause 7
39     Print
40     Print "VOUS AVEZ ";T;" POINTS EN ";R;" ESSAIS"
41     Print
42     Print "ON CONTINUE ( O/N ):";
43     R$=Upper$(Input$(1))
44 Until R$="N"
45 End
46 Procedure Tirage
47     X=Random(6)+1
48     Sound 1,10,2*X,6,25
49     Sound 1,0
50 Return

```


Jackpot

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : hasard (un joueur).

Difficulté du jeu : facile.

LE JEU :

C'est une variante simplifiée d'un jeu de hasard classique, dans lequel le joueur doit faire une mise. S'il obtient certaines combinaisons de symboles, il gagne (plus ou moins, selon sa mise et la combinaison obtenue).

Ici, les symboles sont remplacés par des chiffres, et le fait d'avoir trois chiffres identiques permet de gagner. Il existe de plus une combinaison particulière qui permet de remporter le Jackpot.

LE PROGRAMME :

La boucle I (10-12) permet le dessin de la machine, les lignes 13 et 14 affichant les emplacements (vides) des chiffres.

La mise sera entrée en ligne 15, et testée en ligne 16 : si on mise 0, la partie est terminée (départ vers la ligne 35).

Les lignes 17, 19 et 21 permettent le tirage des 3 chiffres, en appelant un sous-programme qui se trouve aux lignes 38 à 42.

Le résultat est affiché par la ligne 23, puis testé en ligne 24, et en ligne 25, si l'on a gagné.

POUR JOUER :

Lancer le programme par RUN ; à l'affichage de la question POT DE DEBUT, on répond en entrant un nombre suffisamment élevé, sur lequel vos mises seront décomptées.

Puis on tape RET, et on entre une première mise (différente de 0), suivie de RET.

Les trois chiffres tirés sont alors affichés un bref instant, ainsi que la mise faite, et le résultat obtenu (gain ou perte).

On peut ensuite miser de nouveau, faire RET et ainsi de suite.

Pour terminer, il suffit de répondre 0 à la question VOTRE MISE.

Le restant de votre pot initial sera alors affiché.

On sortira en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra envisager d'améliorer sérieusement les contrôles, utiliser la couleur et des possibilités graphiques plus élaborées pour simuler davantage un véritable Jackpot.

```

1  Rem JACKPOT
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";K
5  F=-K*(K<>3)-1.5*(K=3)
6  Input "POT DE DEBUT DE PARTIE :";P
7  Do
8      Clearw 2
9      Deftext 1
10     For I=1 To 3
11         Print At(8*F,5+I);String$(12*F+9,35)
12     Next I
13     Print At(4+8*F,7);"      ";At(5+12*F,7);"      "
14     Print At(6+16*F,7);"      ";At(1,12);"VOTRE MISE :";
15     Input M
16     Exit If M=0 Or P-M<0
17     Gosub Tirage
18     A=X
19     Gosub Tirage
20     B=X
21     Gosub Tirage
22     Deftext 2
23     Print At(5+8*F,7);A;At(6+12*F,7);B;At(7+16*F,7);X
24     If A=B And B=X Then
25         N=-M*50+M*450*(A=7)
26     Endif
27     N=-M
28     Deftext 3
29     Print At(1,14);"MISE :";M;" > RESULTAT :";N;"      "
30     Deftext 1
31     Print At(1,16);"TAPER - RETURN - POUR CONTINUER";
32     R$=Input$(1)
33     P=P+N
34 Loop
35 Print
36 Print "VOUS AVEZ ";P;" POINTS"
37 Print "A BIENTOT...."
38 Procedure Tirage
39     X=Random(9)+1
40     Sound 1,12,X,5,10
41     Sound 1,0
42 Return

```

Calendrier perpétuel

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : gadget publicitaire.

Difficulté du jeu : aucune.

LE JEU :

Ce programme vous permettra de confectionner tous les ans votre calendrier ; vous pourrez ainsi vous en servir pour envoyer à vos amis un calendrier du mois de leur naissance, ou pour leur souhaiter une bonne fête, etc.

LE PROGRAMME :

Les lignes 14 et 16 vous permettront d'entrer les mois et l'année.

Les lignes 17, 18 et 19 calculent le numéro du premier jour du mois.

La ligne 20 calcule le nombre de jours du mois (il n'est pas tenu compte ici des 29 février, pour simplifier).

La boucle A (26-32) réalise l'impression du calendrier proprement dit.

POUR JOUER :

Après avoir lancé le programme par l'instruction RUN, on devra introduire l'année, sous la forme, par exemple 1985, suivie de RET.

On obtient alors la question MOIS, à laquelle il convient de répondre par le numéro du mois, suivant l'ordre usuel (1 pour JANVIER, 2 pour FEVRIER, etc.).

On fait ensuite RET, et le calendrier est alors affiché sur l'écran.

On terminera en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra sans difficulté prévoir l'affichage des 29 février, prévoir une option permettant l'affichage pour une année complète, prévoir l'introduction du nom des mois en clair, etc.

```

1  Rem CALENDRIER PERPETUEL
2  Fullw 2
3  Clearw 2
4  A$="DIMANCHE LUNDI     MARDI     MERCREDI  "
5  A$=A$+"JEUDI     VENDREDI SAMEDI  "
6  K=2.25
7  Input "RESOLUTION 1:BASSE 2:MOYEN 3:HAUTE :";Re
8  Clearw 2
9  If Re=1 Then
10   A$="DIM LUN MAR MER JEU VEN SAM"
11   K=1
12 Endif
13 Deftext 1
14 Print At(1,1);"QUELLE ANNEE :";
15 Input A
16 Input "QUEL MOIS      :";M
17 D=Int(365.25*(A+(M<3)))+1
18 D=D+Int(30.6*(M+1-(M<3)*12))-Int((Int((A+(M<3))/10
   0)-7)*0.75)
19 D=D-7*Int(D/7)
20 M=31+(M=4 Or M=6 Or M=9 Or M=11)+(M=2)*3
21 Deftext 3
22 Print
23 Print A$
24 Print
25 Deftext 2
26 For A=1 To M
27   Print Tab(K+4*(D+A-1)*K);A;
28   If D+A-1=6 Then
29     Print
30     D=D-7
31   Endif
32 Next A
33 R$=Input$(1)

```

Le jeu du 21

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu contre le ST.

Difficulté du jeu : facile.

LE JEU :

On devra entrer une mise au début du jeu, et on pourra lancer les dés en tapant 1 ; à chaque lancer, on obtiendra un nouveau total, qui augmentera peu à peu, et se rapprochera de 21. On devra essayer de se rapprocher le plus possible de 21, sans dépasser ce nombre. Lorsque l'on décidera de s'arrêter, il suffira de taper 0 pour laisser jouer le ST, qui essaiera alors d'avoir un meilleur score.

A la fin de chaque partie, le gain sera affiché ainsi que le total des gains (ou des pertes).

LE PROGRAMME :

Les lignes 11 et 12 réalisent le contrôle de celui qui doit jouer (joueur ou ST).

Le sous-programme (49-53) lance les dés.

Les lignes 18-19 calculent le total fait par le joueur et l'affichent.

Le total du ST est calculé en ligne 28 et affiché par la ligne 29.

La ligne 32 contrôle la fin du jeu pour le ST.

Les lignes 36 et 37 affichent les résultats, et la ligne 41 cherche quel est le gagnant.

Le gain (ou la perte) sera affiché par la ligne 41, et le total par la ligne 43.

La ligne 45 contrôle la fin de partie.

POUR JOUER :

Taper RUN pour lancer le jeu.

A la question VOTRE MISE, entrer la mise initiale, suivie de RET, et taper 1 pour lancer le dé. Le premier tirage est affiché sous la forme VOUS AVEZ... et on devra refaire 1, tant que l'on désire recommencer à tirer.

Dès que l'on a terminé, on tape 0 pour laisser jouer le ST (qui tient ici le rôle de la banque).

Le gain (ou la perte, s'il y a le signe -) sera affiché en fin de jeu, ainsi que le total des gains (ou pertes) réalisé depuis le début. Si l'on désire arrêter de jouer, on devra taper N en réponse à la question ON CONTINUE (O/N).

EXTENSIONS POSSIBLES :

Ce jeu sera évidemment plus intéressant s'il est pratiqué à plusieurs, et il sera aisé de modifier le programme en conséquence. On pourra aussi dessiner le dé, ou, mieux, utiliser des cartes, ce qui permettra alors de faire un jeu de Black Jack.

```

1  Rem JEU DU 21
2  Fullw 2
3  G=0
4  Do
5      Clearw 2
6      T=0
7      Input "VOTRE MISE :";M
8      Do
9          Deftext 1
10         Print
11         Print "1 :TIRER, 0 :FINIR "
12         R=Val(Input$(1))
13         St=Crslin
14         Exit If R=0
15         Gosub Tirage
16         Deftext 2
17         Print "TIR :";X;
18         T=T+X
19         Print Tab(11);"TOTAL :";T;
20     Loop
21     Deftext 3
22     Print At(25,1);"LA BANQUE JOUE"
23     S=0
24     K=4
25     Do
26         Gosub Tirage
27         Print At(23,K);"TIR :";X;
28         S=S+X
29         Print "  TOTAL :";S
30         Pause 3
31         K=K+2
32         Exit If S>T Or S>17
33     Loop
34     Dt=Max(Crslin,St)+1
35     Deftext 1
36     Print At(1,Dt);"VOTRE TOTAL EST DE ";T
37     Print "CELUI DE LA BANQUE EST DE ";S
38     If S<=21 And (S>=T Or T>21) Then
39         M=-M
40     Endif
41     Print "VOTRE GAIN EST DE ";M
42     G=G+M
43     Print "VOTRE GAIN TOTAL SE MONTE A ";G
44     Print "ON CONTINUE ( O/N ):";
45     R$=Upper$(Input$(1))
46     Exit If R$="N"
47 Loop
48 End
49 Procedure Tirage
50     X=Random(6)+1
51     Sound 1,10,2*X,4,30
52     Sound 1,0
53 Return

```

Solfège

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion.

Difficulté du jeu : facile.

LE JEU :

Ce programme vous propose une initiation élémentaire au solfège, en vous permettant d'apprendre à reconnaître facilement les différentes notes, qui apparaissent les unes après les autres sur une portée ; après l'affichage de la note, vous devez entrer son nom en clair et votre score sera affiché en permanence.

Le programme est initialement prévu pour la clé de sol, mais il sera très facilement adaptable à toute autre clé.

LE PROGRAMME :

La boucle I (11-13) dessine une portée.

La note est choisie en ligne 14 et affichée en lignes 16 à 20.

La ligne 22 affiche le score.

La ligne 22 affiche le nom de la clé (ici clé de sol) et vous interroge.

La ligne 28 s'occupe de la fin de la partie.

Les lignes 29 et 30 effectuent le décodage de la note entrée.

La ligne 32 contrôle le résultat.

POUR JOUER :

Après avoir lancé le programme par RUN, la portée sera affichée, ainsi que la première note.

Vous répondrez à la question NOTE en donnant le nom en clair de la note (do, ré, mi, fa, sol, la ou si), suivi de RET.

Le score vous indiquera si c'est exact ou non, et une nouvelle note sera affichée.

Pour arrêter, il suffira de répondre 0 suivi de RET, à la question NOTE.

EXTENSIONS POSSIBLES :

On pourra tout d'abord facilement modifier le programme pour lui permettre de vous faire travailler les différentes clés. On pourra aussi améliorer le graphisme, introduire la couleur, le son (lorsque l'on a répondu juste, on entendra la note, par exemple). On pourra aussi réaliser suivant le même principe un jeu vous permettant de reconnaître des accords, etc.

```

1  Rem SOLFEGE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  K=1-(R<>1)
6  S=0
7  P=0
8  Do
9      Clearw 2
10     Deftext 1-2*(R=2)-3*(R=1)
11     For I=1 To 5
12         Print At(1,2*I+1);String$(36*K,95)
13     Next I
14     X=3+Random(12)
15     Deftext 1-(R<3)
16     Print At(17*K+2,X-3);Chr$(124)
17     Print At(17*K+2,X-2);Chr$(124)
18     Print At(17*K,X-1);Chr$(47);Chr$(92)
19     Print At(17*K+2,X-1);Chr$(124)
20     Print At(17*K,X);Chr$(92);Chr$(47)
21     Deftext 1
22     Print At(10,16);"SCORE ";S;" EN ";P;" ESSAIS"
23     Repeat
24         Print At(1,17);"CLE DE SOL- QUELLE EST LA NOTE
25         :";
26         Input R$
27         R$=Upper$(R$)
28         Until R$<>""
29         Exit If R$="D" Or R$="O"
30         Y=(3 And R$="SOL")+(4 And R$="FA")+(5 And R$="MI
31         ")+(6 And R$="RE")
32         Y=Y+(7 And R$="DO")+(8 And R$="SI")+(9 And R$="L
33         A")
34         P=P+1
35         S=S-(Y=X)-(Y=X-7)
36     Loop

```


Le jeu de la fève

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : réflexion (deux joueurs).

Difficulté du jeu : moyenne.

LE JEU :

Présentant une certaine analogie avec le jeu de WYTTHOF, cette version offre en fait quatre possibilités de déplacement ; la position de départ est ici fixée en $(0, 0)$, et pourrait éventuellement être modifiée.

Les directions de déplacement disponibles sont : droite, bas, diagonale bas et droite, diagonale haut et droite.

On joue en principe avec la règle : le premier arrivé en $(9, 9)$ a perdu.

LE PROGRAMME :

Les boucles emboîtées I (7-12) et J (9-11) dessinent le terrain de jeu.

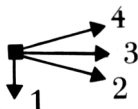
La ligne 19 réalise le test de fin de partie.

La ligne 24 permet d'exécuter l'un des sous-programmes de déplacement.

La ligne 25 assure le changement de joueur.

POUR JOUER :

On lance le programme par RUN, et le terrain de jeu s'affiche, ainsi que la question DIRECTION (1, 2, 3, 4) ; les quatre directions proposées correspondent aux déplacements suivants :



Le joueur 1 entre son déplacement.

Le mouvement correspondant est réalisé, puis on obtient l'affichage JOUEUR 2 et DIRECTION.

Le joueur 2 procède de même, et ainsi de suite, jusqu'à ce que l'un des joueurs arrive en $(9, 9)$, auquel cas le jeu se termine.

On devra taper la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On peut imaginer d'introduire en début de partie des cases pièges (chacun essayant alors de faire venir l'autre sur l'une de ces cases).

On peut aussi jouer avec plusieurs symboles, si possible de couleurs différentes, déplacés simultanément, envisager des terrains de jeu triangulaires, aux dimensions arbitraires, etc.

```

1  Rem LE JEU DE LA FEVE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  Clearw 2
6  K=1-0.5*(R<>1)
7  For I=0 To 9
8      Print At(2*K*I+4,2);I;At(1,I+3);I;
9      For J=0 To 9
10         Print At(2*K*J+3,I+3);" .";
11     Next J
12 Next I
13 X=0
14 Y=0
15 J=1
16 Do
17     Deftext 1+J
18     Print At(2*K*X+3,Y+3);" *";
19     Exit If X=9 And Y=9
20     Print At(1,12+3*K);"JOUEUR ";J;Spc(26)
21     Deftext 1
22     Print At(11,12+3*K);"DIRECTION (1,2,3,4) :";
23     D=Val(Input$(1))
24     On D GOSUB Bas,Diagbas,Droit,Diaghaut
25     J=Abs(3-J)
26 Loop
27 Print At(1,17);"LE JOUEUR ";J;" GAGNE"
28 R#=Input$(1)
29 Procedure Bas
30     Y=Y+1
31     Y=Y+(Y>9)
32 Return
33 Procedure Diagbas
34     Gosub Bas
35     Gosub Droit
36 Return
37 Procedure Droit
38     X=X+1
39     X=X+(X>9)
40 Return
41 Procedure Diaghaut
42     Gosub Droit
43     Y=Y-1
44     Y=Y-(Y<0)
45 Return

```

Rebonds

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu d'adresse.

Difficulté du jeu : moyenne.

LE JEU :

Il utilise l'idée de la simulation d'un billard à bandes.

De part et d'autre de l'écran se trouvent la balle et le but visé : une série de chiffres et de lettres, en haut de l'écran, permettent de choisir l'endroit où l'on désire rebondir.

La balle va alors partir vers le haut de l'écran (vers le bas si l'on introduit un signe moins), y rebondir, revenir éventuellement frapper le bas, rebondir, et ainsi de suite jusqu'à atteindre la ligne de but ; si l'on est suffisamment près de la cible, on a gagné et le nombre de rebonds sera alors affiché.

LE PROGRAMME :

La ligne 6 imprime les chiffres et les lettres en haut de l'écran.

Les lignes 7 et 8 définissent les positions initiales de la balle et de la cible.

La ligne 10 effectue leur impression.

Le symbole correspondant à la direction choisie est entré en ligne 13.

La boucle J (17-30) réalise le déplacement de la balle.

La ligne 28 imprime la nouvelle position.

Le changement de direction, lors de la rencontre d'un bord, est effectué par la ligne 22.

La ligne 33 effectue l'impression de fin de partie.

Le nombre de rebonds est compté en ligne 23.

POUR JOUER :

On lance le programme par RUN.

Le terrain de jeu est affiché, ainsi que les positions initiales de la balle et de la cible. On choisit l'endroit où l'on désire faire rebondir la balle (de 1 à Z), et on l'entre.

La trajectoire est alors affichée, suivie s'il y a lieu de la mention BRAVO ; X REBONDS.

On peut lancer une autre partie en tapant une touche quelconque, et s'arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera très intéressant, en utilisant un graphisme plus fin, de dessiner un véritable billard, avec son tapis vert, et d'indiquer la direction par un segment, symbolisant la queue de billard, se déplaçant autour de la bille. On peut aussi ajouter des trous, mettre plusieurs billes, sonoriser les rebonds, ou les heurts avec d'autres balles, les comptabiliser pour faire une partie à plusieurs joueurs, etc.

```

1  Rem REBONDS
2  Fullw 2
3  Repeat
4    Clearw 2
5    Deftext 3
6    Print At(1,1);" 0123456789ABCDEFGHIJKLMNPOQRSTUVWXYZ"
7    X=Random(14)+3
8    Y=Random(14)+3
9    Deftext 2
10   Print At(39,Y);Chr$(35);At(1,X);Chr$(249)
11   Deftext 1
12   Print At(1,17);"VOTRE COUP :";
13   R$=Upper$(Input$(1))
14   Print R$
15   S=0
16   T=X
17   For J=1 To 38
18     Print At(J,T);" "
19     Do
20       T=T-X/(Asc(R$)-Asc("0")+7*(R$>="A")+4)
21       Exit If T<=16 And T>=1
22       X=-X
23       S=S+1
24       Sound 1,12,4,4,3
25       Sound 1,0
26     Loop
27     Deftext 2
28     Print At(J,T);Chr$(249)
29     Pause 3
30   Next J
31   Deftext 1
32   If Abs(T-Y)<3 Then
33     Print At(1,17);"BRAVO :";S;" REBONDS"
34   Endif
35   Print At(1,18);"UNE AUTRE PARTIE (O/N) :";
36   Input S$
37   Until Upper$(S$)="N"

```

Jeu de l'oie aléatoire

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : hasard (deux joueurs).

Difficulté du jeu : facile.

LE JEU :

Le jeu reprend la très ancienne idée du jeu de l'oie, mais en y ajoutant la possibilité offerte par l'ordinateur de choisir au hasard les cases "à problème".

La règle du jeu est sans grand intérêt, puisque les joueurs n'interviennent à aucun moment dans le déroulement de la partie, qui est entièrement gérée par le ST ; il suffira d'appuyer sur une touche après chaque lancer de dé, et de regarder ce qui se passe.

LE PROGRAMME :

La ligne 16 permet d'appeler le sous-programme 50 à 70 qui réalise l'affichage du terrain de jeu, le contrôle de la position des joueurs et l'affichage de cette position. Cet affichage sera d'ailleurs refait en cours de partie, pour tenir compte du déplacement réellement effectué (qui n'est pas nécessairement celui indiqué par le dé).

La ligne 17 s'occupe du changement de joueur, et effectue le lancer de dé.

La ligne 19 traite le cas où l'un des joueurs termine.

La ligne 27 tire un nombre aléatoire entre 0 et 3, qui servira s'il y a lieu pour déterminer le vrai déplacement du joueur, en envoyant dans l'un des sous-programmes de "chance".

Les lignes 51 et 52 font avancer les pions, et les lignes 55 à 68 redessinent à chaque affichage le terrain de jeu en entier.

POUR JOUER :

On lancera le jeu par RUN. Le terrain de jeu est alors dessiné, puis on voit apparaître l'affichage JOUEUR 1 :...

On devra alors appuyer sur une touche quelconque pour que le coup du joueur numéro 1 soit pris en compte ; ne pas s'inquiéter du temps de réaction, qui pourra sembler un peu long.

Le déplacement indiqué par les dés sera exécuté, puis, au bout de quelque temps, un second déplacement sera éventuellement effectué : c'est le déplacement "CHANCE". On obtiendra alors l'affichage JOUEUR 2 :...

Le joueur 2 appuie à son tour sur une touche quelconque et ainsi de suite...

Le joueur gagnant sera signalé par un affichage BRAVO en fin de partie.

On peut s'arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut agrandir le terrain de jeu, introduire davantage d'aléas, changer leur nature. On pourrait aussi choisir les cases "chance" en début de jeu, ajouter des affichages en conséquence, etc.

```

1  Rem JEU DE L'OIE ALEATOIRE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:AUTR
   E :";R
5  Clearw 2
6  K=35+3*(R=1)
7  X=0
8  Y=0
9  J=1
10 I=X
11 Deftext 1-7*(R=1)
12 For I=0 To K
13   Print At(I+1,4);Chr$(48+I
     -7*(I>9))
14 Next I
15 Do
16   Gosub Dessin
17   J=Abs(J-1)
18   I=Random(6)+1
19   Exit If X=K Or Y=K
20   Deftext 2+J
21   Print At(1,12);"JOUEUR ";
     J+1;
22   Deftext 1
23   Print " - VOTRE COUP :";I
24   Print At(1,15);"POUR JOUE
     R, APPUYER SUR - RETURN -"
25   R$=Input$(1)
26   Gosub Dessin
27   I=Random(4)
28   F=Random(5)
29   On F Gosub Tir1,Tir2,Tir3
     ,Tir4
30 Loop
31 Deftext 1,0,0,16
32 Print At(1,17);"BRAVO !"
33 End

34 Procedure Tir1
35   I=I+3
36 Return
37 Procedure Tir2
38   I=-I
39 Return
40 Procedure Tir3
41   I=2*I
42 Return
43 Procedure Tir4
44   If J=0 Then
45     X=Random(20)
46   Else
47     Y=Random(20)
48   Endif
49 Return
50 Procedure Dessin
51   X=X-(1-J)*I*(X+I<=K)
52   Y=Y-J*I*(Y+I<=K)
53   Sound 1,12,10*J+2,5,20
54   Sound 1,0
55   For I=0 To K
56     Deftext 2
57     Print At(I+1,6);"#";
58     If I=X Then
59       Deftext 1-6*(R=1)
60       Print At(I+1,6);"1";
61     Endif
62     Deftext 3
63     Print At(I+1,7);"#";
64     If I=Y Then
65       Deftext 1-3*(R=1)
66       Print At(I+1,7);"2";
67     Endif
68   Next I
69   Pause 3
70 Return

```


Niveau 2

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Le jeu de saute-mouton	Un jeu de réflexion, pour deux joueurs.
La course au 20	Jeu de réflexion, pour deux joueurs ou plus.
Le jeu de Marienbad	Un jeu de logique pour deux joueurs, dans une version étendue.
Dé truqué	A utiliser, si vous êtes malhonnête, dans d'autres programmes.
Espion	Dites-lui votre amour en code.
Jour de la semaine	Profitez-en pour faire le point sur les anniversaires.
421	Une simulation simplifiée, pour plusieurs joueurs.
Compose	Faire de la musique, c'est d'abord l'écrire.
Tour de cartes	Votre ordinateur a plus d'un tour dans son sac.
Pendu inversé	A l'ordinateur d'essayer de deviner vos mots !
Musique	Révissez vos notions de solfège, grâce à l'ordinateur.
Lettres	Un jeu sur les mots, assez simple.
Machinamot	Laissez l'ordinateur créer des vocables nouveaux.
Tirage d'un dé truqué	Révissez vos connaissances en statistiques.
Jeu de loto	Gagnez sans problème, au prochain tirage.
Le jeu de la vie	Un grand classique, en deux versions : manuelle et automatique.
Un taquin de calcul	Un bon entraînement (pour deux joueurs).
Belote	Pour simuler une donne, avec un jeu de 32 cartes.
Hanoï	Un conseil : commencez petit.
Calculatrice	Un vrai programme de simulation, très réaliste.

Dimensionnement d'un tableau, Recherche d'un élément dans un tableau et dans une chaîne, L'instruction DATA, Manipulation des chaînes

L'INSTRUCTION DIM OU INSTRUCTION DE DIMENSIONNEMENT D'UN TABLEAU

Les structures de tableaux sont très couramment utilisées en programmation, se révélant même indispensables dans un grand nombre de cas.

Si vous voulez, par exemple, entrer vos recettes et dépenses pour tous les jours du mois (on supposera pour simplifier des mois de 30 jours), vous pourriez définir 30 variables appelées PREMIER JOUR, DEUXIEME JOUR, etc. Il est plus facile, et plus rationnel, d'utiliser une boucle I, I variant de 1 à 30, et une variable indicée R(I) pour les recettes, et D(I) pour les dépenses ; ce faisant, vous avez, sans le savoir, utilisé une structure de tableau à une dimension. Ceci ne pourra être admis par le GFA Basic que si vous l'avez auparavant prévenu que vous alliez utiliser un tel tableau, en donnant une instruction **DIM R(30)** pour les recettes, et **DIM D(30)** pour les dépenses.

On pourrait aussi imaginer de conserver dépenses et recettes dans un même tableau, en lui donnant cette fois deux dimensions : par exemple **DIM RD(30,2)** (ou **DIM RD(2,30)**). Ainsi, avec **DIM RD(30,2)**, on pourra convenir que RD(I,1) représente les recettes, et RD(I,2) les dépenses. Des tableaux de plus de deux dimensions sont également possibles.

On peut aussi concevoir des tableaux dont les éléments sont des chaînes de caractères, et non plus des nombres.

Ainsi, **DIM NOM\$(12)** réserve 12 emplacements en mémoire, pour des chaînes.

En GFA Basic, il est impératif de dimensionner tous les tableaux, même s'ils ont moins de 10 éléments.

LA RECHERCHE D'UN ELEMENT DANS UN TABLEAU : R(I)

Si l'on a un tableau à une dimension, numérique ou alphanumérique, défini par une instruction comme **DIM R(30)** ou **DIM N\$(20)**, le I^{ème} élément de ce tableau est repéré par **R(I)** ou **N\$(I)**. Il sera alors manipulé exactement comme une variable ordinaire (affectation, calculs, tests, etc.).

Ainsi **INPUT R(3)** permet d'entrer à la main la valeur du 3^e élément numérique du tableau R, **PRINT R\$(5)** permet l'affichage du 5^e élément du tableau alphanumérique R\$, ...

LA RECHERCHE D'UN ELEMENT DANS UNE CHAÎNE : MID\$

Pour extraire le I^{ème} élément de la chaîne alphanumérique A\$, on utilisera l'instruction **MID\$**.

Ainsi, si A\$="BONJOUR", **MID\$(A\$,3,1)** donne le caractère N ; de plus **MID\$(A\$,3,3)** donne la sous-chaîne "NJO".

Nota : **MID\$(A\$,3)** renvoie la fin de la chaîne : "NJOUR".

LE TRAITEMENT APPROFONDI DES CHAINES DE CARACTERES : MID\$, LEFT\$, RIGHT\$

Pour faciliter les manipulations sur les chaînes, il est possible d'utiliser deux fonctions supplémentaires : **LEFT\$** et **RIGHT\$**.

LEFT\$(A\$,1) permet d'extraire les I premiers caractères à partir de la gauche du mot A\$.

Ainsi, **LEFT\$(A\$,2)**="MO", si A\$="MONSIEUR" (on notera que **MID\$(A\$,1,2)** donne le même résultat).

RIGHT\$(A\$,J) permet d'extraire les J derniers caractères (on part de la droite du mot A\$).

Toujours avec **A\$="MONSIEUR"**, on obtient **RIGHT\$(A\$,3)="EUR"**.

Le GFA Basic permet en outre le remplacement direct à l'intérieur d'une chaîne, par une affectation du type **MID\$(xx,xx,xx)="..."**

Ainsi, il est possible d'écrire **MID\$(A\$,2,1)="-"**, pour remplacer le second caractère de la chaîne A\$ par un signe -. Cela permet d'alléger considérablement certaines écritures.

LE TRAITEMENT DE DONNEES MULTIPLES : **INSTRUCTIONS READ, DATA, RESTORE**

Pour remplir facilement un tableau, numérique ou alphanumérique, il sera intéressant d'utiliser des lignes **DATA**, qui contiennent les données à lire, inscrites les unes à la suite des autres, et séparées par une virgule.

Ainsi, on pourra écrire :

DATA LEON, JULES, 13, rue MARTIN...

Ces données peuvent être lues dans une boucle FOR...NEXT, au moyen de l'instruction **READ**. Il doit y avoir correspondance étroite entre les variables utilisées dans **READ**, et le type de donnée à lire. On peut recommencer la lecture au début de **DATA**, avec l'instruction **RESTORE**, ou se placer au début d'une ligne de **DATA** repérée par une étiquette, avec l'instruction **RESTORE étiquette**.

Le jeu de saute-mouton

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : facile.

LE JEU :

Bien que l'idée en soit très simple, il ne sera pas forcément évident de découvrir une stratégie convenable. On dispose de 3 moutons d'une couleur, de 3 moutons d'une autre, représentés par des symboles graphiques, et de 8 cases ; les moutons peuvent se déplacer d'une case à une case libre. Plusieurs règles pourront intervenir, selon votre propre estimation de l'intérêt de telle ou telle : soit l'on contraint les moutons à se placer sur la première case disponible, en sautant éventuellement par-dessus les moutons adverses, soit on leur laisse choisir leur case d'arrivée, ce qui peut rendre les parties interminables, à vous de choisir.

LE PROGRAMME :

La boucle I (10-13) initialise une table A (I), qui contient les positions des moutons. La boucle I (17-24) dessine le terrain de jeu et les moutons.

Les lignes 28 à 35 s'occupent du texte et des entrées.

La ligne 36 contrôle la validité du coup. Le changement de joueur est réalisé en ligne 39.

POUR JOUER :

On commence l'exécution du programme par RUN. On obtient alors un affichage JOUEUR 1, qui indique le numéro du joueur, suivi de la question DEBUT, FIN.

On répond en indiquant le numéro du pion à déplacer, puis le numéro de la case d'arrivée, tous deux séparés par une virgule, suivi de RET.

Le déplacement est alors réalisé ; puis le deuxième joueur procède de la même façon. Le gagnant est en principe celui qui réussit à placer ses trois moutons côte à côte sur les emplacements initialement occupés par l'adversaire.

On arrête la partie à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On commencera par rajouter un contrôle du joueur gagnant.

On peut également améliorer la présentation en dessinant de vrais moutons, rendre le jeu plus attractif en les numérotant, et en imposant un certain ordre à l'arrivée.

Enfin, on pourra définir une stratégie pour l'ordinateur, en figeant les règles.

```

1  Rem LE JEU DE SAUTE-MOUTON
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  Dim A(8)
6  A=2
7  B=3-(R=1)
8  A(1)=0
9  A(8)=0
10 For I=1 To 3
11   A(I+1)=1
12   A(I+4)=-1
13 Next I
14 H=1
15 Clearw 2
16 Do
17   For I=1 To 8
18     Deftext 1
19     Print At(4*I-3,8);I
20     Sound 1,10,I,4,8
21     Sound 1,0
22     Deftext -A*(A(I)=1)-B*(A(I)=-1)
23     Print At(4*I-3,6);Chr$(32-47*(A(I)=1)-56*(A(I)
=-1))
24   Next I
25   J=1-(H=-1)
26   K=(A And J=1)+(B And J=2)
27   Repeat
28     Print At(1,15);Spc(30)
29     Deftext 1
30     Print At(1,13);"JOUEUR :";
31     Deftext K
32     Print Chr$(48+J)
33     Deftext 1
34     Print
35     Input "DEBUT, FIN :";D,F
36   Until A(D)=H And A(F)=0
37   A(D)=0
38   A(F)=H
39   H=-H
40 Loop

```

La course au 20

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : logique (deux joueurs, ou plus).

Difficulté du jeu : moyenne.

LE JEU :

Il est possible d'imaginer pour ce jeu diverses règles, et nous n'avons retenu ici que la règle usuelle pour deux joueurs, qui en fait un jeu apparenté aux jeux de déplacement, ou d'allumettes : 4 tas de 20 étoiles vont apparaître en début de partie, et chaque joueur peut prendre autant d'étoiles qu'il le désire ; mais cela dans un seul tas à la fois. Celui qui prend la dernière étoile est considéré comme perdant.

LE PROGRAMME :

La boucle I (5-11) dessine 4 rangées de 20 étoiles.

La variable indicée F (I), définie en ligne 4, servira à enregistrer le nombre d'étoiles prises dans chaque rangée.

La variable T contiendra le nombre total d'étoiles enlevées (ligne 13).

Les lignes 23 et 28 vont contrôler qu'il reste suffisamment d'étoiles sur la ligne indiquée.

La boucle J (33-37) supprime les étoiles ; la ligne 39 teste la fin de partie.

POUR JOUER :

Lancer le programme par RUN ; le terrain de jeu s'affiche, puis les inscriptions JOUEUR 1 et LIGNES, puis CASES. On répond en entrant un numéro de ligne (entre 1 et 4) suivi de RET, et le nombre d'étoiles à enlever suivi de RET. Les étoiles sont alors supprimées, et c'est au tour du deuxième joueur.

La partie se termine, lorsque la dernière étoile a été enlevée, par l'affichage du joueur gagnant.

On arrête le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Outre une amélioration de la qualité du graphisme et des contrôles, de nombreuses variantes sont envisageables, en modifiant, si besoin est, les règles de départ : on peut autoriser l'enlèvement d'un nombre impair d'étoiles seulement, ou alterner pair et impair, ou tirer au hasard le nombre à enlever, ou encore jouer à plusieurs...

```

1  Rem LA COURSE AU 20
2  Fullw 2
3  Clearw 2
4  Dim F(8)
5  For I=1 To 4
6      Deftext 3
7      Print At(1,2*I);I
8      F(I)=0
9      Deftext 2
10     Print At(5,2*I);String$(20,237)
11 Next I
12 Deftext 1
13 T=0
14 B=1
15 Debut:
16 Print At(1,12);"JOUER " ;
17 Deftext B+1
18 Print B
19 Print At(1,14);SpC(30);At(1,16);SpC(30)
20 Deftext 1
21 Print At(1,14);"LIGNE :";
22 Input L
23 If L<>Int(L) Or L<1 Or L>4 Then
24     Goto Debut
25 Endif
26 Print At(1,16);"CASES :";
27 Input C
28 If C+F(L)>20 Or C<0 Or C<>Int(C) Then
29     Goto Debut
30 Endif
31 F(L)=F(L)+C
32 T=T+C
33 For J=F(L)-C To F(L)
34     Print At(25-J,2*L);" "
35     Sound 1,10,4,4,5
36     Sound 1,0
37 Next J
38 B=Abs(B-3)
39 If T<80 Then
40     Goto Debut
41 Endif
42 Print At(1,10);"LE JOUEUR ";B;" GAGNE"
43 For I=1 To 5 Step 2
44     Sound 1,10,I,4,10
45 Next I
46 Sound 1,0

```


Le jeu de Marienbad

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : logique (2 joueurs).

Difficulté du jeu : moyenne.

LE JEU :

Un jeu rendu très célèbre grâce à un film qui le popularisa, présenté toutefois ici dans une version plus conforme à ses origines, à savoir le jeu de NIM.

En effet, il est possible, en début de partie, de choisir le nombre d'allumettes que l'on veut mettre dans chaque tas.

Chaque joueur joue à son tour, et celui qui retire la dernière allumette a perdu.

LE PROGRAMME :

La ligne 8 dimensionne une variable A (I) qui contiendra le nombre d'allumettes de chaque tas.

La boucle I (9-14) permet d'entrer ce nombre.

La boucle I (18-23) imprime les tas.

La ligne 26 assure l'affichage du joueur.

La ligne 38, ainsi que la ligne 39, contrôlent si le coup est autorisé.

La ligne 42 contrôle la fin de la partie.

POUR JOUER :

On lance le programme avec RUN, puis on entre le nombre de tas (entre 1 et 8), suivi de RET.

On entrera ensuite le nombre d'allumettes dans chaque tas (faire à chaque entrée N suivi de RET). On obtiendra alors l'affichage des différents tas, avec le numéro du joueur concerné (1 ou 2), ainsi que le message DANS QUEL TAS ; on répond en tapant le numéro du tas dans lequel on désire faire un retrait suivi de RET. On obtient ensuite la question QUANTITE A ENLEVER, et l'on répond en indiquant le nombre d'allumettes que l'on désire enlever du tas concerné.

Puis ce sera au tour du deuxième joueur, et l'on jouera alternativement jusqu'à ce qu'il n'y ait plus d'allumettes disponibles.

En fin de partie, on obtiendra l'affichage du numéro du joueur gagnant.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut désigner chaque joueur par son nom, faire un graphisme plus élaboré, écrire un programme permettant au ST de servir de partenaire, etc.

```

1  Rem LE JEU DE MARIENBAD
2  Repeat
3      Fullw 2
4      Clearw 2
5      Input "NOMBRE DE TAS ( <= 7 ) :";N
6  Until N<8
7  S=0
8  Dim A(N)
9  For I=1 To N
10     Print
11     Print "TAS ";I;
12     Input " - NOMBRE D'ALLUMETTES :";A(I)
13     S=S+A(I)
14 Next I
15 H=1
16 Repeat
17     Clearw 2
18     For I=1 To N
19         Deftext I
20         Print At(4*I+3,4);A(I)
21         Deftext 1
22         Print At(1,6);"TAS ";At(4*I+3,6);Chr$(48+I)
23     Next I
24     Saisie:
25     Repeat
26         Print At(1,10);"JOUEUR ";H
27         Sound 1,10,5*H,4,10
28         Sound 1,0
29         Print
30         Input "DANS QUEL TAS :";T
31     Until T<=N And T=Int(T)
32     If A(T)=0 Then
33         Goto Saisie
34     Endif
35     Repeat
36         Print At(1,14)
37         Input "QUANTITE A ENLEVER :";O
38     Until O<=A(T) And O<>0
39     A(T)=A(T)-O
40     S=S-O
41     H=Abs(H-3)
42 Until S=0
43 Print
44 Print "LE JOUEUR ";H;" A GAGNE"
45 For I=1 To 5 Step 2
46     Sound 1,10,I,4,10
47 Next I
48 Sound 1,0

```

Dé truqué

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : lancer de dés.

Difficulté du jeu : facile.

LE JEU :

On veut essayer de fabriquer un dé truqué selon sa propre convenance. La méthode proposée ici consiste à définir au départ une probabilité de réalisation pour chaque type de tirage (soit la probabilité d'obtenir un 1, un 2, un 3, etc. que chacune de ces probabilités soit comprise entre 0 et 1, et que la somme de toutes les probabilités soit un nombre très voisin de 1).

LE PROGRAMME :

Il utilise le générateur aléatoire du ST supposé de qualité convenable (on pourra aisément utiliser tout autre générateur, si besoin est). La boucle I (9-22) s'occupe de l'entrée de chacune des probabilités, avec vérification à chaque entrée et sur le total (ligne 14) ; chaque probabilité est conservée dans une variable indicée E (I) : ligne 13 du programme.

Le lancer de dé truqué est effectué à l'aide d'une autre boucle (28-30), la ligne permettant le trucage se trouvant en 30. Un nouveau tirage pourra être effectué sur pression d'une touche (ligne 40).

POUR JOUER :

Après le lancement de l'exécution par RUN, on obtient l'affichage de la question **PROBABILITE DU DE 1**, à laquelle on devra répondre par un nombre compris entre 0 et 1. Il conviendra de ne pas oublier que le total de toutes les probabilités entrées ne devra en aucun cas dépasser 1.1 (tolérance de 0.1).

Des messages d'erreurs pourront être affichés et le programme sera éventuellement relancé. Lorsque toutes les probabilités auront été définies, le premier tirage sera automatiquement effectué, et il suffira de presser une touche quelconque pour effectuer les tirages suivants. On arrête le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera évidemment bien tentant d'utiliser sournoisement ce petit programme pour améliorer vos chances de gain dans un jeu de hasard utilisant des lancers de dés. On pourra alors envisager de figer les données définissant le dé truqué dans une variable A\$, etc. On pourra aussi envisager de présenter les résultats correspondant à une série de lancers de dés sous une forme spectaculaire, qui mette le trucage en évidence...

```

1  Rem DE TRUQUE
2  Clearw 2
3  Fullw 2
4  Dim A(6),B(6),C(6),E(6)
5  For I=1 To 6
6      Read A(I),B(I),C(I)
7  Next I
8  P=0
9  For I=1 To 6
10     Debut:
11     Deftext 1
12     Print At(1,3);Spc(38);At(1,3);"PROBABILITE DE SO
RTIR UN ";I;
13     Input " ";E(I)
14     If E(I)<0 Or E(I)>1 Or P+E(I)>1.1 Then
15         Gosub Faux
16         Goto Debut
17     Endif
18     P=P+E(I)
19     Print At(6*I-5,6);"DE ";I
20     Deftext 2
21     Print At(6*I-5,8);E(I)
22 Next I
23 If P<0.9 Then
24     Gosub Faux
25     Run
26 Endif
27 Do
28     Repeat
29         N=Random(6)+1
30     Until Rnd<E(N)
31     Deftext 7
32     Print At(9,13);"RESULTAT DU TIRAGE :";N;"
"
33     For I=4 To 6
34         Sound 1,10,A(I),B(I),C(I)
35     Next I
36     Sound 1,0
37     Deftext 1
38     Print
39     Print "APPUYEZ UNE TOUCHE POUR RETIRER";
40     R$=Input$(1)
41     Print At(9,13);Spc(30)
42 Loop
43 Procedure Faux
44     Print At(1,4);"IMPOSSIBLE : 0<= P <=1"
45     For J=1 To 3
46         Sound 1,10,A(J),B(J),C(J)
47     Next J
48     Sound 1,0
49     Pause 7
50     Print At(1,4);Spc(30)
51 Return
52 Data 1,4,10,3,4,10,5,4,10,1,5,8,5,5,8,8,5,8

```

Espion

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de réflexion.

Difficulté du jeu : facile.

LE JEU :

Le programme peut servir simultanément de codeur et de décodeur de message secret, suivant une méthode très facile à mettre en œuvre sur ce type d'appareil.

On utilise ici un générateur aléatoire "maison" pour modifier de façon arbitraire chaque lettre du message ; on devra bien veiller à utiliser toujours le même nombre (noyau) pour commencer le codage ou le décodage de deux messages identiques.

LE PROGRAMME :

Le noyau initial (qui sera un entier quelconque) est entré dans une boucle, de la ligne 6 à la ligne 11. Le message est entré à la ligne 15.

La ligne 19 permet de choisir entre les options de codage et de décodage.

La boucle I (23-30) est utilisée soit pour effectuer le codage, soit pour effectuer le décodage.

Les lignes 25 et 26 contiennent le générateur aléatoire.

La ligne 27 réalise la transposition des lettres.

La ligne 28 vérifie que l'on reste bien dans les limites autorisées.

Les résultats seront imprimés par la ligne 33.

POUR JOUER :

Lancer le programme par RUN.

Entrer ensuite un entier N qui servira de noyau pour le générateur (il faudra le retenir pour le décodage).

Puis taper le message, suivi de RET.

Choisir l'option C pour CODER, et l'option D pour DECODER, et taper RET.

Le message codé ou décodé, selon le cas, sera alors affiché.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra étendre le codage à d'autres signes que les lettres de l'alphabet, utiliser éventuellement plusieurs générateurs, introduire dans le message un codage du ou des nombres ayant servi de générateurs, etc.

```

1  Rem ESPION
2  Fullw 2
3  Clearw 2
4  Print "DONNEZ VOTRE CODE SECRET :";
5  R$=""
6  Do
7      S$=S$+R$
8      R$=Input$(1)
9      Exit If R$=Chr$(13)
10     Print "X";
11 Loop
12 Print
13 Print
14 Deftext 3
15 Input "QUEL EST VOTRE MESSAGE :";A$
16 A$=Upper$(A$)
17 Print
18 Deftext 1
19 Input "VOULEZ VOUS (C)ODER OU (D)ECODER :";R$
20 R$=Upper$(R$)
21 B$=""
22 A=1/Val(S$)
23 For I=1 To Len(A$)
24     T=Asc(Mid$(A$,I,1))
25     A=(A+Pi)^8
26     A=Int(26*Frac(A)+1)
27     T=T-A*((R$="C")-(R$="D"))
28     T=T+26*(T>90)-26*(T<65)
29     B$=B$+Chr$(T)
30 Next I
31 Print
32 Deftext 2
33 Print "RESULTAT : ";B$
34 Sound 1,10,4,4,10
35 Sound 1,0
36 R$=Input$(1)

```

Jour de la semaine

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : utilitaire.

Difficulté du jeu : facile.

LE JEU :

Le programme vous permet de retrouver, sans aucun calcul, le jour de la semaine correspondant à une date donnée.

La date sera entrée sous forme de trois nombres, par exemple 11, 3, 1956 pour le 11 mars 1956, et le résultat sera affiché en clair.

LE PROGRAMME :

Les lignes 7 à 12 contrôlent les entrées. La ligne 8 s'occupe de la fin de partie. La ligne 15 permet une correction pour les dates antérieures à 1582 (mise en service du calendrier actuel).

Les lignes 18 à 20 permettront l'affichage en clair du mois dans la réponse (sur 9 lettres).

La formule utilisée apparaît aux lignes 31 et 32.

Les lignes 33 et 34 permettront l'affichage en clair du jour dans la réponse.

Les lignes 35 à 45 impriment les résultats.

POUR JOUER :

Lancer le programme par RUN, puis entrer le jour suivi de RET, le mois (entre 1 et 12) suivi de RET, et l'année (sous la forme 1845, par exemple) suivie de RET.

Le résultat est très rapidement affiché.

On tape une touche quelconque et le programme revient sur la question JOUR.

Pour continuer, il suffit de retaper un jour, un mois, une année.

Pour terminer, il faut répondre Ø à la question JOUR.

EXTENSIONS POSSIBLES :

On pourra entrer les mois en clair et faire afficher le nom du jour en entier.

On peut envisager de faire afficher les dates et fêtes particulières (Noël, Pâques, les Rameaux, Pentecôte...).

On peut écrire un programme qui afficherait les jours fériés pour une année donnée, etc.

```

1  Rem JOUR DE LA SEMAINE
2  Do
3      Fullw 2
4      Clearw 2
5      Setcolor 0,800
6      Deftext 3
7      Input "JOUR.....":J
8      Exit If J=0
9      Print
10     Input "MOIS.....":M
11     Print
12     Input "ANNEE.....":A
13     B=A
14     S=Int(A/100)-Int(A/400)
15     If A<=1582 Then
16         S=2
17     Endif
18     M$="JANVIER FEVRIER MARS     AVRIL     MAI
        JUIN     JUILLET  "
19     M$=M$+"AOUT     SEPTEMBREOCTOBRE  NOVEMBRE DECEM
        BRE  "
20     M$=Mid$(M$,9*M-8,9)
21     R$=""
22     For I=1 To Len(M$)
23         If Mid$(M$,I,1)<>" " Then
24             R$=R$+Mid$(M$,I,1)
25         Endif
26     Next I
27     If M<3 Then
28         M=M+12
29         A=A-1
30     Endif
31     S=J-1+Int(13*(M+1)/5)+Int(5*A/4)-S
32     S=S-7*Int(S/7)
33     J$="DIMANCHELUNDI  MARDI  MERCREDIJEUDI  VEND
        REDISAMEDI  "
34     J$=Mid$(J$,8*S+1,8)
35     Print At(1,13);"LE ";
36     Deftext 2
37     Print J;
38     Deftext 4
39     Print " ";R$;" ";
40     Deftext 5
41     Print B;
42     Deftext 1
43     Print " ETAIT UN ";
44     Deftext 8
45     Print J$
46     For I=1 To 5 Step 2
47         Sound 1,10,I,4,10
48     Next I
49     Sound 1,0
50     Deftext 1
51     Print
52     Print "APPUYEZ SUR UNE TOUCHE";
53     R$=Input$(1)
54 Loop
55 Setcolor 0,1911

```


REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : lancer de dés.

Difficulté du jeu : moyenne.

LE JEU :

Chacun des joueurs devra, à son tour, lancer 3 dés, et peut ensuite choisir, sur les deux coups suivants, s'il désire relancer ou non 1, 2 ou 3 dés. Il marque alors un score correspondant au nombre indiqué par les dés, suivant les règles choisies, et le joueur suivant procède de la même manière.

LE PROGRAMME :

Le nombre de joueurs est tout d'abord entré en ligne 5.

La ligne 18 affiche le numéro de joueur et la ligne 20 affiche le nombre d'essais déjà effectués.

La ligne 48, utilisée 3 fois dans la boucle J (47-54), réalise le lancer de dés.

Les retirages sont effectués par les lignes 37, 40, et 43.

La variable R\$ gère les 2^e et 3^e tirages. L'impression du résultat est réalisée dans tous les cas par les lignes 22-24.

Les tests pour le nombre d'essais et pour le changement de joueur sont effectués aux lignes 32 à 36.

POUR JOUER :

On lance l'exécution du programme par RUN ce qui conduit à l'affichage NOMBRE DE JOUEURS. On entre le nombre de joueurs, et l'on tape RET. L'écran s'efface et affiche JOUEUR N° 1. ESSAI A, PARTIE P, ainsi que le tirage réalisé, et les noms des dés (A, B, C).

Si l'on ne désire pas refaire de tirage, il suffit de frapper RET. La main passe alors au joueur suivant.

Pour effectuer un autre tirage, il suffit de taper le nom du ou des dés à retirer, et de faire RET : par exemple, pour retirer A et C, on tapera AC suivi de RET.

Les joueurs suivants procéderont de la même façon.

On devra taper la combinaison CSA pour terminer la partie.

EXTENSIONS POSSIBLES :

La plus évidente consiste à assurer la gestion des résultats des joueurs, en se référant aux règles de gain usuelles du jeu du 421. On pourra envisager l'affichage après chaque tour d'un tableau récapitulant les résultats des joueurs.

On pourra aussi améliorer l'affichage des dés (se référer aux divers programmes d'affichage des dés se trouvant dans le même ouvrage).

```

1  Rem 421
2  Fullw 2
3  Clearw 2
4  P=1
5  Input "NOMBRE DE JOUEURS :";N
6  Do
7      I=1
8      Repeat
9          K=0
10         Clearw 2
11         R$="ABC"
12         Deftext 2
13         Print At(14,1);"-421-"
14         Gosub Tirage
15         Repeat
16             K=K+1
17             Deftext 1
18             Print At(1,4);"JOUEUR NO. ";I;
19             Deftext 1
20             Print "- ESSAI ";K;" - PARTIE ";P
21             Deftext 3
22             Print At(7,7);A;At(17,7);B;At(27,7);C
23             Deftext 1
24             Print At(7,9);"A";At(17,9);"B";At(27,9);"C"
25             R$=""
26             Print At(1,14);Spc(35);At(1,14);"ABC :";
27             Input R$
28             R$=Upper$(R$)
29             Exit If Len(R$)=0
30             R$=R$+"DDD"
31             Gosub Tirage
32         Until K=3
33         I=I+1
34     Until I>N
35     P=P+1
36 Loop
37 Procedure Premier
38     A=D
39 Return
40 Procedure Deuxieme
41     B=D
42 Return
43 Procedure Troisieme
44     C=D
45 Return
46 Procedure Tirage
47     For J=1 To 3
48         D=Random(6)+1
49         Sound 1,10,2*D-1,4,10
50         Sound 1,0
51         X$=Mid$(R$,J,1)
52         X=Asc(X$)-64
53         On X Gosub Premier,Deuxieme,Troisieme
54     Next J
55 Return

```

Compose

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : composition musicale.

Difficulté du jeu : aucune, a priori.

LE JEU :

Il s'agit, en fait, d'un programme capable de composer, de façon très élémentaire, de la musique, et cela d'une façon complètement aléatoire. L'ordinateur va, dans un premier temps, dessiner une portée, puis il affiche une succession de notes (allant du DO grave au SI aigu) et donne leur nom. Une nouvelle série de notes sera ensuite affichée, et ainsi de suite.

LE PROGRAMME :

La boucle I (10-12) dessine la portée. La variable N\$, en ligne 13, contient les noms des notes et P\$ contient les notes sous forme codée.

La deuxième boucle I (15-25) permet de faire le choix d'une note de façon aléatoire, à la ligne 16 ; son nom est calculé en ligne 17 et la note est dessinée par la ligne 19.

La ligne 27 permet de recommencer en tapant une touche quelconque (sauf S).

POUR JOUER :

Lancer l'exécution du programme en tapant RUN.

La portée sera alors rapidement affichée, puis les notes sont représentées, avec leur nom affiché en bas de l'écran.

Une pression sur une touche quelconque provoquera l'affichage d'une nouvelle série de notes, et une pression sur la touche "S" provoque la fin de la partie.

EXTENSIONS POSSIBLES :

On pourra affiner les règles de composition, afficher davantage de notes, introduire des dièses et des bémols, améliorer le graphisme (qui fait appel à l'un des caractères prédéfinis de l'ATARI ST).

```

1  Rem COMPOSE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  K=-(R>1)
6  Setcolor 0,1911+800*(R=1)
7  Repeat
8    Clearw 2
9    Deftext 3,1
10   For I=1 To 5
11     Print At(1,2*I+3);String$(39+25*K,45)
12   Next I
13   N$="SI LA SOLFA MI RE DO "
14   P$="12100806050301"
15   For I=1 To 11+11*K
16     X=Random(14)+1
17     D$=Mid$(N$,3*(X+7*(X>7))-2,3)
18     Deftext 2,1
19     Print At(3*I-1,X);Chr$(124);At(3*I-2,X+1);Chr$
      (79)
20     Deftext 4,0
21     Print At(3*I-2,17);D$;
22     Sound 1,10,Val(Mid$(P$,2*(X+7*(X>7))-1,2)),4-(
      X<=7),15
23     Sound 1,0
24     Pause 3
25   Next I
26   Deftext 1
27   R$=Upper$(Input$(1))
28   Until R$="S"
29   Setcolor 0,1911

```

Tour de cartes

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de prestidigitation.

Difficulté du jeu : facile.

LE JEU :

C'est un tour assez facile mais, ici, les cartes sont manipulées sous vos yeux par le ST qui tient le rôle du prestidigitateur ; 21 cartes, prises au hasard dans un paquet de 52, sont affichées, et le joueur devra retenir mentalement l'une des cartes présentées. Puis, il devra indiquer au ST dans quelle colonne (1^{re}, 2^e ou 3^e à partir de la gauche) se trouve la carte choisie ; les cartes sont alors redistribuées et le joueur indique de nouveau la colonne correspondant à sa carte ; cela est fait encore une fois et, au 4^e affichage, le ST est en mesure de deviner la carte choisie.

LE PROGRAMME :

La boucle I (7-13) permet de choisir, au hasard, 21 cartes sur 52, et la boucle emboîtée (8-10) permet de s'assurer que l'on ne réalise pas deux fois le même tirage. La ligne 17 contient les couleurs des cartes ainsi que les noms des cartes figures (VA, DA, RO, AS), dans une variable alpha-numérique B\$.

La boucle I (18-36) réalise l'affichage du jeu.

La ligne 19 décode la couleur de la carte.

La ligne 20 choisit la position d'affichage.

La ligne 22 décode la hauteur.

L'affichage proprement dit est effectué de la ligne 27 à la ligne 35.

La ligne 41 permet au joueur d'entrer le numéro de la colonne où se trouve sa carte. La boucle J (46-48) réalise le brouillage, et les boucles emboîtées J (51-55) et I (52-54) fabriquent la nouvelle variable A\$ à utiliser pour le prochain affichage.

POUR JOUER :

On lance le programme par RUN.

On obtient un premier affichage de 21 cartes réparties sur 3 colonnes.

On indique alors le numéro (1, 2 ou 3) de la colonne où se trouve la carte choisie, suivi de RET.

On obtient un 2^e affichage et l'on fait de même ; puis un 3^e affichage et l'on fait de même.

Au 4^e affichage, la carte choisie est automatiquement affichée.

On sortira du programme en tapant la combinaison CSA à un moment quelconque.

EXTENSIONS POSSIBLES :

On peut envisager une amélioration de l'affichage en écrivant les noms des cartes en entier, ou même, en dessinant de véritables cartes, en couleur.

On pourra aussi essayer de programmer d'autres tours, utilisant le même principe de base, mais avec davantage de cartes, etc.

```

1  Rem TOUR DE CARTES
2  Fullw 2
3  Clearw 2
4  Setcolor 0,220
5  Dim T(100)
6  A$=""
7  For I=1 To 21
8      Repeat
9          F=Random(52)+38
10         Until T(F)<>1
11         T(F)=1
12         A$=A$+Chr$(F)
13     Next I
14     H=3
15     P=1
16     Do
17         B$="TRCACOPIVADARQAS"
18         For I=1 To Len(A$)*P-11*(
P=0)
19             S=Int((Asc(Mid$(A$,I,1)
)-38)/13)
20             J=7*Int((I-1)/7)
21             Deftext 1
22             T=Asc(Mid$(A$,I,1))-37-
13*S
23             If P=0 Then
24                 Setcolor 0,1911
25                 Print At(1,1);"VOTRE
CARTE EST :";Spc(40)
26             Endif
27             Print At(J*P+Int(J/2)-1
6*(P=0)+1,(I-J)*P+1);
28             If T<=9 Then
29                 Print Str$(T+1);
30             Else
31                 Print Mid$(B$,Abs(2*T
-11),2);
32         Endif
33         Print At((J+3)*P+1+Int(
J/2)-21*(P=0)+1,(I-J)*P+1);
34         Deftext S+1
35         Print Mid$(B$,2*S+1,2)
36     Next I
37     Exit If P=0
38     Print At(1,15);
39     Sound 1,10,4,4,15
40     Sound 1,0
41     Input "QUELLE COLONNE :";
N
42     Clearw 2
43     H=H-1
44     R$="20120"
45     B$=""
46     For J=0 To 2
47         B$=B$+Mid$(A$,7*Val(Mid
$(R$,N+J,1))+1,7)
48     Next J
49     If H<>0 Then
50         A$=""
51         For J=0 To 2
52             For I=1 To 7
53                 A$=A$+Mid$(B$,3*I-2
+J,1)
54             Next I
55         Next J
56     Else
57         A$=B$
58         P=0
59     Endif
60     Loop
61     For I=1 To 5 Step 2
62         Sound 1,10,1,5,20
63     Next I
64     Sound 1,0

```

Pendu inversé

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu sur les mots.

Difficulté du jeu : facile.

LE JEU :

Le Pendu est un jeu de lettres très connu, présenté par ailleurs dans ce recueil, dans lequel il s'agit de deviner un mot caché par l'ordinateur et où chaque lettre non découverte vous rapproche un peu plus de la potence. Ici, c'est le ST qui va chercher à deviner votre mot, mais il joue quelque peu au hasard et ne possède pas votre intuition ; il faudra donc, en règle générale, beaucoup d'essais pour découvrir la solution. Pour augmenter ses chances, on a fait appel à une formule qui lui permet de tirer davantage dans les voyelles puis, dans les consonnes courantes, et, enfin, dans les consonnes plus rares.

LE PROGRAMME :

La ligne 20 permet d'enregistrer, dans une variable alphanumérique D\$, les lettres de l'alphabet, dans l'ordre : voyelles, consonnes usuelles, autres consonnes. Le mot à deviner est introduit lettre à lettre par une boucle (7-16).

La ligne 18 imprime des tirets aux emplacements des lettres à trouver.

La boucle J (24-29) sert uniquement à simuler le temps de recherche d'une lettre.

La formule, placée en ligne 31, permet de modifier légèrement la probabilité de tirage, en faveur des voyelles et des consonnes usuelles.

La ligne 33 provoque un nouveau tirage, et la boucle I (42-51) réimprime les tirets et les lettres trouvées.

La ligne 52 incrémente le compteur d'essais et affiche les lettres non prises.

La ligne 60 affiche le nombre d'essais réalisés, en fin de partie.

POUR JOUER :

Lancer le programme par RUN.

Entrer le mot à découvrir, lettre par lettre. On pourra effacer en tapant Ø, en cas d'erreur. On termine en tapant RET.

Des tirets sont alors affichés (un par lettre) et le ST propose une première lettre, regarde si elle convient et continue automatiquement, jusqu'à ce qu'il ait trouvé.

On pourra sortir du programme à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera sans doute intéressant, pour de jeunes enfants, de prévoir une version non automatique où le joueur répondrait OUI ou NON pour chaque lettre proposée par le ST ; on pourra aussi imaginer d'ajouter un affichage graphique, par exemple une potence ou une voiture qui avance vers un précipice, etc.

```

1  Rem LE PENDU INVERSE
2  Fullw 2
3  Clearw 2
4  R$=""
5  T$=""
6  U=1
7  Do
8      Print At(1,1);"MOT :";T$;
9      S$=Upper$(Input$(1))
10     Exit If S$=Chr$(13)
11     R$=R$+S$
12     T$=T$+"#"
13     If S$="0" Then
14         Run
15     Endif
16 Loop
17 P$=String$(Len(R$)," ")
18 Print At(1,5);P$
19 C=1
20 D$="YAEIOUBCDFGLMNPSTVWHKJ
   QXZ"
21 Repeat
22     Deftext 4
23     Print At(1,9);"JE CHERCHE
   ...; ESSAI ";C
24     For J=1 To Random(800)+50
25         If Rnd<0.01 Then
26             Sound 1,10,Random(12)
27             +1,Random(3)+3,2
28             Sound 1,0
29         Endif
30     Next J
31     Repeat
32         X=-6*(Rnd<0.7)-13*(Rnd<
33         0.5)-7*(Rnd<0.4)
34         X=Random(X)+1
35         Until Mid$(D$,X,1)<>"-"
36         E$=Mid$(D$,X,1)
37         Mid$(D$,X,1)="-"
38         K=0
39         Print At(1,11);"JE PRENDS
   ";
40         Deftext 2
41         Print E$
42         Sound 1,10,4,5,8
43         Sound 1,0
44         For I=1 To Len(R$)
45             If E$=Mid$(R$,I,1) Then
46                 K=1
47                 Mid$(P$,I,1)=E$
48                 Sound 1,10,6,6,5
49                 Endif
50                 Deftext 7
51                 Print At(I,5);Mid$(P$,I
   ,1)
52             Next I
53             C=C+1
54             If K=0 Then
55                 Deftext 3
56                 Print At(U,13);E$
57                 Inc U
58             Endif
59             Until P$=R$
60             Deftext 1
61             Print At(1,16);"TROUVE EN "
62             ;C-1;" COUPS"
63             For I=1 To 5 Step 2
64                 Sound 1,10,I,5,8
65             Next I
66             Sound 1,0

```


Musique

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu éducatif.

Difficulté du jeu : moyenne.

LE JEU :

Il se présente comme un exercice d'apprentissage du solfège : l'appareil annonce une note (en clé de sol), suivie de la mention GRAVE ou AIGUE, selon sa place sur la portée. Il faudra répondre en indiquant sur quelle ligne la note sera affichée.

La note est alors dessinée sur la portée, suivie d'un message BRAVO ou PERDU, selon que l'on avait trouvé juste ou non.

LE PROGRAMME :

La boucle I (7-10) dessine la portée. Une deuxième boucle I (12-14) affiche les numéros des lignes.

La ligne 15 définit le nom des notes, et la ligne 17 réalise le tirage aléatoire d'une note.

Le nom de la note est calculé à la ligne 18 et il est affiché par la ligne 19, suivi de la mention GRAVE ou AIGUE, selon le cas.

La réponse sera entrée en ligne 25.

La ligne 27 s'occupe du décodage de la réponse.

Les lignes 29 à 31 dessinent la note.

Les lignes 33 à 44 regardent si l'on a gagné ou perdu et affichent le message correspondant.

La ligne 48 permet de faire un nouvel essai.

POUR JOUER :

On lance le programme par RUN.

On obtient alors l'affichage de la portée, ainsi que le nom de la note à placer et la question NUMERO DE LA LIGNE.

On répond en donnant un numéro de ligne (de 1 à E) suivi de RET.

La note est alors dessinée et le message BRAVO ou PERDU apparaît, dans ce cas, à l'écran. La note est ensuite jouée.

On doit taper une touche quelconque pour continuer, et s'arrêter en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut étendre à une autre clé que la clé de sol, on peut introduire de la couleur, des sons, etc.

```

1  Rem MUSIQUE
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
5  Do
6      Clearw 2
7      For I=1 To 5
8          Deftext 4,1
9          Print At(3,2*I+3);String$(22*R,95)
10         Next I
11         Deftext 1,0
12         For I=1 To 14
13             Print At(1,I+1);Chr$(48+I-7*(I>9))
14         Next I
15         N$="SI LA SOLFA MI RE DO "
16         P$="12100806050301"
17         X=Random(14)+1
18         T=X-1-7*Int((X-1)/7)
19         Print At(1,18);"NOTE  ";Mid$(N$,3*T+1,3);"  ";
20         If X<=7 Then
21             Print "AIGUE"
22         Else
23             Print "GRAVE"
24         Endif
25         Input "NUMERO DE LA LIGNE :";L$
26         L$=Upper$(L$)
27         L=- (Asc(L$)-48)*(L$<"")-(Asc(L$)-55)*(L$>="A")
28         Deftext 2,1
29         Print At(19,Abs(X-2)+1);Chr$(124);At(19,X);Chr$(
30         124)
31         Print At(17,X+1);Chr$(47);Chr$(92);At(19,X+1);Ch
32         r$(124)
33         Print At(17,X+2);Chr$(92);Chr$(47);At(23,16);
34         Deftext 1,0
35         If L<>X Then
36             Print "PERDU"
37             Sound 1,10,2,2,30
38         Else
39             Print "BRAVO"
40             N=Val(Mid$(P$,2*T+1,2))
41             If X>7 Then
42                 Sound 1,10,N,4,20
43             Else
44                 Sound 1,10,N,5,20
45             Endif
46         Endif
47         Sound 1,0
48         Print At(1,20);"TAPEZ SUR UNE TOUCHE  ";
49         R$=Input$(1)
50     Loop

```

Lettres

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : moyenne.

LE JEU :

La règle est fortement inspirée d'un jeu de lettres très connu en France, pratiqué notamment dans une émission télévisée (*Les chiffres et les lettres*). Le ST choisit au hasard un certain nombre de symboles (voyelles et consonnes) dans l'alphabet latin, et chacun des joueurs devra indiquer la longueur du mot qu'il a réussi à obtenir (l'appareil ne vérifiera pas si le mot est correct !).

Le score total de chaque joueur est affiché ensuite.

LE PROGRAMME :

La boucle I (12-22) choisit dans l'alphabet entre 7 et 12 lettres, en fonction de la variable aléatoire L, tirée en ligne 7. Pour éviter d'avoir trop de consonnes, on utilise en ligne 16 une petite astuce, qui consiste à refaire au hasard un tirage dans la table A\$, contenant des voyelles (on peut en avoir plus ou moins en utilisant d'autres RND).

La boucle 28-47 demande à chacun des joueurs le mot proposé.

La boucle J (34-38) contrôle la qualité des réponses.

La ligne 39 mesure les longueurs.

Les lignes 48 et 49 comparent les résultats et totalisent le score de chaque joueur, affiché par la ligne 50.

POUR JOUER :

On lance l'exécution du jeu avec RUN.

Les lettres sont alors affichées, puis la question JOUEUR 1 LONGUEUR à laquelle le premier joueur répond en donnant le mot qu'il a obtenu (l'autre joueur devra toutefois vérifier si le mot est correct) ; puis le joueur 2 fera de même et le score de chaque joueur sera alors affiché.

Pour continuer la partie, il suffira d'appuyer sur une touche quelconque.

On devra taper Ø pour finir.

EXTENSIONS POSSIBLES :

On pourra bien entendu songer à introduire une temporisation avant de demander leurs réponses aux joueurs.

On pourra aussi prévoir pour chaque mot de répondre à la question CORRECT OU NON ? On peut aussi étendre le jeu à plus de deux joueurs. Et pourquoi ne pas essayer de définir un jeu contre le ST, auquel on aurait auparavant entré une liste de mots assez longue, dont il afficherait les lettres dans le désordre, avec éventuellement quelques lettres de complément choisies au hasard ?

```

1  Rem LETTRES
2  Fullw 2
3  J1=0
4  J2=0
5  A$="AEIOUEA"
6  Repeat
7    L=Random(6)+7
8    Clearw 2
9    F$=""
10   Deftext 1
11   Print "LETTRES TIREES"
12   For I=1 To L
13     Deftext Random(12)+1,1
14     R$=Chr$(Random(26)+65)
15     If Rnd<0.4 Then
16       R$=Mid$(A$,Random(Len(A$))+1,1)
17     Endif
18     Sound 1,10,Random(12)+1,4,5
19     Sound 1,0
20     Print At(2*I+1,6);R$
21     F$=F$+R$
22   Next I
23   Deftext 1,0
24   Print At(11,8);"ATTENTION"
25   Pause 200
26   I=Random(2)+1
27   K=1
28   Repeat
29     Print At(1,6);Spc(30);At(1,8);Spc(38);At(1,6);
30     "JOUER ";
31     Deftext I+1,1
32     Print I;
33     Deftext 1,0
34     Input " - VOTRE-MOT : ";R$
35     For J=1 To Len(R$)
36       M$=Mid$(Upper$(R$),J,1)
37       S=Instr(1,F$,M$)
38       Exit If S=0
39     Next J
40     R$=Left$(R$,-Len(R$)*(S<>0))
41     If I=1 Then
42       P=Len(R$)
43     Else
44       L=Len(R$)
45     Endif
46     K=K+1
47     I=Abs(I-3)
48   Until K=3
49   J1=J1+(P And P>L)
50   J2=J2+(L And L>P)
51   Print At(1,12);"SCORES : JR.1 : ";J1;" - JR.2 : ";
52   J2
53   Print
54   Print "TAPER 0 POUR FINIR";
55   R$=Input$(1)
56   Until R$="0"

```

Machinamot

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de hasard (sur des mots).

Difficulté du jeu : aucune.

LE JEU :

Ce très court programme permet au ST de faire la démonstration de ses capacités à manipuler des chaînes de caractères : on part simplement d'un préfixe et d'un suffixe d'origine grecque, que l'on va assembler au hasard. On tombera, bien sûr, de temps à autre, sur des mots absurdes ou même légèrement équivoques.

Un nouveau mot est confectionné à chaque essai, mais il est possible d'obtenir plusieurs fois le même mot.

LE PROGRAMME :

Deux variables A\$ et B\$ sont dimensionnées à 15 en ligne 6 : l'une contiendra les préfixes, l'autre les suffixes (si l'on désire rajouter des possibilités, il suffira de modifier cette limitation à 15).

La boucle I (7-9) lit les lignes DATA (26-30) concernant les préfixes suivis des suffixes.

La ligne 11 effectue le choix d'un préfixe au hasard.

La ligne 12 effectue le choix d'un suffixe au hasard.

Le mot obtenu en ligne 13 est affiché lettre par lettre par 2 boucles emboîtées I (14-22) et J (16-19), puis on recommence (ligne 25).

POUR JOUER :

Lancer l'exécution du programme par RUN. Une première série de lettres vient former un mot sur l'écran ; puis un nouveau mot sera affiché à la suite du précédent.

Taper la combinaison CSA pour sortir du programme.

EXTENSIONS POSSIBLES :

On pourra très facilement augmenter la liste des mots disponibles.

On pourra aussi utiliser un procédé analogue pour faire confectionner des phrases à l'ordinateur, lui faire inventer des poèmes, etc. On pourra aussi transformer le programme initial en un jeu de société qui consisterait, par exemple, à prendre des paris sur le prochain mot qui va apparaître. Et confier, éventuellement, la gestion des paris au ST...

```

1  Rem MACHINAMOT
2  Fullw 2
3  Clearw 2
4  Input "ECRAN 1:COULEUR 2:NOIR :";R
5  Clearw 2
6  Dim A$(15),B$(15)
7  For I=1 To 15
8      Read A$(I),B$(I)
9  Next I
10 Do
11     I=Random(15)+1
12     J=Random(15)+1
13     C#=A$(I)+B$(J)
14     For I=1 To Len(C#)
15         M#=Mid$(C#,Len(C#)-I+1,1)
16         For J=0 To 22-I
17             Deftext -(Random(15))*(R=1)+1,,,20
18             Print At (J+1,15);" ";M#;
19         Next J
20         Sound 1,10,Random(12)+1,4,4
21         Sound 1,0
22     Next I
23     Pause 200
24     Print
25 Loop
26 Data ANDRO,CRATE,COSMO,PHILE,MICRO,PHOBE
27 Data HELIO,TROPE,SEXO,GYNE,STEREO,MANE
28 Data PALEO,NAUTE,MACRO,PHAGE,HYDRO,FUGE
29 Data STENO,PATHE,PATHO,CARDE,OLEO,DERME
30 Data MYTHO,SCOPE,HOMEO,VORE,OCTO,PHONE

```

Tirage d'un dé truqué

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion.

Difficulté du jeu : moyenne.

LE JEU :

Il reprend une idée d'un programme précédent (DE TRUQUE) dans une application statistique : on fabrique un dé volontairement truqué, que l'on va lancer un certain nombre de fois, et l'on dresse un tableau des résultats.

LE PROGRAMME :

Les probabilités désirées pour chaque tirage sont introduites par une boucle I (7-20) avec toutes les vérifications de validité nécessaires. Le nombre de tirages demandés est introduit en ligne 26.

La boucle J (27-37) comptabilise les tirages.

La boucle I (28-30) assure la conformité de chaque tirage avec les probabilités introduites en début de jeu.

Les résultats sont comptabilisés en ligne 31, et affichés grâce aux lignes 33 et 34.

POUR JOUER :

Commencer l'exécution en frappant RUN, puis à la question PROBABILITE DU DE 1, introduire un nombre compris entre 0 et 1.

Faire de même pour les dés 2, 3, etc., sans oublier que le total des probabilités introduites devra être proche de 1.

Ensuite, à l'affichage de la question NOMBRE DE TIRAGES, on introduira un entier suffisamment grand, suivi de RET. L'affichage des résultats mettra d'autant plus de temps à apparaître que l'entier introduit sera grand.

Les résultats sont affichés sous les numéros des tirages : ainsi 32.30 apparaissant sous le 1 signifie que le tirage de 1, auquel on avait affecté une probabilité de 0.30, a été réalisé 32 fois.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager d'améliorer la représentation des résultats en utilisant par exemple un histogramme, ou toute autre méthode graphiquement plus parlante.

```

1  Rem TIRAGE D'UN DE TRUQUE
2  Clearw 2
3  Fullw 2
4  P=0
5  Dim E(6),A(3),B(3),C(3)
6  A$="###.##"
7  For I=1 To 6
8      Debut:
9      Deftext 1,0,0,13+7*(Xbios(4)<2)
10     Print At(1,3);Spc(38);At(1,3);"PROBABILITE DE SO
RTIR UN ";I;
11     Input " : ";E(I)
12     If E(I)<0 Or E(I)>1 Or P+E(I)>1.1 Then
13         Gosub Faux
14         Goto Debut
15     Endif
16     P=P+E(I)
17     Print At(6*I-5,6);"DE ";I
18     Deftext 2
19     Print At(6*I-5,8);E(I)
20 Next I
21 If P<0.9 Then
22     Gosub Faux
23     Run
24 Endif
25 Print
26 Input "NOMBRE DE TIRAGES :";P
27 For J=1 To P
28     Repeat
29         N=Random(6)+1
30         Until Rnd<Frac(E(N))
31         E(N)=E(N)+1
32         Deftext N+1
33         Print At(6*N-2,13);N
34         Print At(6*N-5,15);Using A$,E(N)
35         Sound 1,10,2*N-1,4,2
36         Sound 1,0
37     Next J
38     Deftext 1
39     Sound 1,10,4,5,40
40     Sound 1,0
41     Print
42     Print "C'EST FINI"
43     R$=Input$(1)
44     Procedure Faux
45         Print At(1,4);"IMPOSSIBLE : 0<= P <=1"
46         For J=1 To 3
47             Sound 1,10,A(J),B(J),C(J)
48         Next J
49         Sound 1,0
50         Pause 40
51         Print At(1,4);Spc(30)
52     Return
53 Data 1,4,10,3,4,10,5,4,10,1,5,8,5,5,8,8,5,8

```


Jeu de loto

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : hasard (plusieurs joueurs).

Difficulté du jeu : facile.

LE JEU :

Il s'agit de réaliser un tirage complet de loto, c'est-à-dire d'obtenir 6 numéros, compris entre 1 et 49, plus un numéro complémentaire, tous différents, qui pourraient être, par exemple, utilisés pour remplir votre prochaine grille de loto.

LE PROGRAMME :

On utilisera un tableau N(7), dimensionné en ligne 3, pour enregistrer les résultats.

Le tirage des nombres est effectué dans une boucle I (9-15), une deuxième boucle (10-13) étant utilisée pour contrôler qu'il n'y a pas deux tirages identiques.

Le tirage (ou le retraitage) est effectué en ligne 11.

La boucle I (24-34) affiche le résultat, et la ligne 36 affiche le numéro complémentaire.

Les lignes 41 et 43 s'occupent du cas où l'on voudrait refaire une autre partie.

POUR JOUER :

Il suffit de taper RUN pour lancer le programme et obtenir l'affichage d'un premier loto.

La question VOULEZ-VOUS UN AUTRE TIRAGE (O/N) s'affiche en bas de l'écran ; on répondra O si l'on désire continuer.

EXTENSIONS POSSIBLES :

On peut imaginer un programme qui pourrait tenir compte, soit de ses propres résultats, soit de résultats entrés par un joueur pour améliorer le tirage, en tenant compte des probabilités de sortir un numéro donné.

On peut aussi envisager un programme qui réaliserait des cartons pour le jeu de loto plus usuel, tel qu'il est encore couramment pratiqué dans le Midi de la France.

```

1  Rem JEU DE LOTO
2  Fullw 2
3  Dim N$(7),N(7),T(50)
4  Repeat
5      Clearw 2
6      Setcolor 0,Random(1912)
7      Deffill 1,2,4
8      Fill 1,1
9      For I=1 To 7
10         Repeat
11             N(I)=Random(49)+1
12             N$(I)=Str$(N(I))
13             Until T(N(I))=0
14             T(N(I))=1
15         Next I
16         A$="TIRAGE DU LOTO"
17         Deftext 1,,16
18         For I=1 To Len(A$)
19             Print At(2*I,1);"      ";
20         Next I
21         For I=1 To Len(A$)
22             Print At(2*I+2,1);Mid$(A$,I,1);
23         Next I
24         For I=1 To 6
25             C=Random(8)+2
26             C=C+(C-1)*(C=6)
27             For J=1 To Len(N$(I))
28                 Deftext C,9,,20
29                 Print At(4*I-3+J,3);Mid$(N$(I),J,1);
30                 Sound 1,10,Val(Mid$(N$(I),J,1)),4,2
31                 Sound 1,0
32             Next J
33             Print " ";
34         Next I
35         Deftext 1,0,,13
36         Print At(1,9);"NUMERO COMPLEMENTAIRE : ";N(7)
37         For I=1 To 5 Step 2
38             Sound 1,10,I,5,5
39         Next I
40         Sound 1,0
41         Print At(1,11);"VOULEZ-VOUS UN AUTRE TIRAGE";
42         R$=Upper$(Input$(1))
43         Until R$<>"0"
44         Setcolor 0,1911

```

Le jeu de la vie

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu solitaire (2 versions).

Difficulté du jeu : facile.

LE JEU :

La version présentée ici respecte les règles posées à l'origine par le mathématicien J.H. CONWAY.

Une certaine population est définie au début du jeu soit par le ST (version automatique) soit par vous-même (version manuelle). Cette population va évoluer sous vos yeux, en respectant certaines règles, très élémentaires, de naissance, de vie et de mort. L'évolution se poursuivra jusqu'à ce que l'on décide d'arrêter le programme.

LE PROGRAMME :

Deux tables (B et C) sont utilisées en ligne 5, pour enregistrer l'état de la population. Les boucles F (30-44) et G (31-43) sont utilisées en début de jeu pour définir la population initiale (lignes 33 à 41). Le calcul de la nouvelle population est ensuite réalisé à l'aide d'une quadruple boucle : 10-26, 11-25, 13-19, 14-18. La ligne 16 réalise la gestion du nouvel état suivant les règles.

L'affichage de ce nouvel état est obtenu en réutilisant la double boucle d'affichage F (30-44) et G (31-43).

La ligne 48 permet de recommencer.

POUR JOUER :

Dans la version automatique, il suffit de lancer le programme par RUN et de regarder. Pour arrêter, il faudra taper la combinaison CSA. Dans la version manuelle, la configuration de départ sera entrée, après le lancement du programme par RUN, en tapant 0 si l'on ne veut rien mettre sur la ligne et la colonne indiquée, et 1 si l'on désire y mettre une cellule. Puis l'évolution se déroulera ensuite de façon automatique.

EXTENSIONS POSSIBLES :

Il sera facile, si l'on dispose d'une mémoire plus importante, d'augmenter les dimensions des tables B et C, en redéfinissant la variable A. On pourra aussi introduire la couleur, améliorer la précision du dessin en remplaçant les PRINT par un affichage haute résolution.

```

1  Rem LE JEU DE LA VIE ( AUTOMATIQUE )
2  Fullw 2
3  Clearw 2
4  Input "DIMENSION :";A
5  Dim B(A,A),C(A,A)
6  Deftext 2
7  I=0
8  Goto Affiche
9  Calcul:
10 For F=1 To A
11   For G=1 To A
12     S=0
13     For I=-1 To 1
14       For J=-1 To 1
15         If F+J<>0 And G+I<>0 And F+J<>A+1 And G+I<
>A+1 Then
16           S=S+B(F+J,G+I)
17         Endif
18       Next J
19     Next I
20     S=2*S-B(F,G)
21     C(F,G)=0
22     If S>=5 And S<=7 Then
23       C(F,G)=1
24     Endif
25   Next G
26 Next F
27 Affiche:
28 Clearw 2
29 K=0
30 For F=1 To A
31   For G=1 To A
32     If I=0 Then
33       If Rnd<0.5 Then
34         C(F,G)=1
35       Endif
36     Endif
37     Print At(2*G,2*F);
38     If C(F,G)=1 Then
39       K=1
40       Print "*"
41     Endif
42     B(F,G)=C(F,G)
43   Next G
44 Next F
45 Sound 1,10,5,6,5
46 Sound 1,0
47 If K=1 Then
48   Goto Calcul
49 Endif
50 Color 1
51 For I=1 To 5 Step 2
52   Sound 1,10,1,5,5
53 Next I
54 Sound 1,0
55 Print At(1,17);"C'EST FINI..."
56 R$=Input$(1)

```

```

1  Rem LE JEU DE LA VIE ( MANUEL )
2  Fullw 2
3  Clearw 2
4  Input "DIMENSION :";A
5  Dim B(A,A),C(A,A)
6  Deftext 2
7  I=0
8  Goto Affiche
9  Calcul:
10 For F=1 To A
11   For G=1 To A
12     S=0
13     For I=-1 To 1
14       For J=-1 To 1
15         If F+J<>0 And G+I<>0 And F+J<>A+1 And G+I<
>A+1 Then
16           S=S+B(F+J,G+I)
17         Endif
18       Next J
19     Next I
20     S=2*S-B(F,G)
21     C(F,G)=0
22     If S>=5 And S<=7 Then
23       C(F,G)=1
24     Endif
25   Next G
26 Next F
27 Affiche:
28 Clearw 2
29 K=0
30 For F=1 To A
31   For G=1 To A
32     If I=0 Then
33       Print At(25,2);"LIGNE ";F;At(25,4);"COLONNE
";G
34       Print At(25,6);"0 ou 1 ? ";
35       R$=Upper$(Input$(1))
36       C(F,G)=Val(R$)
37     Endif
38     Print At(2*G,2*F);
39     If C(F,G)=1 Then
40       K=1
41       Print "*"
42     Endif
43     B(F,G)=C(F,G)
44   Next G
45 Next F
46 Sound 1,10,5,6,5
47 Sound 1,0
48 If K=1 Then
49   Goto Calcul
50 Endif
51 Color 1
52 For I=1 To 5 Step 2
53   Sound 1,10,1,5,5
54 Next I
55 Sound 1,0
56 Print At(1,17);"C'EST FINI..."
57 R$=Input$(1)

```

Un taquin de calcul

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : difficile.

LE JEU :

Il se pratique à deux joueurs, l'un tenant le rôle du codeur, l'autre celui du décodeur.

Le codeur part d'un nombre (proposé par le ST) et doit le transformer à l'aide de 5 opérations qu'il choisit lui-même, ainsi d'ailleurs que les opérateurs.

Le résultat qu'il a obtenu et les opérations qu'il a utilisées seront affichés sur l'écran : le décodeur devra alors essayer de retrouver le nombre initial en effectuant les opérations en sens inverse.

LE PROGRAMME :

La ligne 5 permet d'obtenir l'entier de départ compris entre 5 et 20.

Une boucle I (12-35) permet d'entrer successivement les 5 opérations.

4 sous-programmes sont utilisés pour les calculs (aux lignes 53 à 64).

Après chaque calcul, le résultat est affiché (seul le codeur devra regarder l'écran pendant cette opération) et le résultat définitif restera affiché pour le décodeur à la fin des calculs (ligne 28).

La boucle I (38-45) affiche les 5 opérations utilisées, ainsi que les opérateurs correspondants.

POUR JOUER :

On frappe RUN pour lancer le programme.

Le nombre du début est affiché, puis la question ENTREZ 5 OPERATIONS ; on devra les entrer l'une après l'autre, sous la forme, par exemple : + 4 suivi de RET.

Le résultat sera affiché après chaque opération entrée. A la fin, les opérations entrées sont toutes affichées sur une même ligne, et le message BON COURAGE est envoyé à l'adresse du décodeur (il n'y a pas de contrôle au décodage).

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra, bien sûr, songer à améliorer la partie décodage du programme en introduisant des contrôles sur le résultat ; on peut aussi envisager de laisser le ST participer davantage en lui confiant le rôle du codage, etc.

```

1  Rem UN TAQUIN DE CALCUL
2  Fullw 2
3  Clearw 2
4  Dim R$(6),T(6)
5  X=Random(16)+5
6  Sound 1,10,X,5,10
7  Sound 1,0
8  Print At(1,2);"NOMBRE INITI
  AL :";X
9  Print At(1,4);"ENTREZ CINQ
  OPERATIONS"
10 Print At(1,6);"SOUS LA FORM
  E +5, /2, *4,... "
11 F=X
12 For I=1 To 5
13   Repeat
14     Deftext 2
15     Print At(1,9);
16     Input "QUELLE OPERATION
  : ";R$(I)
17     Print At(1,9);Spc(30)
18     T(I)=0
19     A=Asc(Left$(R$(I),1))-4
20     1
21     Sound 1,10,4,3,8
22     Sound 1,0
23     Until A>0 And A<>3 And A<
  >5 And A<=6
24     A=A+((A/2)-1)*(A>3)
25     On A GOSUB Multi,Addit,So
  ust,Divis
26     Print At(1,11);Spc(30)
27     F$=Str$(F)
28     Deftext 3
29     Print At(1,11);"NOMBRE OB
  TENU : ";Left$(F$,1);
30     F$=Right$(F$,Len(F$)-1)
31     For J=1 To Len(F$)
32       Print Chr$(Asc(Mid$(F$,
  J,1)));
33       Sound 1,20,#(Val(F$)*40
  ),Random(30)+20
34     Next J
35     Sound 1,0
36   Next I
37   Deftext 4
38   Print At(1,13);"OPERATIONS
  : ";
39   For I=1 To 5

```

```

39 Repeat
40   X=Random(5)+1
41   Until T(X)<>1
42   T(X)=1
43   Deftext X
44   Print R$(X);" ";
45 Next I
46 Print At(1,15);"BON COURAGE
...."
47 For I=1 To 5 Step 2
48   Sound 1,20,I,5,10
49 Next I
50 Sound 1,0
51 Deftext 1
52 R$=Input$(1)
53 Procedure Multi
54   F=F*Val(Right$(R$(I),Len(
R$(I))-1))
55 Return
56 Procedure Addit
57   F=F+Val(Right$(R$(I),Len(
R$(I))-1))
58 Return
59 Procedure Soust
60   F=F-Val(Right$(R$(I),Len(
R$(I))-1))
61 Return
62 Procedure Divis
63   F=F/Val(Right$(R$(I),Len(
R$(I))-1))
64 Return

```


Belote

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de cartes.

Difficulté du jeu : facile.

LE JEU :

Le jeu proposé est plutôt une simulation, réalisée sur 32 cartes, de la distribution de 4 paquets de 5 cartes. Il pourra être utilisé pour l'étude de problèmes faisant intervenir des données de 20 cartes sur 32.

LE PROGRAMME :

En ligne 4, on crée un tableau pouvant contenir les 20 cartes.

La boucle I (12-19) assure le tirage aléatoire de 20 nombres, pris parmi 32 ; la boucle 13-15 vérifie que l'on ne réalise pas deux fois le même tirage.

La deuxième boucle I (21-35) assure ensuite l'affichage, les noms des cartes étant affichés en clair, et la couleur étant décodée et affichée en clair, grâce à la ligne 22.

POUR JOUER :

Il suffit de taper RUN pour obtenir immédiatement l'affichage de 4 paquets de 5 cartes chacun (une donne).

Un nouveau RUN permet d'obtenir une nouvelle distribution.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra facilement mettre au point un affichage plus graphique en utilisant les SPRITE ; on pourra également se servir de ce programme comme d'un noyau pour la mise au point de programmes de jeux de cartes sur l'ordinateur (qui pourra, si nécessaire, servir de partenaire : voir l'exemple du Poker, dans le même ouvrage).

```

1  Rem BELOTE
2  Fullw 2
3  Clearw 2
4  Dim S$(10),A(20),T(32)
5  For I=1 To 8
6      Read S$(I)
7  Next I
8  F=7+3*(Xbios(4)=0)
9  G=-4*(Xbios(4)=2)
10 K=1
11 Print At(4*G+10,9);"PATIENTEZ, JE TIRE"
12 For I=1 To 20
13     Repeat
14         A(I)=Random(32)+1
15     Until T(A(I))=0
16     T(A(I))=1
17     Sound 1,10,Random(12)+1,Random(5)+3,1
18     Sound 1,0
19 Next I
20 Clearw 2
21 For I=1 To 20
22     T=-((A(I)<=8)+(A(I)<=16)+(A(I)<=24)+(A(I)<=32))
23     S=1+A(I)-8*Int(A(I)/8)
24     R$=Str$(S+6)+" "
25     R$=Left$(R$,2)
26     If S>=5 Then
27         R$=S$(S)
28     Endif
29     Deftext T,0,0,17+10*(Xbios(4)<2)
30     Print At((F+G+4)*Int((I-1)/5)+2,2*K);R$;" ";Left
$(S$(T),F)
31     K=K+1
32     Sound 1,10,A(I)/2,S/2,4
33     Sound 1,0
34     K=K+(K-1)*(K=6)
35 Next I
36 Deftext 1,0,0,13
37 R$=Input$(1)
38 Data TREFLE,COEUR,CARREAU,PIQUE,VA,DA,RO,AS

```

Hanoi

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion.

Difficulté du jeu : difficile.

LE JEU :

Il est présenté, ici, dans une version pour un joueur. On dispose d'un certain nombre de disques (8 au plus) et l'on doit les transférer de la tour numéro 1 à la tour numéro 3, en respectant une seule règle : à aucun moment, un disque ne peut être recouvert par un disque de diamètre supérieur.

Le nombre de coups n'est pas comptabilisé par ce programme mais, en revanche, les coups illicites seront refusés.

LE PROGRAMME :

La variable E\$, en ligne 8, servira à dessiner les disques, et D\$ dessine les socles.

Le nombre de disques est entré en ligne 15 et refusé s'il y a plus de 8 disques.

La boucle I (17-19) permet de fabriquer une variable A\$ (1) qui contient l'état de la première pile sous forme numérique (en ligne 3, trois variables sont prévues pour conserver l'état des trois piles).

Les socles sont dessinés par la ligne 24 et les trois piles par les boucles emboîtées I (27-36) et J (28-35).

La ligne 38 contrôle la fin de la partie (tours empilées sur le socle 3).

Le contrôle du coup est effectué par la ligne 49 et l'état des piles est modifié par les lignes 52 et 53.

POUR JOUER :

Après un RUN qui lance le programme, on entre le nombre de disques demandé, suivi de RET. On obtient le dessin de l'état initial et, à la question TOUR DE DEPART, on répond en donnant le numéro de la tour de départ. Ce numéro (1, 2 ou 3) sera refusé si la tour est vide. Puis on tape RET et, à la question TOUR D'ARRIVEE, on répond par le numéro de la tour d'arrivée.

Si le coup n'est pas possible, on sera obligé de recommencer à TOUR DE DEPART. Si le coup est possible, le déplacement est réalisé et l'on recommence.

On peut arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra dessiner des disques de différentes couleurs, introduire un compteur de coups et, surtout, définir un jeu pour l'ordinateur.

On peut aussi essayer de compliquer le jeu avec deux tiges de plus et deux joueurs, chacun ayant des disques de couleur différente, par exemple.

```

1  Rem LES TOURS DE HANDI
2  Fullw 2
3  Dim A$(3)
4  H=28+13*(Xbios(4)=0)
5  K=17+11*(Xbios(4)=0)
6  U=13-14*(Xbios(4)>0)
7  D$=String$(20+7*(Xbios(4)=0), "=")
8  E$=String$(18, "#")
9  A$(1)="9"
10 A$(2)="9"
11 A$(3)="9"
12 Deftext 1,0,0,13+7*(Xbios(4)<2)
13 Repeat
14   Clearw 2
15   Input "NOMBRE DE TOURS ( <9 ) : ";N
16   Until N=Int(N) And N>0 And N<9
17   For I=0 To N-1
18     A$(1)=A$(1)+Right$(Str$(N-I),1)
19   Next I
20   T$=A$(1)
21   Do
22     Clearw 2
23     Deftext 3
24     Print At(1,11);D$;At(U+1,11);D$;At(2*U+1,11);D$
25     Deftext 1
26     Print At(6,11);"1";At(6+U,11);"2";At(6+2*U,11);"3"
27     For I=0 To N-1
28       For J=3 To 1 Step -1
29         B$=A$(J)
30         F=Val(Mid$(B$,I+2,1))
31         Deftext F+3
32         If Len(B$)>I+1 Then
33           Print At(H*KJ-F-K,10

```

```

-I);Left$(E$,F-F*(Xbios(4)>0))
34   Endif
35   Next J
36   Next I
37   Deftext 1
38   Exit If A$(3)=T$
39   Saisie:
40   Print At(1,13);Spc(30);At(1,13);
41   Input "TOUR DE DEPART : " ;D
42   If D<>Int(D) Or D>3 Or D<1 Or A$(D)="9" Then
43     Goto Saisie
44   Endif
45   Print At(1,15);Spc(3);At(1,15);
46   Input "TOUR D'ARRIVEE : " ;A
47   B$=A$(D)
48   C$=A$(A)
49   If A<>Int(A) Or A=D Or A>3 Or A<1 Or Mid$(B$,Len(B$),1)>Mid$(C$,Len(C$),1) Then
50     Goto Saisie
51   Endif
52   A$(A)=C$+Mid$(B$,Len(B$),1)
53   A$(D)=Left$(B$,Len(B$)-1)
54   Sound 1,10,4,5,8
55   Sound 1,0
56 Loop
57 Print At(1,17);"BRAVO....." ;
58 For I=1 To 12
59   Sound 1,10,I,5,8
60 Next I
61 Sound 1,0
62 R$=Input$(1)

```

Calculatrice

REFERENCES :

Difficulté de programmation : très difficile.

Catégorie de jeu : simulation d'une calculatrice.

Difficulté du jeu : moyenne.

LE JEU :

Il s'agit de la simulation, en affichage grand format, du fonctionnement d'une petite calculatrice de poche, de type 4 opérations. Les applications pédagogiques de ce programme sont évidentes et son fonctionnement très simple à maîtriser.

LE PROGRAMME :

Les variables B\$ et C\$ des lignes 9 et 10 contiennent les informations codées qui permettront le dessin des chiffres en grand format.

Les lignes 23 à 46 s'occupent de la gestion des entrées.

La boucle I (53-58) réalise la représentation, en grand format, des chiffres sur l'écran, au fur et à mesure des entrées.

Le point décimal et le signe — sont gérés par les lignes 63 et 67.

Le nombre total des symboles entrés est contrôlé par la ligne 70.

Le résultat est inscrit grâce à un programme fonctionnant en automatique qui se trouve aux lignes 83 à 105 (l'automatisme est géré par les lignes 71 à 75, dans le programme principal).

Les sous-programmes placés aux lignes 106, 111, 118 et 125 effectuent l'opération indiquée, selon la valeur de Z (ligne 94).

Chaque résultat est mémorisé en A (F-1) (ligne 100).

POUR JOUER :

On devra lancer le programme par RUN.

Puis, on tape les chiffres du premier nombre à entrer qui vont s'afficher un par un, au fur et à mesure (on ne pourra entrer plus de 8 symboles, y compris le signe — et le point décimal). La frappe d'un signe opératoire laisse l'affichage intact, mais l'introduction du prochain chiffre ou symbole provoque l'effacement immédiat et l'affichage du nouveau symbole. Il est à remarquer que les chiffres s'introduisent de gauche à droite, sens de l'écriture usuelle.

L'appareil accepte, comme signes opératoires les symboles +, —, *, / ; le signe = provoquera l'affichage du résultat. Les opérations sont enchaînées automatiquement, le résultat précédent étant pris en compte comme opérateur du calcul suivant.

Pour faire une autre opération, il faudra un espace, puis RUN.

Taper la combinaison CSA pour sortir du programme.

EXTENSIONS POSSIBLES :

On devra, tout d'abord, améliorer l'affichage du signe — (représenté, ici, par un trait), puis on pourra améliorer le dessin des chiffres (en utilisant, par exemple, le générateur du ST), on peut aussi envisager une extension à plus de 8 chiffres, autoriser des opérations plus complexes (COS, SIN, ...), ajouter des mémoires, etc.

```

1  Rem CALCULATRICE
2  Clearw 2
3  Fullw 2
4  Titlew 2,"CALCULATRICE"
5  Deftext 2,17,0,15+9*(Xbios(
4)<2)
6  Cn=4-4*(Xbios(4)>0)
7  B=32-32*(Xbios(4)>0)
8  Dim A(9),B$(5)
9  C$="111100001101"
10 B$="14441333331312113131441
331213112141133331414114131
"
11 B$(1)=""
12 B$(2)=""
13 B$(3)=Chr$(240)
14 B$(4)=Chr$(255)+Chr$(255)
15 B$(5)=Chr$(47)
16 F=1
17 L=0
18 Debut:
19 T=0
20 N=0
21 K=0
22 R$=""
23 Entree:
24 D$=Input$(1)
25 Sound 1,10,4,7,10
26 Sound 1,0
27 If D$="" Then
28 End
29 Endif
30 If K=1 Then
31 N=0
32 R$=""
33 K=0
34 Endif
35 If D$="" Then
36 Goto Calcul
37 Endif
38 X=(1 And D$="+")+(2 And D$=
"-")+(3 And D$="*")+(4 And
D$="/")
39 R$=R$+D$
40 If X<>0 Then
41 Goto Calcul
42 Endif
43 Change:
44 If R$<>"-" Then
45 If D$<"." Or D$>"9" Or D$
="/" Then
46 Goto Entree
47 Endif
48 Efface:
49 If N=0 Then
50 Clearw 2
51 Endif
52 If D$<>"." And D$<>"-" Th
en
53 For I=1 To 5
54 S=3*Val(Mid$(B$,I+5*V
al(D$),1))-2
55 For J=0 To 2
56 Print At(N+J+1,I+3)
;B$(Val(Mid$(C$,S+J,1))+2)
57 Next J
58 Next I
59 Endif
60 Else
61 D$=R$
62 Endif
63 If D$="-" Then
64 Clearw 2
65 Print At(2,6);B$(4)
66 Endif
67 If D$="" Then
68 Print At(N+2,8);B$(5)
69 Endif
70 N=N-Cn*(N<B)
71 If T<>0 Then
72 T=T+1
73 If T<=Len(R$) Then
74 Goto Avance
75 Endif
76 Endif
77 If K=1 And L<>1 Then
78 Goto Debut
79 Endif
80 T=0
81 Goto Entree
82 Calcul:
83 If R$="-" And A(F)=0 Then
84 Goto Change
85 Endif
86 N=0
87 K=1
88 If (D$="*" Or D$="/") And V
al(R$)=0 Then
89 A(F)=1
90 Else
91 A(F)=Val(R$)
92 Endif
93 Z=Z+(Z-X)*(X<>0)
94 On Z Gosub Add,Sous,Multi,D
ivis
95 If D$<>"=" Then
96 F=F+1
97 Else
98 F=2
99 Endif
100 A(F-1)=Val(R$)-A(F)*(Val(R$
)=0)
101 R$=Mid$(Str$(Val(R$)),1-(Le
ft$(R$,1)=" "))
102 T=1
103 Avance:
104 D$=Mid$(R$,T,1)
105 Goto Efface
106 Procedure Add
107 R$=Str$(A(F-1)+A(F))
108 A(F)=Val(R$)
109 L=1
110 Return
111 Procedure Sous
112 If F<>1 Then
113 R$=Str$(A(F-1)-A(F))
114 A(F)=Val(R$)
115 L=1
116 Endif
117 Return
118 Procedure Multi
119 If F<>1 Then
120 R$=Str$(A(F-1)*A(F))
121 A(F)=Val(R$)
122 L=1
123 Endif
124 Return
125 Procedure Divis
126 If F<>1 Then
127 R$=Str$(A(F-1)/A(F))
128 A(F)=Val(R$)
129 L=1
130 Endif
131 Return

```


Niveau 3

Nom du programme

Thèmes et particularités

Visions	Un jeu sur les mots, pour l'ordinateur.
Mathieu	Vous ne savez pas dessiner ? Laissez faire votre ordinateur.
Plus grand, plus petit	Un jeu de déduction intéressant, pour un joueur.
La bombe	Saurez-vous la découvrir à temps, pour un seul joueur.
Mots	Une idée très simple, mais ce n'est pas si facile à jouer.
Idem	Un jeu de calcul mental, avec plusieurs niveaux, pour plusieurs joueurs.
Mastermind	Avec des chiffres, mais très classique, pour un joueur.
Nicomaque	Pour se perfectionner en calcul mental, pour un ou plusieurs joueurs.
Le pendu	Un jeu de lettres très classique, pour deux joueurs.
Accrochez les wagons	Pour les plus jeunes : très spectaculaire.
Le loup et les agneaux	Un jeu de réflexion, pour deux joueurs.
Le lézard d'Alice	Beaucoup de hasard, mais très spectaculaire.
Reines sur l'échiquier	Un vieux problème, pour vous aider à réfléchir.
Un cube	En 3D, et mobile...
Le jeu du missile	Une mission bien périlleuse, pour un seul joueur.
Chasse au sous-marin	Il vous faudra pas mal de réflexion, et beaucoup de chance.
Ordre	Un peu d'astuce pour remettre en ordre, le plus vite possible.
Bridge	Mettez au point vos propres problèmes de bridge.
A travers	Un jeu de stratégie, pour deux joueurs.
Evasion	Deux versions pour mieux réfléchir...
Permutations	Un jeu de réflexion, pas toujours évident.
Répétitions	Retrouver une séquence de chiffres de plus en plus longue.

Graphisme et jeux

INSTRUCTIONS GRAPHIQUES

Le GFA Basic supporte tous les modes graphiques de l'Atari ST, à savoir le mode 640*400 (monochrome), le mode moyenne résolution (640*200, en 4 couleurs) et le mode basse résolution (320*200 en 16 couleurs). Le mode de fonctionnement est défini a priori, par exemple par le choix du moniteur, et ne peut être modifié en cours d'utilisation. Tous les programmes du livre sont prévus pour fonctionner dans les trois modes.

Nous avons déjà rencontré, dès le premier chapitre, une instruction qui permet d'afficher des textes en couleur (instruction **DEFTTEXT N**, N étant le numéro de la couleur choisie). En principe, cette instruction est étudiée pour fonctionner conjointement à l'instruction **TEXT**, qui permet l'affichage de texte en mode graphique (la position d'écriture est dans ce cas donnée par rapport à l'écran graphique, et non par rapport aux coordonnées standards d'un texte). L'instruction **DEFTTEXT** agit cependant aussi sur les textes affichés avec **PRINT**, lorsque l'on travaille en mode plein écran (**FULLW 2**). Dans le cas contraire, il est néanmoins possible d'afficher du texte en couleur, à l'aide de séquences dites "Escape". Cela consiste à préparer l'affichage en envoyant derrière l'ordre **PRINT** une succession de caractères commençant par **CHR\$(27)** (caractère Escape).

Ainsi **PRINT AT(12,12) ; CHR\$(27) ; "b" ; 2 ; "BONJOUR"** affiche le mot bonjour dans la couleur 2.

L'instruction **COLOR N** permet de choisir la couleur N pour les tracés effectués avec des instructions graphiques.

L'instruction **DEFFILL** permet de choisir la couleur et le motif de remplissage d'une figure.

L'instruction **DEFLINE** permet de choisir les caractéristiques des lignes à tracer.

L'instruction **DEFTTEXT** permet de choisir les caractéristiques des textes.

L'instruction **PLOT X,Y** affiche un point à l'emplacement X,y.

L'instruction **LINE** tire un trait entre deux points.

L'instruction **DRAW** permet d'enchaîner une série de traits les uns aux autres.

L'instruction **BOX** trace des rectangles, **PBOX** trace des rectangles pleins, **RBOX** des rectangles arrondis et **PRBOX** des rectangles pleins et arrondis.

CIRCLE trace des contours circulaires, **PCIRCLE** des disques, **ELLIPSE** des contours elliptiques, et **PÉLLIPSE** des ellipses pleines.

FILL permet de colorer une figure fermée.

AFFICHAGE D'UN TEXTE GRAPHIQUE : TEXT

Il est possible d'afficher du texte à n'importe quel emplacement de l'écran graphique, en utilisant l'instruction **TEXT** sous la forme **TEXT X,Y,A\$**.

La taille et le style des caractères à afficher peuvent être définis au préalable par **DEFTEXT**.

PROCEDURES AVEC PASSAGE DE PARAMETRES

La notion de procédure, que nous avons rencontré dès le premier chapitre sous l'appellation de sous-programme, recouvre en fait une réalité beaucoup plus complexe que les simples sous-programmes des anciens Basic, qui n'étaient en fait que des parties du programme principal.

En Basic GFA, les procédures sont indépendantes du programme principal, qui ne peut y accéder que s'il y a eu un appel (il est donc inutile de terminer un programme par **END**, lorsqu'il se continue par des procédures : le programme ne pourra pas y entrer). De plus, les variables peuvent être définies comme locales (instruction **LOCAL**) à la procédure, ce qui permet d'écrire des bibliothèques de sous-programmes complètement autonomes. En complément, il est possible de transmettre lors de l'appel des valeurs, ou même des variables, à la procédure. Il suffit de faire suivre l'appel de la procédure de la liste des paramètres à transmettre (cette liste devra être reprise sous forme concordante dans la définition même de la procédure).

Exemple

A=10

GOSUB calcul(A)

PRINT B

GOSUB calcul(30)

PRINT B

Procédure calcul(K)

B=5*K+3

RETURN

Visions

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin.

Difficulté du jeu : facile.

LE JEU :

Le ST vous permet, pour la première fois, d'entrer dans son intimité en vous laissant entrevoir sa propre représentation du monde ! Vous allez, en effet, lui proposer des mots, n'importe lesquels, et il va vous répondre en vous proposant une représentation graphique de ce que représente ce mot pour lui. Le jeu sera beaucoup plus amusant si vous le pratiquez à plusieurs, et vous pourrez quelquefois avoir des surprises.

LE PROGRAMME :

Une boucle I (27-43) se charge de réaliser les dessins.

La ligne 28 permet de transformer en nombre chaque lettre du mot entré au début (ligne 9). Ce nombre sera ensuite traduit en base 2 par une boucle J (lignes 30 à 42) ; selon le résultat de cette transformation, un cercle sera dessiné, colorié ou non (lignes 37 et 39).

POUR JOUER :

On débute l'exécution du programme par RUN, ce qui amène l'affichage de la question quel dessin voulez-vous ? On répond en entrant un mot quelconque qui sera transformé en dessin par le ST dès que l'on aura frappé RET.

Il faudra taper la combinaison CSA pour sortir du programme.

EXTENSIONS POSSIBLES :

La même idée peut être reprise en introduisant des éléments de symétrie dans les dessins, en mettant plusieurs couleurs. On peut aussi imaginer de coder par cette méthode un certain nombre de dessins, que l'on introduirait au fur et à mesure dans la machine et l'on pourra alors créer un jeu dans lequel le ST deviendra de plus en plus savant. On peut aussi réaliser un jeu à deux joueurs dans lequel le ST vous demande de trouver le mot qui correspond à un certain dessin, etc.

```

1  Rem VISIONS
2  Fullw 2
3  Clearw 2
4  Deftext 1,0,0,13
5  Input "ECRAN 1:PETIT 2:MOYEN 3:GRAND :";R
6  Clearw 2
7  Do
8      Deftext 1,,,6
9      Input "QUEL DESSIN VOULEZ-VOUS :";R$
10     Repeat
11         F=Random(16)
12         C=Random(16)
13         Until Abs(F-C)>=2
14         If R>1 Then
15             C=Random(4)
16             F=3-C
17             If R=3 Then
18                 C=2
19                 F=1
20             Endif
21         Endif
22         Clearw 2
23         Deffill F,2,4
24         Fill 1,1
25         Deffill F,1
26         Color C
27         For I=1 To Len(R$)
28             N=9*Asc(Mid$(R$,I,1))-340
29             J=1
30             While J<10
31                 S=Frac(N/2)
32                 N=Int(N/2)
33                 B=Random(12)+1
34                 Sound 1,10,B,5,5
35                 Sound 1,0
36                 If S<>0 Then
37                     Pcircle 16*J+70*R,(16+8*(R=2))*I+24,8
38                 Else
39                     Circle 16*J+70*R,(16+8*(R=2))*I+24,8
40                 Endif
41                 Inc J
42             Wend
43         Next I
44         Deftext 1
45         Print At(1,1);"DESSIN DE ";R$
46         Print
47     Loop

```

Mathieu

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin aléatoire.

Difficulté du jeu : aucune.

LE JEU :

On utilise la définition d'une courbe en coordonnées polaires et la fonction PLOT pour obtenir sur l'écran des dessins aléatoires (de type fleur), plus ou moins élégants.

Les deux paramètres principaux A et B pourront être, si nécessaire, entrés à la main (instructions INPUT) pour faire une étude plus systématique des différentes courbes possibles.

LE PROGRAMME :

Les 4 premiers paramètres Z, K, T et L, introduits aux lignes 5 à 8, définissent le rayon maximal et l'occupation de l'écran.

Les variables aléatoires A et B, choisies entre 0 et 3, des lignes 19 et 20, définissent l'allure de la courbe ; il pourra être intéressant de les introduire manuellement (par exemple : INPUT A, B).

La boucle H (21-27) réalise le tracé, point par point. Le rayon R est calculé en ligne 22, et le passage en coordonnées cartésiennes est effectué aux lignes 23 et 24.

L'affichage des points est réalisé en ligne 26.

La ligne 28 imprime un message de fin.

La ligne 34 permet de refaire un nouveau dessin, sans effacer l'écran.

POUR JOUER :

On lance le programme en tapant RUN.

Le dessin sera ensuite exécuté automatiquement.

Le message "c'est fini" signale la fin du tracé.

Si on veut un nouveau tracé, on tape une touche quelconque.

Si on veut refaire un tracé sans effacer le précédent, on tape la barre d'espace.

Pour arrêter, taper 0, ou la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra essayer d'utiliser d'autres fonctions pour réaliser des tracés encore plus extraordinaires (par exemple, avec simulation de la troisième dimension).

```

1  Rem MATHIEU
2  P=Xbios(4)
3  Fullw 2
4  Openw 2
5  Z=9
6  K=18+11*(P<2)
7  L=24+12*(P=0)
8  T=3
9  Pa=0.6-0.7*(P=0)
10 U=300+150*(P=0)
11 V=200+100*(P<2)
12 Repeat
13   Clearw 2
14   If P<>2 Then
15     Setcolor 0,Random(1912)
16   Endif
17   Debut:
18   C=Random(4^(2-P))-(P=2)
19   A=3*Rnd
20   B=3*Rnd
21   For H=0 To 6.28 Step Pa/180
22     R=Z*Sin(H*T)
23     X=U+Int(L*R*Cos(A*H))
24     Y=V+Int(K*R*Sin(B*H))
25     Color C
26     Plot X,Y
27   Next H
28   Print At(1,18);"C'EST FINI...";
29   For I=1 To 12
30     Sound 1,10,I,5,7
31   Next I
32   Sound 1,0
33   R$=Input$(1)
34   If R$=" " Then
35     Goto Debut
36   Endif
37 Until R$<>"0"
38 Setcolor 0,1911

```

Plus grand, plus petit

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de déduction (un seul joueur).

Difficulté du jeu : difficile.

LE JEU :

Il s'inspire d'un principe légèrement différent de celui du Mastermind ; il s'agit également de deviner un nombre (de 1 à 7 chiffres), mais les indications dont on dispose sont cette fois, d'une part, le nombre de chiffres qui, dans le nombre essayé, sont supérieurs à ceux du nombre à deviner, et, d'autre part, le nombre de chiffres qui sont inférieurs. Si, par exemple, le nombre à deviner est 3054, et que l'on essaie 4251, la machine répondra 2 plus grands (ce sont le 4 et le 2), et 1 plus petit (c'est le 1).

LE PROGRAMME :

La ligne 11 interdit de demander des nombres de plus de 8 chiffres.

Le nombre à deviner est placé en ligne 14, dans une variable alphanumérique X\$.

Les comparaisons avec le nombre essayé sont faites à la ligne 23 (qui teste si l'on a trouvé juste), et dans la boucle I (28-35) où l'on fait la comparaison "lettre à lettre".

L'affichage du résultat est réalisé grâce aux lignes 37 et 39.

Le compteur du nombre d'essais est en ligne 41.

POUR JOUER :

Le lancement du programme par RUN provoque l'affichage NOMBRE DE CHIFFRES.

On entre un nombre inférieur à 8 et on appuie sur la touche RET.

Le numéro de l'essai s'affiche, ainsi que la question NOMBRE ESSAYE ; on répond en entrant un nombre, suivi de RET.

Le résultat est alors affiché sous la forme > et <.

S'il n'est pas exact, on recommence (attention à entrer uniquement des nombres ayant autant de chiffres que le nombre à deviner).

Si l'on a trouvé le ST affichera VOUS AVEZ TROUVE... EN... ESSAIS. Puis AUTRE PARTIE (O/N) ; on fera O pour commencer une nouvelle partie.

On devra taper la combinaison CSA pour arrêter.

EXTENSIONS POSSIBLES :

On pourra, comme pour le Mastermind, imaginer un jeu analogue avec des lettres, des couleurs, ou même en utilisant d'autres symbolismes.

On peut aussi prévoir de laisser jouer le ST, qui tiendra alors le rôle de décodeur.

```

1  Rem PLUS GRAND, PLUS PETIT
2  Fullw 2
3  K=8-8*(Xbios(4)>0)
4  H=40+20*(Xbios(4)<2)
5  For I=0 To Xbios(4)
6    Read G1,G2,G3,G4
7  Next I
8  Repeat
9    Input "NOMBRE DE CHIFFRES :";C
10   Clearw 2
11  Until C<=8
12  E=1
13  G=1
14  X$=Str$(Int(Rnd*9^C+10^(C-E)))
15  Deffill 3,2,14
16  Pbox 0,5,639,H
17  Line K*10,H+1,K*10,399
18  Line K*20,H+1,K*20,399
19  Line K*30,H+1,K*30,399
20  Deftext 1
21  Print At(G1,2);"ESSAI NO. ";At(G2+4,2);"NBRE. ESSAY
   E"
22  Print At(G3+6,2);" > ";At(G4+8,2);" < "
23  While P$<>X$
24    Print At(G1+2,G+4);E;At(G2+6,G+4);Spc(C);At(G2+6
   ,G+4);
25    Input P$
26    A=0
27    B=0
28    For I=1 To C
29      If Mid$(P$,I,1)>Mid$(X$,I,1) Then
30        A=A+1
31      Endif
32      If Mid$(P$,I,1)<Mid$(X$,I,1) Then
33        B=B+1
34      Endif
35    Next I
36    Deftext 3
37    Print At(G3+8,G+4);A
38    Deftext 2
39    Print At(G4+10,G+4);B
40    Deftext 1
41    E=E+1
42    G=G+1+13*(G=13)
43  Wend
44  Print
45  Print "VOUS AVEZ TROUVE EN ";E;" ESSAIS"
46  For I=1 To 5 Step 2
47    Sound 1,10,I,5,8
48  Next I
49  Sound 1,0
50  Print
51  Print "AUTRE PARTIE (O/N) : "
52  R$=Upper$(Input$(1))
53  If R$="O" Then
54    Run
55  Endif
56  Data 1,6,17,25,1,20,43,61,4,20,43,61

```


La bombe

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : solitaire.

Difficulté du jeu : moyenne.

LE JEU :

Vous êtes censé trouver, à l'aide d'un détecteur spécialisé, un colis radioactif piégé disposé par un voyageur mal intentionné dans une chambre d'hôtel. Seul problème, mais de taille : l'hôtel comporte un million de chambres ($100*100*100$).

Votre détecteur affiche, en permanence, un signal qui vous renseigne sur la proximité de la bombe, et le temps qui vous reste est affiché en permanence, décompté de 10 en 10.

Pourrez-vous découvrir la bombe avant l'explosion ?

LE PROGRAMME :

Une variable numérique R, dimensionnée en ligne 4, contient les trois coordonnées définissant la position du joueur.

La position de la chambre contenant le colis piégé est définie par un tirage aléatoire de 3 variables A, B, C, aux lignes 5, 6 et 7.

La ligne 9 affiche un titre en haut de l'écran.

Les lignes 11 et 12 dessinent le détecteur.

Le signal de proximité est calculé aux lignes 14 et 15.

Les lignes 16 et 17 permettent l'affichage, à la fois, du signal et du temps restant à courir.

La ligne 20 réalise le décompte du temps.

La ligne 21 contrôle si l'on a trouvé ou non.

La boucle I (29-33) permet d'entrer les nouvelles coordonnées, en utilisant le sous-programme placé aux lignes 43 à 55.

La ligne 36 affiche les coordonnées de la chambre à découvrir, dans le cas où l'on a perdu.

POUR JOUER :

Après le lancement du programme par RUN, le signal et le temps (en principe 200) sont affichés.

Le joueur est alors en position $x=0$, $y=0$, $z=0$, et on devra introduire successivement les trois coordonnées x, y, z de la chambre à explorer.

On fera, par exemple, 13 puis RET, 45 puis RET, etc.

Le nouveau signal est alors affiché, et le temps est décrémenté de 10.

On continuera ainsi, soit jusqu'au bon résultat (arrêt du jeu et fin de programme), soit jusqu'à l'apparition de 3 nombres (x,y,z) sans sortie du programme, ce qui signifie alors que l'on a perdu.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera sans doute indispensable d'améliorer le graphisme, d'ajouter une explosion (sonore et visuelle) pour symboliser l'éclatement de la bombe, d'afficher symboliquement la position dans l'hôtel, etc.

```

1  Rem LA BOMBE
2  L=100
3  T=200
4  Dim R(3)
5  A=Random(L)
6  B=Random(L)
7  C=Random(L)
8  Clearw 2
9  Titlew 2,"LA BOMBE"
10 Fullw 2
11 Deffill 2*Abs(Xbios(4)-2)+1,2,2
12 Pbox 20,20+10*(Xbios(4)<2),190,100+50*(Xbios(4)<2)
13 While T<>0
14   S=Abs((A/L+B+C*L)-(R(1)/L+R(2)+R(3)*L))
15   S=Val(Left$(Str$(S),7))
16   Print At(7,4);Using "####.##",S
17   Print At(17,4);Using "###",T
18   Sound 1,10,4,5,6
19   Sound 1,0
20   T=T-Int(Sqr(L))
21   If S=0 Then
22     Print
23     Print "BRAVO"
24     Goto Fin
25   Endif
26   Print At(1,10);" - X :      "
27   Print At(1,12);" - Y :      "
28   Print At(1,14);" - Z :      "
29   For I=1 To 3
30     R$=""
31     Gosub Entree
32     R(I)=Val(R$)
33   Next I
34 Wend
35 Print
36 Print "PERDU. LA SOLUTION ETAIT :";A;" ";B;" ";C
37 Fin:
38 For I=1 To 12
39   Sound 1,10,I,5,8
40 Next I
41 Sound 1,0
42 R$=Input$(1)
43 Procedure Entree
44   Do
45     Print At(8,2*I+8);"?";R$+"      "
46     Sound 1,10,3*I,5,2
47     Sound 1,0
48     A$=Input$(1)
49     Exit If Asc(A$)=13
50     R$=R$+A$
51     If Len(R$)>2 Then
52       R$=""
53     Endif
54   Loop
55 Return

```

Mots

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de lettres (pour 1 ou plusieurs joueurs).

Difficulté du jeu : moyenne.

LE JEU :

Il est difficile d'imaginer plus simple dans son principe que ce jeu, qui se contente d'afficher dans le désordre les lettres prises dans une trentaine de mots. Mais essayez de jouer, et vous verrez que ce n'est pas si facile, la difficulté variant évidemment selon la liste de mots que vous aurez introduits en DATA au préalable.

LE PROGRAMME :

La ligne 6 permet de dimensionner la variable A\$(I) qui contiendra les mots.

La boucle I (10-12) saisit la liste des mots (inscrits en DATA aux lignes 74 à 78).

La ligne 16 choisit un mot au hasard ; la boucle I (22-28) contrôle si une lettre n'a pas déjà été tirée, et reprend les lettres dans le désordre.

La ligne 33 affiche les lettres, et la ligne 37 attend votre proposition, qui est testée en ligne 39.

Le nombre d'essais est incrémenté en ligne 49.

La fin de partie est contrôlée par la ligne 53, et le score est affiché en ligne 58.

POUR JOUER :

On lance le programme par RUN. Une série de lettres est affichée dans le désordre, et il faut essayer de retrouver le mot correspondant (le temps est contrôlé).

On tape le mot supposé, puis RET, et une nouvelle série de lettres sera affichée, si l'on répond O à la question AUTRE ESSAI (O/N).

Le score et le nombre d'essais sont affichés en fin de partie.

EXTENSIONS POSSIBLES :

On peut imaginer de minuter le temps d'apparition du mot, ou de faire défiler les lettres l'une après l'autre sur l'écran.

On peut aussi corser la difficulté en introduisant des JOKERS, jouer à plusieurs, tenir le compte des points, etc.

```

1  Rem MOTS
2  Fullw 2
3  P=Abs(Xbios(4)-2)
4  D=32+18*(P>0)
5  E=80+38*(P>0)
6  Dim A$(30)
7  S=0
8  U=0
9  F$=""
10 For I=1 To 30
11   Read A$(I)
12 Next I
13 Repeat
14   Clearw 2
15   Repeat
16     Z=Random(30)+1
17     Until A$(Z)<>""
18     R$=A$(Z)
19     A$=""
20     A$(Z)=""
21     V$=R$
22     For I=1 To Len(R$)
23       Repeat
24         Z=Random(Len(R$))+1
25         Until Mid$(R$,Z,1)<>"-"
26         A$=A$+Mid$(R$,Z,1)
27         Mid$(R$,Z,1)="-"
28     Next I
29     Deftext 4,0,0,20+13*(P>0)
30     Gosub Affiche(2,2,"LES LE
TTRES SONT",0)
31     L=Len(A$)*D+20-10*(P>0)
32     Rbox 4,E,L,E+32+10*(P>0)
33     Deftext 1,0,0,10+5*(P>0)
34     Gosub Affiche(8,2,A$,1)
35     Deftext 3,0,0,11+5*(P>0)
36     Gosub Affiche(12,1,"SOLUT
ION : ",0)
37     Input S$
38     S$=Upper$(S$)
39     If S$=V$ Then
40       S=S+1
41       For I=1 To 5
42         Sound 1,10,I,6,7
43       Next I
44     Else
45       Sound 1,10,2,3,20
46       F$=F$+V$+" "
47     Endif
48     Sound 1,0
49     U=U+1
50     Deftext 1,0,0,9+4*(P>0)
51     Gosub Affiche(19,2,"AUTRE
ESSAI (0/N) ",0)
52     Deftext 1,0,0,13+7*(P>0)
53     R$=Upper$(Input$(1))
54     Until R$<>"0"
55     Clearw 2
56     Print At(1,5);"VOUS N'AVEZ
PAS TROUVE : ";F$
57     Print
58     Print "SCORE ";S;" SUR ";U;
" ESSAIS"
59     For I=1 To 12
60       Sound 1,10,I,5,8
61     Next I
62     Sound 1,0
63     R$=Input$(1)
64     Procedure Affiche(R,S,P$,H)
65       For I=1 To Len(P$)
66         If H=1 Then
67           Deftext 1+Random(4^P)
68         Endif
69         Print At(2*I+S,R);Mid$(
P$,I,1);
70       Sound 1,10,I,5,1
71       Sound 1,0
72     Next I
73     Return
74     Data JAMBON,ENERGUMENE,AFFI
LIE,ERUDIT,INSIGNIFIANT,REA
LISTE
75     Data IMPRESSIONNANT,ALLECHA
NT,EPIQUE,INTREPID,COUARD,
ORDINATEUR
76     Data TREVE,BANDEROLLE,SENAT
,ACTUEL,CONSACRE,DIEDRE,LIV
RE
77     Data TERMINE,SIGNATURE,MALO
DORANT,AIMANT,ARBITRAIRE
78     Data CHALEUR,OPPORTUN,PELUR
E,ALTRUISTE,RAMONEUR,NOVICE

```

Idem

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de calcul mental (2 joueurs).

Difficulté du jeu : selon le niveau choisi.

LE JEU :

Le ST choisit un nombre de n chiffres (selon le niveau de difficulté indiqué au début). Ce nombre (le même au début pour les 2 joueurs) sera régulièrement décrémenté à chaque tour, d'une quantité également choisie au hasard par le ST en début de jeu ; vous devez, à chaque coup, choisir une colonne (les dizaines, les centaines, etc.) et ajouter dans cette colonne un certain entier ; le but est d'arriver à obtenir à un moment l'affichage d'un nombre ayant tous ses chiffres identiques.

Chaque résultat sera affiché un court instant, après quoi la main passera au joueur suivant, qui repartira de son précédent résultat.

LE PROGRAMME :

La ligne 12 permettra d'entrer le niveau de difficulté choisi.

Le nombre initial est confectionné par la ligne 16. La quantité qui interviendra dans les décrémentations est obtenue en ligne 20.

La ligne 27 affiche le numéro du joueur.

Le nombre est affiché en ligne 29 et le jeu du joueur est entré en ligne 36 (nombre de deux chiffres).

La ligne 38 incrémente la colonne choisie, et la ligne 39 décrémenté le nombre de la quantité choisie au départ.

Le résultat sera affiché un court instant (ligne 41). La boucle I (49-51) permet de vérifier si tous les chiffres sont identiques.

L'impression et la fin de partie sont effectuées aux lignes 52 à 61.

La ligne 22 s'occupe du changement de joueur.

POUR JOUER :

Lancer l'exécution du programme par RUN.

Le nombre initial sera affiché pour le joueur n° 1, qui joue en tapant XY, suivi de RET ; X est le chiffre à ajouter, et Y est le numéro de la colonne où il faut l'ajouter.

Le résultat (provisoire s'il est trop petit) est alors affiché un court instant, et le trait passe au deuxième joueur, et ainsi de suite. On peut arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra étendre le jeu à plus de 2 joueurs, améliorer le choix du nombre intervenant dans la décrémentation, améliorer les divers contrôles, laisser les différents résultats affichés, etc.

```

1  Rem IDEM
2  Fullw 2
3  Clearw 2
4  Dim N(2)
5  C=74+4*(Xbios(4)=2)
6  D=44-44*(Xbios(4)=2)
7  E=70-10*(Xbios(4)=2)
8  F=25-25*(Xbios(4)=2)
9  H=2
10 Repeat
11   Print At(1,1);
12   Input "NIVEAU ( 2-7 ) :";N
13   Until N=Int(N) And N<8 And N>1
14   U=10
15   P=0
16   N1=Int((U^N-U^(N-1))*Rnd+U^(N-1))
17   N(1)=N1
18   Repeat
19     N(H)=N1
20     A=Int(2*U^(N-1)*Rnd+U^(N-1))
21     Debut:
22     H=Abs(3-H)
23     P=P-(H=1)
24     Clearw 2
25     R$=""
26     Deftext H+1
27     Print At(1,3);"JOUEUR ";H;" - ESSAI ";P
28     Deftext 4
29     Print At(1,7);"NOMBRE :";Tab(18-N);N(H)
30     Deftext 7
31     Print At(12,8);"7654321"
32     Box C,D,C+E,D+F
33     Deftext 1
34     Repeat
35       Print At(1,11);
36       Input "VOTRE JEU ( SOUS FORME NC ) :";R$
37       Until Len(R$)=2 And Val(Right$(R$,1))<=N
38       N(H)=N(H)+Int(Val(Left$(R$,1))*U^(Val(Right$(R$,
1)))-1))
39       N(H)=N(H)-A
40       N(H)=Abs(Int(N(H)+U^N*(N(H)>U^N)))
41       Print At(1,13);"RESULTAT : ";N(H)
42       For I=1 To 5
43         Sound 1,10,2*I,H+4,5
44       Next I
45       Sound 1,0
46       Pause 4
47       R$=Str$(N(H))
48     Until Len(R$)=N
49     For I=1 To Len(R$)-1
50       Exit If Mid$(R$,I,1)<>Mid$(R$,I+1,1)
51     Next I
52     If I<>Len(R$) Then
53       Goto Debut
54     Endif
55     Print
56     Print "BRAVO, JOUEUR ";H
57     For I=1 To 12
58       Sound 1,10,I,4,8
59     Next I
60     Sound 1,0
61     R$=Input$(1)

```

Mastermind

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : logique (un seul joueur).

Difficulté du jeu : difficile.

LE JEU :

Devenu un grand classique des jeux sur micro-ordinateur, ce jeu développe le raisonnement déductif ; il faut réussir à trouver le plus vite possible un nombre de n chiffres (on pourra prendre n compris entre 1 et 5), en disposant à chaque essai de renseignements.

En effet, le ST répondra en indiquant, d'une part, le nombre de chiffres qui sont, dans le nombre proposé, exactement à la même place que dans le nombre à découvrir, et, d'autre part, le nombre de chiffres qui figurent dans le nombre à découvrir sans être exactement au bon emplacement.

LE PROGRAMME :

Le choix du nombre de chiffres du nombre à découvrir est effectué aux lignes 7 à 10.

Le nombre est fabriqué à la ligne 13 (il pourra y avoir des chiffres répétés).

L'affichage est préparé par les lignes 14 à 21.

La ligne 22 s'occupe de la fin de partie.

Une première boucle I (27-37) teste les chiffres qui sont à la bonne place.

Une deuxième boucle J (31-35) teste les chiffres qui sont présents et mal placés.

Les compteurs sont aux lignes 29 et 33.

Les lignes 39 et 41 affichent le résultat, et la ligne 43 comptabilise les essais.

POUR JOUER :

On tape RUN et l'on choisit la longueur du nombre désiré (de 1 à 8), suivie de RET.

Après l'affichage NO. NOMBRE BON PLACE, on introduit son propre essai, suivi de RET.

On obtient immédiatement la réponse (en cas de double chiffre, les résultats peuvent être un peu délicats à interpréter).

On introduit ensuite le nouvel essai, et ainsi de suite.

Lorsque l'on a trouvé, on obtiendra l'affichage TROUVE EN ... COUPS.

On peut sortir du programme à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut chercher à éviter la présence de chiffres doubles dans le nombre à trouver, on peut aussi prévoir un changement de page automatique, avec rappel des résultats précédents ; on pourra aussi réaliser des jeux analogues avec des mots, ou d'autres symboles (couleur, taille...).

```

1  Rem MASTERMIND
2  Fullw 2
3  Clearw 2
4  For J=0 To Xbios(4)
5      Read X,Y,A,B,K,G1,G2,G3,G4
6  Next J
7  Repeat
8      Input "NOMBRE DE CHIFFRES :";N
9      Clearw 2
10 Until N<9
11 E=1
12 F=1
13 X$=Str$(Int(Rnd*9^N+10^(N-1)))
14 Box 0,A,X,B
15 Fill 2,A+1
16 Line K*G2-40,B+1,K*G2-40,Y
17 Line K*G3-40,B+1,K*G3-40,Y
18 Line K*G4-40,B+1,K*G4-40,Y
19 Deftext 1
20 Print At(G1,2);"ESSAI";At(G2,2);"NBRE. ESSAYE"
21 Print At(G3,2);"BON";At(G4,2);"PLACE"
22 While E<>X$
23     Print At(2,F+4);E;At(G2,F+4);"      ";At(G2,F+4
24 );
25     Input E$
26     A=0
27     B=0
28     For I=1 To N
29         If Mid$(E$,I,1)=Mid$(X$,I,1) Then
30             A=A+1
31         Else
32             For J=1 To N
33                 If J<>I And Mid$(E$,J,1)=Mid$(X$,I,1) Then
34                     B=B+1
35             Endif
36         Next J
37     Endif
38 Next I
39 Deftext 3
40 Print At(G3+1,F+4);A
41 Deftext 2
42 Print At(G4+2,F+4);B
43 Deftext 1
44 E=E+1
45 F=-F*(F<>13)+1
46 Wend
47 Print
48 Print "TROUVE EN ";E;" COUPS"
49 For I=1 To 5 Step 2
50     Sound 1,10,I,5,10
51 Next I
52 Sound 1,0
53 R$=Input$(1)
54 Data 304,176,4,24,9,1,10,24,31
55 Data 638,176,4,24,8,6,22,48,62
56 Data 635,344,0,56,8,1,20,48,64

```


Nicomaque

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de réflexion (calcul mental).

Difficulté du jeu : moyenne.

LE JEU :

Il utilise une propriété un peu particulière des entiers de 1 à 100 qui peuvent être identifiés sans ambiguïté par leurs restes dans les divisions par 3, 5 et 7.

Le programme, proposé ici, est un programme codeur qui affiche simplement les restes d'un entier donné n dans les divisions indiquées.

Le jeu consiste à essayer de découvrir l'entier (mais, dans cette version, vous n'avez droit qu'à un seul essai).

LE PROGRAMME :

La ligne 11 permet de confectionner l'entier.

La boucle I (21-31) calcule les trois restes successifs, et la boucle emboîtée J (23-30) permet de les afficher sous une forme assez visuelle, aux bons emplacements.

Les lignes 15 à 20 sont là pour indiquer de quelle division il s'agit.

Le nombre proposé par le joueur est entré à la ligne 34 ; il est affiché et testé aux lignes 35 à 47 (affichage de fin de partie : BRAVO ou PERDU).

Les lignes 51 à 62 s'occupent de gérer la prochaine partie, selon la réponse du joueur.

POUR JOUER :

On lance le programme en tapant RUN.

Les trois restes seront alors affichés sur la même ligne, symbolisés par des étoiles (l'absence d'étoiles signifiant : reste 0).

A la question VOTRE ESSAI, on répondra en donnant un entier compris entre 0 et 100, suivi de RET.

On obtiendra, en réponse, un affichage PERDU ou BRAVO, puis la question la même, une autre, fin (M/A/F).

Pour faire un nouvel essai, on répondra A, ou M suivi de RET.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager de réaliser le programme décodeur (on donne les restes et le ST devra vous donner N) et chercher si la propriété peut être étendue (en effectuant davantage de divisions, par exemple).

```

1  Rem NICOMAUQUE
2  Fullw 2
3  Dim A(4)
4  For I=0 To Xbios(4)
5      Read A(1),A(2),A(3),A,B,C,K
6  Next I
7  Data 60,130,200,40,60,80,4
8  Data 110,240,370,40,80,80,8
9  Data 110,245,380,80,140,160,8
10 X=Random(100)+1
11 Essai:
12 Clearw 2
13 Deftext 7
14 Print At(1,4);"RESTE DANS LA DIVISION PAR"
15 For I=3 To 7 Step 2
16     Print At(K*I-1,7);Chr$(48+I)
17 Next I
18 For I=1 To 3
19     Box A(I),A,A(I)+B,C
20 Next I
21 For I=1 To 3
22     A=X-(2*I+1)*Int(X/(2*I+1))
23     For J=1 To A
24         If A<>0 Then
25             Deftext I+1
26             Print At(J+2*K*I+2,8);"o"
27             Sound 1,10,4,5,4
28         Endif
29         Sound 1,0
30     Next J
31 Next I
32 Deftext 2
33 Print At(1,13);
34 Input "VOTRE ESSAI : ";N
35 If N=X Then
36     Print
37     Print "BRAVO"
38     For I=1 To 5
39         Sound 1,10,I,5,6
40     Next I
41 Else
42     For I=1 To 3
43         Sound 1,10,2,3,8
44     Next I
45     Print
46     Print "PERDU"
47 Endif
48 Deftext 1
49 Sound 1,0
50 Print
51 Print "LA MEME, UNE AUTRE, FIN ( M/A/F ) :";
52 R$=Upper$(Input$(1))
53 If R$="A" Then
54     Run
55 Else
56     If R$="M" Then
57         Goto Essai
58     Endif
59 Endif
60 Print
61 Print "IL FALLAIT TROUVER .";X
62 R$=Input$(1)

```

Le pendu

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de lettres (version deux joueurs).

Difficulté du jeu : moyenne.

LE JEU :

Il serait sans doute assez difficile de retrouver le nom de l'inventeur de ce jeu très à la mode, mais il est certain qu'il existe depuis des temps immémoriaux, quelquefois pratiqué sous des formes assez différentes (par exemple, reconstituer un petit cochonnet en lançant des dés).

L'apparition des jeux sur micro-ordinateur et la magie des écrans de télévision lui ont fait connaître un renouveau certain : dans la version retenue, l'un des joueurs doit entrer un mot de longueur quelconque. L'ordinateur affiche alors des tirets à l'emplacement des lettres de ce mot et demande au deuxième joueur de proposer une lettre. Chaque réponse correcte fait apparaître la lettre à son emplacement exact dans le mot, et chaque réponse fausse fait progresser le dessin de la potence.

Pour gagner, il suffit de réussir à trouver le mot inconnu avant d'être pendu.

LE PROGRAMME :

Le mot à deviner est introduit en début de programme (lignes 8 à 18) grâce à une boucle d'entrée lettre par lettre.

La ligne 13 fabrique un mot P\$ qui contient des tirets à l'emplacement de chacune des lettres du mot initial.

La boucle I (26-40) s'occupe du dessin (plus ou moins complet, selon la valeur de P).

Le dessin est effectué par lecture des DATA des lignes 73 à 88.

Les lignes 61 à 67 affichent le résultat et arrêtent le jeu.

La lettre essayée est entrée en ligne 44 et testée dans une boucle I (48-55).

La ligne 51 remplace chaque tiret par la lettre correspondante.

POUR JOUER :

Après le lancement du programme par RUN, l'un des joueurs doit entrer son mot et taper RET. Le mot tapé reste invisible, et on peut l'effacer en tapant Ø en cas d'erreur de frappe. Des tirets sont affichés à l'emplacement de chacune des lettres du mot ; et la question VOTRE LETTRE apparaît sur l'écran ; l'autre joueur répond alors en proposant une lettre.

Si la lettre est bonne, elle viendra remplacer un ou plusieurs tirets ; sinon, la potence va commencer à se dessiner.

Le mot exact sera affiché dans le cas où l'on aurait perdu.

On devra taper la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

Il sera intéressant de constituer une liste de mots dans lesquels le ST pourra piocher au hasard ; on pourra aussi améliorer le dessin (en redéfinissant les caractères, par exemple) et modifier le nombre d'essais permis selon la difficulté du mot.

```

1  Rem LE PENDU
2  Fullw 2
3  Clearw 2
4  Dim X(10),Y(10)
5  R$=""
6  T$=""
7  P$=""
8  Do
9      Print At(1,1);"MOT :";T$;
10     S$=Upper$(Input$(1))
11     Exit If S$=Chr$(13)
12     R$=R$+S$
13     P$=P$+"-"
14     T$=T$+"#"
15     If S$="" Then
16         Run
17     Endif
18 Loop
19 F=40*(Xbios(4)<>2)
20 P=10
21 Clearw 2
22 Do
23     Deftext 2
24     Print At(5,2);P$
25     Restore
26     For I=10 To P Step -1
27         Read N
28         Deffill Random(4^(2-Xbi
os(4)))+1,2,8
29         For J=1 To N
30             Read A,B,C,D
31             X(J-1)=A+110
32             Y(J-1)=B+F
33             X(J)=C+110
34             Y(J)=D+F
35             Line A+110,B+F,C+110,
D+F
36         Next J
37         If I>3 Then
38             Polyfill J-1,X(),Y()
39         Endif
40     Next I
41     Exit If P$=R$ Or P=1
42     Deftext 1
43     Print At(1,17);Spc(30);At
(1,17);"VOTRE LETTRE :";
44     S$=Upper$(Input$(1))
45     Deftext 4
46     Print S$
47     T=0
48     For I=1 To Len(R$)
49         If Mid$(R$,I,1)=S$ Then
50             T=1
51             Mid$(P$,I,1)=S$
52             Sound 1,10,I,5,8
53             Sound 1,0
54         Endif
55     Next I
56     If T=0 Then
57         P=P-1
58     Endif
59 Loop
60 Print
61 If P$=R$ Then
62     Deftext 1
63     Print "BRAVO"
64 Else
65     Deftext 7
66     Print "PERDU. LE MOT ETAI
T :";R$
67 Endif
68 For I=1 To 5 Step 2
69     Sound 1,10,I,4,8
70 Next I
71 Sound 1,0
72 R$=Input$(1)
73 Data 4,50,60,50,50,50,50,15
0,50,150,50,150,60,150,60,5
0,60
74 Data 4,95,80,95,60,95,60,10
5,60,105,60,105,80,105,80,9
5,80
75 Data 4,80,90,95,80,95,80,10
5,80,105,80,120,90,120,90,8
0,90
76 Data 6,90,90,80,100,80,100,
90,110,90,110,110,110,110,1
10,120,100,120,100
77 Data 110,90,110,90,90,90
78 Data 4,95,120,95,110,95,110
,105,110,105,110,105,120,10
5,120,95,120
79 Data 10,50,135,60,120,60,12
0,140,120,140,120,150,135,1
50,135,140,135
80 Data 140,135,130,125,130,12
5,130,135,130,135,70,135,70
,135,70,125,70,125
81 Data 60,135,60,135,50,135
82 Data 4,70,135,130,135,130,1
35,130,155,130,155,70,155,7
0,155,70,135
83 Data 7,70,155,70,165,70,165
,90,165,90,165,90,155,90,15
5,110,155
84 Data 110,155,110,165,110,16
5,130,165,130,165,130,155
85 Data 6,70,165,70,185,70,185
,90,185,90,185,90,165,110,1
65,110,185
86 Data 110,185,130,185,130,18
5,130,165
87 Data 8,50,185,50,200,50,200
,90,200,90,200,90,185,90,18
5,50,185
88 Data 110,185,110,200,110,20
0,150,200,150,200,150,185,1
50,185,110,185

```

Accrochez les wagons

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : solitaire.

Difficulté du jeu : facile.

LE JEU :

Il faut essayer d'accrocher, le plus rapidement possible, la locomotive à son wagon, qui apparaissent tous deux en début de partie, en deux endroits plus ou moins éloignés de l'écran. On devra pour cela déplacer la locomotive en indiquant la suite des mouvements à effectuer (sous forme codée : H pour une case vers le haut ; B pour une case vers le bas ; D pour une case vers la droite et G pour une case vers la gauche).

On tapera, par exemple : BBGDHHH.

LE PROGRAMME :

Les lignes 11 à 20 lisent les données permettant le dessin de la locomotive et du wagon.

La position initiale de la locomotive est définie par les lignes 22 et 23, celle du wagon par les lignes 25 et 26.

La ligne 27 évite qu'ils soient initialement trop proches l'un de l'autre.

Les lignes 30 à 39 réalisent l'affichage du wagon et de la locomotive.

Le test de fin est en ligne 28.

Les mouvements à effectuer sont entrés dans une variable alphanumérique R\$, en ligne 41, et sont décodés dans une boucle I (46-51) pour être exécutés aux lignes 52 et 53.

POUR JOUER :

RUN permet d'afficher les positions de départ de la locomotive et du wagon.

On tape alors la série des déplacements à effectuer, sous la forme DDDHHHBHGG, suivie de RET.

La locomotive sera alors affichée dans sa nouvelle position, et l'on recommence, jusqu'à ce qu'elle soit accrochée au wagon.

On obtiendra alors l'affichage du score.

On tapera la combinaison CSA pour sortir du programme.

EXTENSIONS POSSIBLES :

Il sera intéressant de prévoir la perte de la partie si, par accident, au cours du déplacement, la locomotive vient percuter le wagon.

On pourra aussi prévoir plusieurs wagons et un ordre d'accrochage, jouer à plusieurs avec plusieurs locomotives, dessiner les voies ferrées, introduire des obstacles à contourner obligatoirement d'un certain côté, etc.

```

1  Rem ACCROCHEZ LES WAGONS
2  K=1-(Xbios(4)<2)
3  Deftext 2,0,0,13+7*(Xbios(4)
4  )<2)
5  U=50+20*(Xbios(4)<2)
6  V=120+60*(Xbios(4)=0)
7  W=56+28*(Xbios(4)<2)
8  H=Random(4^(2-Xbios(4)))
9  G=Random(4^(2-Xbios(4)))
10 Fullw 2
11 Dim L(20,2),W(20,2)
12 For I=1 To 20
13   Read A,B
14   L(I,1)=A/K
15   L(I,2)=B/K
16 Next I
17 For I=1 To 19
18   Read A,B
19   W(I,1)=A/K
20   W(I,2)=B/K
21 Next I
22 N=0
23 Ly=4*Random(U)+U
24 Lx=4*Random(V)+U
25 Repeat
26   Wy=4*Random(U)+U
27   Wx=4*Random(V)+U
28 Until Abs(Lx-Wx)>=U+15
29 While Not (Lx=Wx+W And Ly=W
30 y)
31   Clearw 2
32   For I=1 To 19
33     Line L(I,1)+Lx,Ly-L(I,2
34 ),L(I+1,1)+Lx,Ly-L(I+1,2)
35   Next I
36   For I=1 To 18
37     Line W(I,1)+Wx,Wy-W(I,2
38 ),W(I+1,1)+Wx,Wy-W(I+1,2)
39   Next I
40   Deffill H,2,7
41   Fill Lx+3,Ly-12
42   Deffill G,2,14
43   Fill Wx+3,Wy-7+5*(Xbios(4
44 )=2)
45   Print At(1,20);
46   Input "CHEMIN (DGGHBB...
47 ):";R$
48   R%=Upper$(R$)
49   X=0
50   Y=0
51   N=N+1
52   For I=1 To Len(R$)
53     X=X-4*((Mid$(R$,I,1)="D
54 " And Lx<5*V)-(Mid$(R$,I,1)
55 ="G" And Lx>U/2))
56     Y=Y-4*((Mid$(R$,I,1)="B
57 " And Ly<5*U)-(Mid$(R$,I,1)
58 ="H" And Ly>V/2))
59     Sound 1,10,Rnd*13,5,2
60     Sound 1,0
61   Next I
62   Lx=Lx+X
63   Ly=Ly+Y
64 Wend
65 Print At(1,18);"BRAVD ; TRD
66 UVE EN ";N;" COUPS";
67 For I=1 To 12
68   Sound 1,10,I,4,10
69 Next I
70 Sound 1,0
71 R%=Input$(1)
72 Data 0,10,0,30,10,30,10,27,
73 30,27,30,40,40,40,40,27,50,
74 27
75 Data 50,17,55,10,45,10,42,0
76 ,38,0,35,10,25,10,22,0,18,0
77 ,15,10,0,10
78 Data 0,10,0,30,5,30,5,15,45
79 ,15,45,30,50,30,50,12,55,12
80 Data 55,10,45,10,42,0,38,0,
81 35,10,25,10,22,0,18,0,15,10
82 ,0,10

```

Le loup et les agneaux

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : moyenne.

LE JEU :

C'est un jeu assez connu, qui peut être pratiqué indifféremment sur un damier 8 sur 8, ou 10 sur 10. Il est présenté, ici, dans une version pour un damier de 8 cases ; l'un des joueurs choisit le loup qui se déplace suivant les diagonales, en restant sur les cases noires, d'une case à chaque coup, et qui peut avancer ou reculer. Les agneaux se déplacent également sur les cases noires, en diagonale, mais n'ont pas le droit de reculer. Les cases déjà occupées sont interdites (pas de prises), et il n'y a pas de saut.

Le loup gagne s'il réussit à arriver sur la ligne du bas avant que tous les agneaux n'occupent celle du haut (les agneaux gagnent dans ce dernier cas) ou s'ils arrivent à bloquer le loup.

LE PROGRAMME :

Une variable A\$, en ligne 12 contient les positions de départ.

La boucle I (16-21) dessine le damier, et les lignes 24 à 29 placent le loup et les agneaux.

La ligne 30 gère le changement de joueur.

La ligne 32 affiche JEU DU LOUP si c'est le joueur qui manipule le loup qui a le trait, et JEU DES AGNEAUX dans le cas contraire.

Les lignes 35 à 42 analysent la réponse, en fonction du joueur concerné.

Les lignes 45 à 47 permettent l'effacement.

POUR JOUER :

RUN déclenche l'arrêt sur l'attente d'une entrée : Ø et RET si l'on veut que les agneaux commencent, et 1 pour le loup. Après quoi, le damier s'affiche et la phrase JEU DU LOUP (Q, S, W, X) pour le loup et JEU DES AGNEAUX (ND et NG) pour les agneaux.

Les agneaux répondent par un numéro, suivi de D ou G (pour droite ou gauche).

Ainsi, 2D provoque le déplacement de l'agneau numéro 2 qui monte en diagonale d'une case sur la droite. Le déplacement effectué est affiché, puis on obtient une question identique pour le loup qui répondra en tapant Q, S, W ou X, suivant la direction choisie :



On devra veiller à ne pas sortir de l'échiquier et à ne pas venir sur l'adversaire, aucun contrôle n'ayant été prévu.

EXTENSIONS POSSIBLES :

On pourra prévoir un repérage des cases, un graphisme plus soigné, des contrôles de non-venue sur une case déjà occupée...

On pourra aussi envisager de jouer sur un damier de 10 cases sur 10 cases, ce qui augmente légèrement la difficulté du jeu pour les agneaux. Mais l'extension la plus intéressante à réaliser sera certainement la réalisation d'un programme permettant au ST de se comporter en adversaire honorable, soit avec le loup, soit avec les agneaux (et, dans ce cas, il sera possible de définir une stratégie gagnante pour le ST).

```
1  Rem LE LOUP ET LES AGNEAUX
2  P=Xbios(4)
3  Fullw 2
4  Openw 2
5  Clearw 2
6  Dim Tex$(2)
7  Tex$(1)="JEU DES AGNEAUX (N
  D OU NG) : "
8  Tex$(0)="JEU DU LOUP (Q,S,W
  ,X) : "
9  For I=0 To P
10   Read K1,K2,K3
11 Next I
12 A$="8183858714"
13 Deftext 1
14 Input "DEBUT 1:LOUP 0:AGNEA
  UX : ";S
15 Clearw 2
16 For I=1 To 8
17   Print At(1,2*I+1);Chr$(48
  +I);At(K3*I,1);Chr$(64+I)
18   For J=1 To 8
19     Gosub Rectangle
20   Next J
21 Next I
22 Do
23   Deftext 7
24   Print At(K3*Val(Mid$(A$,1
  0,1))+(P>0),2*Val(Mid$(A$,9
  ,1))+1);"L"
25   Deftext 9
26   Print At(K3*Val(Mid$(A$,2
  ,1))+(P>0),2*Val(Mid$(A$,1,
  1))+1);"1"
27   Print At(K3*Val(Mid$(A$,4
  ,1))+(P>0),2*Val(Mid$(A$,3,
  1))+1);"2"
28   Print At(K3*Val(Mid$(A$,6
  ,1))+(P>0),2*Val(Mid$(A$,5,
  1))+1);"3"
29   Print At(K3*Val(Mid$(A$,8
  ,1))+(P>0),2*Val(Mid$(A$,7,
  1))+1);"4"
30   S=Abs(S-1)
31   Deftext 1
32   Print At(1,19);Spc(37);At
  (1,19);Tex$(S);
33   Input R$
34   R$=Upper$(R$)
35   If S=0 Then
36     C=10
37     A=Val(Right$(A$,2))+(C+
  1 And R$="X")+(C-1 And R$="
  W")
38     A=A-(C+1 And R$="Q")-(C
  -1 And R$="S")
39   Else
40     C=2*Val(Mid$(R$,1,1))
41     A=Val(Mid$(A$,C-1,2))-(
  9 And Right$(R$,1)="D")-(11
  And Right$(R$,1)="G")
42   Endif
43   B$=Mid$(A$,C-1,2)
44   A$=Left$(A$,C-2)+Right$(S
  tr$(A$,2)+Right$(A$,Len(A$)
  -C)
45   I=Val(Left$(B$,1))
46   J=Val(Right$(B$,1))
47   Gosub Rectangle
48 Loop
49 Procedure Rectangle
50   Deffill 1-(P<2)-2*(P=0)*((
  I+J) Mod 2),2,7*((I+J) Mod
  2)+1
51   Pbox K1*J-K1+18,K2*I-K2+2
  2+10*(P<2),K1*J+18,K2*I+22+
  10*(P<2)
52 Return
53 Data 32,16,4,64,16,8,64,32,
  8
```


Le lézard d'Alice

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : solitaire.

Difficulté du jeu : assez difficile.

LE JEU :

Il s'inspire très vaguement d'une idée de Lewis Carroll, dans *Alice au Pays des Merveilles*. Dans l'histoire originale, Alice grandissait ou diminuait, alors que dans le jeu, pour des raisons d'encombrement de la mémoire, c'est un lézard qui change de taille.

Deux verres apparaissent sur l'écran : l'un contient le liquide qui fait grandir, l'autre le liquide qui fait rapetisser (malheureusement, ce n'est pas toujours le même verre). Le joueur devra indiquer la quantité de liquide qu'il désire boire et le lézard changera de taille en fonction de cette quantité. Le but du jeu est de parvenir à lui rendre sa taille initiale.

LE PROGRAMME :

Les lignes 10 à 16 dessinent les verres.

Les deux boucles emboîtées I (42-48) et J (43-45) les remplissent avec la quantité indiquée en ligne 31.

La variation de taille du lézard est définie par la ligne 58, avec contrôle par les lignes 59 à 61.

La boucle I (62-64) dessine le corps du lézard, les lignes 65 à 74 dessinent l'œil et les pattes.

Un message gagnant sera affiché par la ligne 82 si l'on a trouvé la bonne taille.

POUR JOUER :

On lance le programme par RUN : les deux verres se dessinent, puis apparaît la question QUEL COTE à laquelle on répond en tapant G ou D ; on obtient l'affichage QUELLE QUANTITE ; on entre alors une quantité (de 0 à 130) et on frappe RET.

Le verre désigné se remplit sous vos yeux, puis, après quelques instants, le lézard est affiché. On devra alors frapper une touche quelconque ; si le lézard n'avait pas la bonne taille, aucun message n'est affiché et l'on revient au dessin des verres.

On devra taper la combinaison CSA pour arrêter le programme.

EXTENSIONS POSSIBLES :

Puisqu'il s'agit d'un jeu essentiellement graphique, il sera intéressant d'améliorer les dessins et la couleur, de modifier le caractère aléatoire du choix des verres, etc.

Pourquoi ne pas relire *Alice au Pays des Merveilles*, les idées n'y manquent pas.

```

1  Rem LE LEZARD D'ALICE
2  P=Xbios(4)
3  K=2+(P=0)
4  Deftext 2,0,0,13+7*(P<2)
5  Repeat
6    Fullw 2
7    Clearw 2
8    Deffill Random(4^(2-P))+1
9    ,2,7
10   N=18
11   Pbox 24,80,32,140
12   Pbox 24+50*K,80,32+50*K,1
13   40
14   Pbox 64+50*K,80,72+50*K,1
15   40
16   Pbox 64+100*K,80,72+100*K
17   ,140
18   Pbox 24,140,32+50*K,148
19   Pbox 64+50*K,140,72+100*K
20   ,148
21   Print At(5-3*(P>0),20+10*
22   (P=2));"GAUCHE";At(16-9*(P>
23   0),20+10*(P=2));"DROIT";
24   Repeat
25     Print At(1,1);Spc(37);A
26     t(1,1);"QUEL COTE :";
27     Sound 1,10,4,5,8
28     Sound 1,0
29     A$=Upper$(Input$(1))
30     Until A$="D" Or A$="G"
31     U=8-(90-50*(P>0))*(A$="D"
32     )
33   Deftext 4,2,4
34   If A$="D" Then
35     Print "DROIT"
36   Else
37     Print "GAUCHE"
38   Endif
39   Deftext 1,0
40   Print At(1,3);Spc(37);At(
41   1,3);"QUELLE QUANTITE :";
42   A$=""
43   Deftext 3
44   Do
45     R$=Input$(1)
46     Exit If Asc(R$)=13
47     A$=A$+R$
48   Print R$;
49   Loop
50   Color 1-2*(P<2)-(P=0)
51   Q=Val(A$)
52   For I=1 To 4*Sqr(Q)
53     For J=1 To 41-50*(P>0)
54       Plot U+J+24,140-I
55     Next J
56     Sound 1,10,7,5,2
57     Sound 1,0
58   Next I
59   For I=1 To 6
60     Sound 1,10,2*1,5,5
61   Next I
62   Sound 1,0
63   R$=Input$(1)
64   Setcolor 0,Random(1912)
65   Color 2
66   Clearw 2
67   H=-2*(Rnd<0.5)-1
68   N=N+H*Int(Sqr(Q))
69   If N<12 Or N>60 Then
70     N=-12*(N<12)-60*(N>60)
71   Endif
72   For I=0 To 8*(N+3)
73     Draw I,100 To I,97
74   Next I
75   T=8*(N-Int((Sqr(N)/2)))
76   Draw T,88 To T-2,90 To T+
77   2,88
78   T=8*Sqr(N)
79   Draw T,102 To T-2,104
80   T=T+2
81   Draw T,102 To T+2,104
82   T=8*(N-Sqr(N))
83   Draw T,102 To T-2,104
84   T=T+2
85   Draw T,102 To T+2,104
86   For I=1 To 12 Step 2
87     Sound 1,10,1,5,7
88   Next I
89   Sound 1,0
90   A$=Input$(1)
91   Setcolor 0,1911
92   Until N=28
93   Print At(1,8);"BRAVO....."

```

Reines sur l'échiquier

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion.

Difficulté du jeu : difficile.

LE JEU :

Le programme utilise un algorithme très performant pour résoudre un problème qui a donné du fil à retordre à des chercheurs, même aussi célèbres que le mathématicien Gauss, qui n'avait pas pu découvrir toutes les solutions. Il s'agit de trouver toutes les manières de disposer des reines sur un échiquier, de telle sorte qu'aucune ne puisse en prendre une autre. Le programme proposé est très souple, puisque vous disposez même de la possibilité de choisir la taille de votre échiquier, et les résultats seront affichés à la fois sous forme visuelle et sous forme numérique.

LE PROGRAMME :

La ligne 9 vous permet de choisir la taille de l'échiquier.

Une boucle d'exploration principale I (12-26) contient une sous-boucle J (16-22) qui réalise le contrôle de non-alignement (ligne 18).

Si, au cours de l'exploration, on rencontre un alignement, on revient en arrière, en utilisant les lignes 30 à 38 et on recommence.

La boucle F (43-54) assure l'impression de l'échiquier, et la ligne 49, que l'on pourra éventuellement supprimer, assure l'impression du résultat sous forme d'un code numérique correspondant à la position des pièces dans chaque colonne.

La ligne 57 permet un arrêt après l'affichage de chacune des solutions.

POUR JOUER :

On lance le programme avec RUN, puis on tape la dimension de l'échiquier désirée, et on fait RET.

Il suffit d'attendre, avec patience (plus l'échiquier sera grand, plus l'attente sera longue). La première solution sera affichée puis le programme s'arrête, et il faut appuyer sur une touche quelconque pour le faire repartir. Pour terminer l'exploration sans avoir eu toutes les solutions, il faudra taper la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager de trouver des algorithmes pour tous les problèmes analogues (position de tours, déplacement d'un cavalier, etc.).

```

1  Rem REINES SUR L'ECHIQUE
2  Fullw 2
3  I=1
4  For I=0 To Xbios(4)
5      Read A,B,C,D
6  Next I
7  Repeat
8      Clearw 2
9      Input "TAILLE DE L'ECHIQUEUR :";S
10 Until S<9
11 Dim D(S)
12 I=1
13 Debut:
14 D(I)=1
15 Suite:
16 J=1
17 Suivant:
18 If I<>J And (D(I)=D(J) Or (Abs(D(I)-D(J))=I-J)) Then
19     Goto Avance
20 Endif
21 Inc J
22 If J<I Then
23     Goto Suivant
24 Endif
25 Inc I
26 If I<=S Then
27     Goto Debut
28 Endif
29 Gosub Affiche
30 Dec I
31 Avance:
32 Inc D(I)
33 If D(I)<=S Then
34     Goto Suite
35 Endif
36 Dec I
37 If I<>0 Then
38     Goto Avance
39 Endif
40 Data 16,32,16,1,16,64,16,2,32,64,32,2
41 Procedure Affiche
42     Clearw 2
43     For F=1 To S
44         For G=1 To S
45             Deffill 1,Even(F+G)+1,2
46             Pbox B*G-4*D,C*F-4,B+B*G-4*D,A+C*F-4
47         Next G
48         Deftext 3
49         Print At(D*4*F+2,1);F;At(1,2*F+1);Chr$(48+F)
50         Deftext 2
51         Print At(D*4*D(F)+2,2*F+1);""
52         Deftext 4
53         Print At(F+1,20);Right$(Str$(D(F)),1);
54     Next F
55     Sound 1,10,4,5,12
56     Sound 1,0
57     R$=Input$(1)
58 Return

```

Cube

REFERENCES :

Difficulté de programmation : assez difficile.

Catégorie de jeu : dessin.

Difficulté du jeu : moyenne.

LE JEU :

Un cube est affiché en 3D, ainsi que 3 axes de coordonnées ox , oy et oz .

Il s'agit d'effectuer des rotations, d'un angle quelconque exprimé en degrés, autour de l'un des 3 axes.

Le cube tourne, puis on peut essayer une nouvelle rotation.

LE PROGRAMME :

La ligne 5 prépare 3 variables indicées X , Y et Z , pour recevoir les coordonnées des sommets.

La boucle I (6-11) lit les DATA (lignes 70 et 71).

Les lignes 12 à 15 calculent la position initiale.

La boucle WHILE-WEND (lignes 16 à 39) permet d'opérer tant que l'angle de rotation n'est pas égal à 0.

Les lignes 18 à 20 dessinent les axes.

La boucle I (23-29) dessine le cube.

Les entrées sont faites aux lignes 31 à 36, et gérées par les lignes 37 et 38, qui appellent les sous-programmes correspondants.

POUR JOUER :

On lance l'exécution par RUN. Un premier cube est affiché. On tape X , Y ou Z , selon l'axe de rotation choisi, puis on introduit un angle en degrés.

La rotation est alors effectuée.

On arrête en tapant 0 à la question « ANGLE ».

EXTENSIONS POSSIBLES :

On pourra imaginer de colorer les faces, et faire disparaître les faces cachées.

On peut travailler avec d'autres objets, en modifiant les DATAs, et faire un véritable programme 3D en modifiant la position de l'œil de l'observateur (changement d'échelle).

```

1  Rem CUBE
2  Fullw 2
3  P=Abs(Xbios(4)-2)
4  K=1-(P=0)
5  Dim X(20),Y(20),Z(20)
6  For I=0 To 15
7    Read A,B,C
8    X(I)=A-100
9    Y(I)=B-100
10   Z(I)=C-50
11 Next I
12 Gosub Angle(30)
13 Gosub Rotz
14 Gosub Angle(10)
15 Gosub Roty
16 While A$<>"0"
17   Clearw 2
18   Draw 10,K#170 To 40,K#120
19   To 120,K#120 To 40,K#120 T
20   o 40,K#80
21   Print At(5,10);"Z";At(5,1
22   5);"0"
23   Print At(14,15);"Y";At(1,
24   19);"X"
25   X=400+150*(P=2)+Y(0)
26   Y=100-Z(0)/2
27   For I=1 To 15
28     U=400+150*(P=2)+Y(I)
29     V=100-Z(I)/2
30     Line X,K#Y,U,K#V
31     X=U
32     Y=V
33   Next I
34   Repeat
35     Print At(6,20);"Rotatio
36     n autour des x,y,z :";
37     R$=Upper$(Input$(1))
38     Until Instr(1,"XYZ",R$)<>
39     0
40     Print R$
41     Print At(6,21);"Quel angl
42     e :";
43
44   Input A$
45   Gosub Angle(Val(A$))
46   On Asc(R$)-87 Gosub Rotx,
47   Roty,Rotz
48 Wend
49 Print "C'EST FINI..."
50 Procedure Angle(A)
51   A=A*Pi/180
52   C=Cos(A)
53   S=Sin(A)
54   Return
55 Procedure Rotx
56   For I=0 To 15
57     B=Y(I)
58     D=Z(I)
59     Y(I)=B*C-D*S
60     Z(I)=B*S+D*C
61   Next I
62 Return
63 Procedure Roty
64   For I=0 To 15
65     A=X(I)
66     D=Z(I)
67     Z(I)=D*C-A*S
68     X(I)=D*S+A*C
69   Next I
70 Return
71 Procedure Rotz
72   For I=0 To 15
73     A=X(I)
74     B=Y(I)
75     X(I)=A*C-B*S
76     Y(I)=A*S+B*C
77   Next I
78 Return
79 Data 0,100,100,0,0,100,100,
80 0,100,100,100,100,0,100,100
81 ,0,100,0,100,100,0,100
82 Data 0,0,0,0,0,0,100,0,100,
83 100,0,100,100,100,100,0,100
84 ,100,0,0,0,0,0,0,100

```

Le jeu du missile

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : contre le ST.

Difficulté du jeu : moyenne.

LE JEU :

Le ST est en alerte : vous êtes aux commandes d'un avion chargé d'une mission d'espionnage sur son territoire ; votre but est de survoler une base ennemie (représentée par un pavé agrandi), et de revenir sain et sauf sur votre base ; mais, dès que le ST vous aura repéré, il va envoyer à votre poursuite un missile très perfectionné qui n'aura de cesse de vous détruire. Votre avion peut se déplacer dans toutes les directions, grâce aux touches de 1 à 9 ; réussirez-vous à remplir votre mission ?

LE PROGRAMME :

La ligne 9 dessine la base ennemie à survoler (sous forme d'un pavé agrandi).

La ligne 11 dessine votre base de départ.

Les lignes 14 et 25 choisissent l'instant de départ du missile.

La ligne 20 dessine votre avion.

Les lignes 22 et 23 définissent son mouvement suivant 8 directions, en utilisant les sous-programmes en lignes 56, 60, 63 et 67.

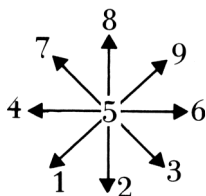
La ligne 28 assure le départ du missile qui est dessiné par la ligne 240 et dont le déplacement est géré par les lignes 29 et 30.

Les lignes 31 à 45 contrôlent la fin de partie.

POUR JOUER :

Lancer le programme par RUN.

Le terrain de jeu s'affiche et vous pouvez déplacer votre appareil en utilisant les touches de 1 à 9, en respectant leur emplacement sur le pavé numérique.



Vous devez lui faire contourner le territoire ennemi (si vous, ou le missile, touchez ce territoire, une deuxième partie commence).

Si le missile vous détruit, vous obtenez un message TOUCHE, et la partie s'arrête.

Taper la combinaison CSA pour arrêter la partie.

EXTENSIONS POSSIBLES :

Il sera sans doute nécessaire d'améliorer les divers contrôles du programme (l'avion a-t-il survolé la base ennemie, est-il revenu à sa propre base...). On peut aussi prévoir divers niveaux de jeu en faisant, par exemple, varier la vitesse du missile, prévoir l'utilisation de leurres par l'avion, l'explosion du missile au bout d'un certain temps, etc.

```
1  Rem LE JEU DU MISSILE
2  P=Xbios(4)
3  Openw 2
4  Debut:
5  Clearw 2
6  Deffill 1,2,4
7  Fill 2,2
8  Deftext 2*Abs(P-2)+1,17,0,3
   2+9*(P=0)
9  Print At(20+9*(P=0),2);Chr$(
   190)
10 Deftext 1,1,0,13+7*(P<2)
11 Print At(1,17);Chr$(35)+Chr
   $(35)+Chr$(35)
12 X=16
13 Y=1
14 S=Random(15)+6
15 U=1
16 Bouge:
17 S=S-1
18 V=72+36*(P=0)
19 Boucle:
20 Print At(Y,X);Chr$(127)
21 A$=Chr$(Inp(2))
22 H=-2*(Val(A$)<5)-1
23 G=Val(A$)+2*(H-1)+2*(A$>"6"
   Or A$="3")-2*(A$="1" Or A$
   ="6")
24 On G Gosub Pr1,Pr2,Pr3,Pr4
25 If S<>0 Then
26   Goto Bouge
27 Endif
28 Print At(V,U);Chr$(174)
29 U=U-Sgn(U-X)
30 V=V-Sgn(V-Y)
31 If U=X And V=Y Then
32   Print "TOUCHE..."
33   Ga=0
34   Goto Fin
35 Endif
36 If (U=5 Or U=6) And (V=40+2
   0*(P=0)) Then
37   Goto Debut
38 Endif
39 If (X<=4 And Y>=40+20*(P=0)
   ) Then
40   M=1
41 Endif
42 If M=1 And X>=17 And Y<=3 T
   hen
43   Ga=1
44   Goto Fin
45 Endif
46 Goto Boucle
47 Fin:
48 For I=1 To 12
49   Sound 1,10,1,5,10
50 Next I
51 Sound 1,0
52 If Ga=1 Then
53   Print "MISSION ACCOMPLIE"
54 Endif
55 R$=Input$(1)
56 Procedure Pr1
57   Gosub Pr2
58   Y=Y+H
59 Return
60 Procedure Pr2
61   X=X+H
62 Return
63 Procedure Pr3
64   Gosub Pr2
65   Gosub Pr4
66 Return
67 Procedure Pr4
68   Y=Y-H
69 Return
```


Chasse au sous-marin

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de réflexion.

Difficulté du jeu : moyenne.

LE JEU :

Vous êtes à bord d'un bâtiment de surface, chargé de traquer un sous-marin caché quelque part dans votre secteur ; vous devez vous déplacer jusqu'à venir au-dessus du sous-marin et tirer à une certaine profondeur ; mais, attention : le sous-marin, lui aussi, est mobile ; il ne coupera ses moteurs que lorsque vous serez exactement à sa verticale, pour ne pas être repéré.

LE PROGRAMME :

La boucle I (12-20) affiche le terrain, en utilisant une sous-boucle J (17-19).

Les lignes 25 et 26 définissent la position initiale du sous-marin ; les lignes 28 et 29 permettent son déplacement en le contrôlant (on reste dans les limites du terrain = ligne 30).

La ligne 33 affiche votre position, grâce à un rectangle colorié.

Les lignes 37 et 38 contrôlent votre déplacement.

La distance est calculée en ligne 39, affichée en ligne 42 et testée en lignes 45 et 48.

Le test de fin de partie est effectué à la ligne 52.

POUR JOUER :

On lance le programme par RUN et on entre la dimension ; la grille de jeu s'affiche. Vous êtes au départ en position (0, 0) et vous vous déplacez avec les flèches.

Après chaque déplacement, la nouvelle distance du sous-marin s'affiche ; si l'on est trop près, le sous-marin se déplace. Si vous réussissez à vous placer à sa verticale, la distance 0 est affichée et il ne bouge plus.

Vous tirez en donnant une profondeur entière (entre 2 et 4, bornes comprises). Si vous le ratez, il se replace n'importe où, et on recommence ; si vous le coulez, vous obtenez l'affichage BRAVO et la partie s'arrête.

On peut arrêter à tout moment le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il est possible de prévoir une amélioration du graphisme, d'améliorer les possibilités de déplacement du chasseur (un peu lent, dans cette version), de prévoir des parcours plus sophistiqués pour le sous-marin, de mettre un compteur de temps...

```

1  Rem CHASSE AU SOUS-MARIN
2  Fullw 2
3  Openw 2
4  K1=52+24*(Xbios(4)<2)
5  K2=34+18*(Xbios(4)<2)
6  Deftext 4,0,0,13+7*(Xbios(4)
   )<2)
7  Repeat
8    Clearw 2
9    Input "DIMENSION DE LA GR
   ILLE :";H
10   Clearw 2
11   Until H<11
12   For I=1 To H
13     Print At(1,2*I+1);Chr$(47
   +I);
14     Deftext 2,0,0,13+8*(Xbios
   (4)<2)
15     Print At((6+2*(Xbios(4)<2
   ))*I+1,1);Chr$(64+I)
16     Deftext 4,0,0,13+7*(Xbios
   (4)<2)
17     For J=1 To H
18       Box K1*J-K1+14,K2*I-4,K
   1*J+14,K2*I+K2-4
19     Next J
20   Next I
21   P=1
22   X=1
23   Y=1
24   Do
25     V=Random(H)
26     U=Random(H)
27     Avance:
28     U=U+P
29     V=V+P
30     P=P+2*P*(U<0 Or U>H Or V<
   0 Or V>H)
31     Repeat
32       Deffill 1,2,8
33       Pbox K1*Y-K1+15,K2*X-3,
   K1*Y+13,K2*X+K2-5
34       F=Inp(2)
35       Deffill 0,0,0
36       Pbox K1*Y-K1+15,K2*X-3,
   K1*Y+13,K2*X+K2-5
37       Y=Y-(F=205 And Y<H)+(F=
   203 And Y>1)
38       X=X-(F=208 And X<H)+(F=
   200 And X>1)
39       D=Int(Sqr(Abs(X-V)^2+Ab
   s(Y-U)^2))
40       F=34-24*(Xbios(4)=2)
41       G=7+6*(H>8)
42       Print At(F,G);"DI. ";St
   r$(D)+" "
43       Sound 1,10,4,5,7
44       Sound 1,0
45       If D<>0 And D>2 Then
46         Goto Avance
47       Endif
48       Until D=0
49       Print At(F,G);Spc(8);At(F
   ,G);
50       Input "PR. ",D$
51       D=Val(D$)
52       Exit If D=Random(3)+2
53       Print At(F,G);Spc(8)
54       For I=1 To 3
55         Sound 1,10,2,3,7
56       Next I
57       Sound 1,0
58     Loop
59   For I=1 To 12 Step 2
60     Sound 1,10,1,5,8
61   Next I
62   Sound 1,0
63   Print "BRAVO....."
64   R$=Input$(1)

```

Ordre

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de réflexion.

Difficulté du jeu : moyenne.

LE JEU :

Les 10 premières lettres de l'alphabet sont affichées dans le désordre ; il faut réussir à les remettre dans l'ordre, avec le moins de coups possible, par des permutations successives de deux des lettres ; mais ceci a aussi pour inconvénient de faire permuter autour d'elles les deux lettres qui entourent celles-ci, suivant le mécanisme expliqué ci-dessous :

B A D G H I J C F E

Ø 1 2 3 4 5 6 7 8 9 : permutation 83

B A D F H I J E G C

Ø 1 2 3 4 5 6 7 8 9 : résultat

LE PROGRAMME :

La boucle I (6-11) définit une variable alphanumérique A\$, contenant les 10 premières lettres de l'alphabet dans le désordre.

La ligne 17 affiche A\$, et la boucle I (19-21) affiche les numéros pour la permutation.

On entre la permutation en ligne 28.

La ligne 31 appelle une première fois le sous-programme d'inversion.

La ligne 36 appelle une deuxième fois le sous-programme d'inversion.

La ligne 37 comptabilise les essais.

La ligne 26 contrôle les résultats.

POUR JOUER :

Lancer le programme par RUN.

Le mot initial est affiché, chaque lettre étant repérée par un numéro de 1 à 9.

A la question INVERSION (I/J), on répond en entrant un nombre de deux chiffres xy ; x est le numéro de la première lettre, et y le numéro de la seconde. Les deux inversions sont alors successivement effectuées, et l'on recommence.

Le nombre de coups sera affiché en fin de partie.

EXTENSIONS POSSIBLES :

On peut imaginer un jeu contre l'ordinateur, qui évaluerait le score du joueur en fonction de ses propres performances ; on peut aussi modifier les règles qui régissent les permutations.

```

1  Rem ORDRE
2  Fullw 2
3  Clearw 2
4  P=Abs(Xbios(4)-2)
5  A$=""
6  For I=1 To 10
7    Repeat
8      R$=Chr$(Random(10)+65)
9      Until Instr(A$,R$)=0
10     A$=A$+R$
11  Next I
12  S=1
13  Do
14    Deftext P+1,16,0,32+20*(P
    >0)
15    Gosub Affiche(-3,2,"RESUL
    TAT",0)
16    Deftext (I+3) Mod (4*P-1)
    ,1,0,18+10*(P>0)
17    Gosub Affiche(-5,6,A$,1)
18    Deftext 1
19    For I=16 To 25
20      Print At(2*I-25,8);Chr$(
    I+32)
21    Next I
22    Deftext 1,0,0,13+7*(P>0)
23    Gosub Affiche(1,14,"ESSAI
    NUMERO "+Str$(S),0)
24    Deftext 1-(P<2),17,0,8+4*(
    P>0)
25    Print At(1,24);Spc(30)
26    Exit If A$="ABCDEFGH IJ"
27    Gosub Affiche(-1,28,"INVE
    RSION (FORME IJ) : ",0)
28    Input "",R$
29    I=Asc(Left$(R$,1))-47
30    J=Asc(Right$(R$,1))-47
31    Gosub Chan
32    If Left$(R$,1)<>"0" And L
    eft$(R$,1)<>"9" Then
33      I=I-1
34      J=J+2
35    Endif
36    Gosub Chan
37    S=S+1
38  Loop
39  Deftext 1,0,0,13+7*(P>0)
40  Print "TROUVE EN ";S;" COUP
    S"
41  For I=1 To 5 Step 2
42    Sound 1,10,I,5,8
43  Next I
44  Sound 1,0
45  R$=Input$(1)
46  Procedure Chan
47    B$=""
48    For T=1 To Len(A$)
49      If T=I Then
50        N$=Mid$(A$,T,1)
51      Endif
52      If T=J Then
53        M$=Mid$(A$,T,1)
54      Endif
55    Next T
56    For T=1 To Len(A$)
57      If T=I Then
58        B$=B$+M$
59        Sound 1,10,T,4,5
60      Else
61        If T=J Then
62          B$=B$+N$
63          Sound 1,10,T,5,5
64        Else
65          B$=B$+Mid$(A$,T,1)
66        Endif
67      Endif
68    Next T
69    A$=B$
70    Sound 1,0
71  Return
72  Procedure Affiche(R,S,T$,H)
73    For I=1 To Len(T$)
74      Deftext 1-(Asc(Mid$(T$,
    I,1)) Mod (4^P))*H)
75      Print At(2*I-R,S);Mid$(
    T$,I,1);
76      Sound 1,10,I,5,1
77      Sound 1,0
78    Next I
79  Return

```

Bridge

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion.

Difficulté du jeu : variable, selon le problème posé.

LE JEU :

Vous ne trouverez pas, dans ce programme, un partenaire pour vos bridges mais, plus exactement, une simulation pour l'affichage d'une donne ; cet affichage est réalisé jeu par jeu. On peut, à partir de là, se poser toutes sortes de problèmes : annonces, jeu de la carte, etc.

LE PROGRAMME :

La ligne 6 définit un jeu de 52 cartes.

La table B\$, (lignes 7 à 10) permettra d'afficher les symboles des couleurs et aussi d'améliorer la désignation de certaines cartes (10, valet, dame, roi et as).

La boucle J (16-46) réalise le tirage de quatre jeux de 13 cartes.

La boucle I (17-43) réalise tirage et impression. La carte est choisie parmi les cartes restantes disponibles, aux lignes 18 à 20.

La ligne 22 calcule la couleur de la carte tirée.

Les lignes 23 et 24 s'occupent de l'impression complémentaire dans le cas où l'on tire un 10, un valet, une dame, un roi ou un as.

L'impression est réalisée par les lignes 29 à 42.

La ligne 45 réalise un arrêt entre chaque affichage.

POUR JOUER :

On lance le programme par RUN.

Une première série de 13 cartes est affichée verticalement. Il suffira d'appuyer sur une touche quelconque pour obtenir la deuxième série de 13 cartes. On procédera de même pour la troisième et la quatrième.

On devra taper la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On peut imaginer de représenter les 13 cartes sous une forme plus conventionnelle.

On peut aussi ajouter la possibilité de faire des annonces (et de les comparer à celles du ST, par exemple), ou introduire des possibilités de jeu de la carte, y compris pour le ST.

```

1  Rem BRIDGE
2  Fullw 2
3  Clearw 2
4  Setcolor 0,1637
5  R$="234567891VDRA"
6  R$=R$+R$+R$+R$
7  Dim T(52),B$(9)
8  For I=1 To 9
9      Read B$(I)
10 Next I
11 For I=0 To Xbios(4)
12     Read K1,K2,K3,K4
13 Next I
14 Deftext 1,16,0,11+7*(Xbios(4)<20)
15 Clearw 2
16 For J=1 To 4
17     For I=1 To 13
18         Repeat
19             X=Random(Len(R$))+1
20             Until T(X)<>1
21             T(X)=1
22             S=Int((X-1)/13)+1
23             T=X-13*Int(X/13)
24             T=T-13*(T=0)
25             Sound 1,10,Int(Rnd*12+1),5,3
26             Sound 1,0
27             A=K1*I-K1+4
28             B=K2*I-K2+1
29             Deffill -(2*S-2)*(Xbios(4)=0)+S,S,2
30             Pbox A,B,A+K3,B+K2
31             If T>=9 Then
32                 C$=B$(T-4)+" "
33             Else
34                 C$=" "
35             Endif
36             Text A+8,B+7," "+Mid$(R$,X,1)+C$
37             Print At(K4*I-K4+6,I);
38             If Xbios(4)>=1 Then
39                 Text A+36,B+7," "+B$(S)+" "
40             Else
41                 Text A+36,B+7,Left$(B$(S),3)+" "
42             Endif
43         Next I
44     Print
45     S$=Input$(1)
46 Next J
47 R$=Input$(1)
48 Deftext 1,0,0,13+7*(Xbios(4)=0)
49 Setcolor 0,1911
50 Data TREFLE,PIQUE,COEUR,CARREAU,0,A,A,0,S
51 Data 72,13,70,9,154,13,110,19,154,16,120,19

```

A travers

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : moyenne.

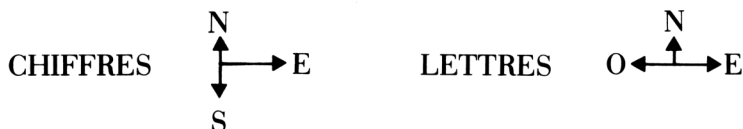
LE JEU :

Sur l'écran apparaît un carré de 9 cases, avec sur le bord vertical des lettres et sur le bord horizontal des chiffres.

L'un des joueurs manipule les chiffres, l'autre les lettres. 3 déplacements sont possibles pour les chiffres, et 3 autres pour les lettres. Les lettres peuvent se déplacer vers le haut, à droite et à gauche et les chiffres vers la droite, en haut ou en bas.

Le gagnant est le premier qui arrive à faire sortir tous ses symboles (les lettres sortent en haut, et les chiffres à droite).

Il n'y a pas de prise, et les sauts ne sont pas autorisés ; on ne peut pas non plus se placer sur une case déjà occupée.



LE PROGRAMME :

Les positions des chiffres et des lettres sont enregistrées dans une table N (6, 6) ; les lignes 9 à 13 définissent les valeurs de départ.

Les lignes 15 à 23 dessinent le terrain de jeu et la ligne 28 affiche le joueur qui a le trait.

La ligne 34 contrôle le coup joué, de même que les boucles I et J (36 à 44), et le changement de joueur est assuré par la ligne 56.

POUR JOUER :

Le terrain de jeu sera affiché après la frappe de RUN et l'ordinateur attend le coup du joueur désigné.

On entrera d'abord le chiffre ou la lettre représentant le pion à déplacer, suivi d'une indication de direction sous la forme N (pour Nord), S (pour Sud), E (pour Est) ou O (pour Ouest).

On pourra ainsi faire 2E.

Si le coup est invalide, il ne sera pas exécuté, et le code du joueur restera à l'affichage ; s'il est autorisé, le déplacement sera effectué, et l'on obtiendra l'affichage correspondant au deuxième joueur.

Le premier qui aura réussi à faire disparaître ses 3 pièces aura gagné.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager d'étendre le jeu avec davantage de chiffres ou de lettres et d'améliorer les contrôles. On peut aussi définir une stratégie pour le ST.

```

1  Rem A TRAVERS
2  P=Abs(Xbios(4)-2)
3  Dim N(6,6),A$(2)
4  K=Random(2)
5  A$(1)="JEU DES CHIFFRES"
6  A$(0)="JEU DES LETTRES"
7  Fullw 2
8  Openw 2
9  For I=1 To 4
10   N(I,1)=Asc("0")+I
11   N(4,I)=Asc("?")+I
12 Next I
13 N(4,1)=0
14 Do
15   For U=1 To 4
16     For V=1 To 4
17       S=N(V,U)-46*(N(V,U)=0)
18       J=(50+22*(P>0))*V
19       I=50*U
20       Deftext 1-(N(V,U)>47)-(N(V,U)>64),0,0,30+14*
(P>0)
21       Text I,J,Chr$(S)
22     Next V
23   Next U
24   Entree:
25   Repeat
26     Repeat
27       Deftext 3-K
28       Print At(1,8);Spc(Len(A$(K))+4);At(1,8);A$(K
);" ";
29       R$=Upper$(Input$(1))
30       Print R$;
31       D$=Upper$(Input$(1))
32       Print D$
33       R$=R$+D$
34       Until Not (D$=Chr$(8) Or (K=1 And D$="0") Or (
K=0 And D$="S"))
35       Kf=0
36       For I=1 To 4
37         For J=1 To 4
38           If Kf=0 And N(I,J)=Asc(Left$(R$,1)) Then
39             Kf=1
40             A=I
41             B=J
42           Endif
43         Next J
44       Next I
45       Until Kf=1
46       Jeu:
47       I=A-(D$="S")+(D$="N")
48       J=B-(D$="E")+(D$="O")
49       If Not ((I=0 And K=0) Or (J>4 And K=1)) Then
50         If N(I,J)<>0 Then
51           Goto Entree
52         Endif
53         N(I,J)=Asc(Left$(R$,1))
54       Endif
55       N(A,B)=0
56       K=Abs(K-1)
57 Loop

```


Evasion

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : solitaire.

Difficulté du jeu : assez difficile.

LE JEU :

Vous êtes enfermé dans un labyrinthe sans issue, et il faut réussir à deviner, le plus vite possible, à quel emplacement vous vous trouvez. Dans la version 1, vous ne vous déplacez pas, quels que soient les essais que vous faites, alors que dans la version 2, si le déplacement est possible, il sera effectué.

LE PROGRAMME :

Le type de jeu (1 ou 2) est choisi en ligne 4.

Le labyrinthe est enregistré dans une table L, définie en ligne 14 (on pourra modifier les DATA pour en dessiner un autre).

Le dessin du labyrinthe est effectué par une boucle WHILE... WEND, aux lignes 23 à 34.

La position du joueur est tirée au hasard et testée en ligne 38.

Le déplacement est entré en ligne 45.

La boucle I (48-59) contrôle si le déplacement est possible. Le résultat est affiché en ligne 60.

Les lignes 65 et 66 permettent d'entrer une solution, contrôlée en ligne 83.

POUR JOUER :

On lance le programme par RUN, puis on choisit la version (joueur fixe ou mobile) en tapant 1 ou 2.

Le labyrinthe est alors dessiné ; on entre le déplacement sous forme d'une suite de lettres, H pour haut, B pour bas, D pour droite, G pour gauche.

On tapera, par exemple HHHDD. Le nombre de cases libres avant la rencontre d'un mur est alors affiché. En version 2, le déplacement est effectué.

A la question "SOLUTION (O/N)", on répondra O si l'on pense avoir trouvé la position du joueur. On entre ensuite les lettres correspondant à la ligne et à la colonne, et un message vient donner confirmation. On tape la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra facilement modifier le dessin du labyrinthe, faire apparaître le joueur, imaginer de le diriger vers une issue, etc.

```

1  Rem EVASION
2  Fullw 2
3  Clearw 2
4  Print "VERSION 1 OU 2 : "
5  Ve=Inp(2)-48
6  P=Abs(Xbios(4)-2)
7  Y=4
8  H=11-12*(P=0)
9  S=24+12*(P=2)
10 Deftext 1,0,0,8-5*(P>0)
11 Read A
12 D=A
13 Clearw 2
14 Dim L(A,A)
15 For I=1 To A
16   Text S*I+50+8*(P=2),10,Chr$(
17     64+I)
18   For J=1 To A
19     L(I,J)=0
20   Next J
21 Next I
22 Deftext 1,0,0,18+12*(P>0)
23 Y=12
24 While A<>0
25   For I=1 To A
26     Read B
27     Deffill 2,2,2
28     Pbox S*B+40,Y,S*B+40+S,Y+H
29     L(1+((Y-4)/H),B)=1
30   Next I
31   Read A
32   Y=Y+H
33   C=64+(Y-4)/H
34   Text 42,Y,Chr$(C)
35 Wend
36 Repeat
37   X=Random(C-65)+1
38   Y=Random(D-1)+1
39 Until L(X,Y)<>1
40 A=X
41 B=Y
42 Deftext 1,0,0,13+7*(P>0)
43 Repeat
44   T=T+1
45   Print At(1,21+2*(P>0));Sp(6
46     0);At(1,21+2*(P>0));
47   Input "DEPLACEMENT (HHGGHB.
48     ... ) :";R$
49   R%=Upper$(R$)
50   S=0
51   For I=1 To Len(R$)
52     X=X+(Mid$(R$,I,1)="H")-(Mi
53       d$(R$,I,1)="B")
54     Y=Y+(Mid$(R$,I,1)="G")-(Mi
55       d$(R$,I,1)="D")
56   If L(X,Y)=1 Then
57     X=X0
58     Y=Y0
59   Else
60     S=S+1
61   Endif
62 Next I
63 Sound 1,0
64 Print At(1,22+2*(P>0));"SOLU
65   TION ( O/N )";
66   R%=Upper$(Chr$(Inp(2)))
67   Print At(1,22+2*(P>0));Sp(4
68     0);
69   If Ve<>2 Then
70     X=A
71     Y=B
72   Endif
73 Until R$<>"N"
74 If Ve=2 Then
75   A=X
76   B=Y
77 Endif
78 Print At(1,21+(P>0));Sp(40);A
79 t(1,21+(P>0));
80 Input "LIGNE :";X$
81 U=Asc(Upper$(X$))-64
82 Print At(1,21+(P>0));Sp(40);A
83 t(1,21+(P>0));
84 Input "COLONNE :";Y$
85 V=Asc(Upper$(Y$))-64
86 If A=U And B=V Then
87   Print At(1,22+(P>0));"BRAVO.
88   TROUVE EN ";T+1;" ESSAIS";
89 Else
90   Print At(1,22+(P>0));"PERDU.
91   C'ETAIT LIGNE ";Chr$(64+A);",
92   COLONNE ";Chr$(64+B);
93 Endif
94 For I=1 To 12 Step 2
95   Sound 1,10,I,5,20
96 Next I
97 Sound 1,0
98 R%=Input$(1)
99 Data 20,1,2,3,4,5,6,7,8,9,10,1
100 1,12,13,14,15,16,17,18,19,20
101 Data 4,1,5,13,20,15,1,3,5,7,8,
102 9,10,11,12,13,15,16,17,18,20
103 Data 7,1,3,5,12,13,15,20,12,1,
104 3,8,9,10,12,13,15,17,18,19,20
105 Data 9,1,3,4,5,6,10,15,18,20,9
106 1,3,8,10,12,14,15,18,20
107 Data 9,1,3,5,7,8,10,12,18,20,9
108 1,5,7,12,14,16,17,18,20
109 Data 10,1,3,4,5,7,9,11,12,14,2
110 0,8,1,9,14,15,16,17,18,20
111 Data 20,1,2,3,4,5,6,7,8,9,10,1
112 1,12,13,14,15,16,17,18,19,20,0

```

Permutations

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion.

Difficulté du jeu : moyenne.

LE JEU :

Le ST vous demande d'abord le nombre de lettres que vous désirez permuter (pas plus de 12). A partir de là, il va vous afficher le nombre correspondant de lettres prises dans le début de l'alphabet, d'abord dans le désordre (DEBUT) puis dans l'ordre (FIN).

Vous devez réussir à remettre dans l'ordre les lettres données au début, en utilisant des permutations sur les n premières lettres à partir de la gauche, avec le moins d'essais possible.

LE PROGRAMME :

La ligne 7 permet de choisir le nombre de lettres, et la ligne 8 contrôle que l'on n'en demande pas plus de 12.

La ligne 14 réalise le brouillage, en tirant au hasard la nouvelle position de chaque lettre. Les lettres sont ensuite ajoutées au fur et à mesure de leur tirage, dans une variable B\$ (ligne 16), supprimées dans A\$ et replacées en C\$, dans le bon ordre (lignes 17-18).

La ligne 21 contrôle le nombre de lettres de B\$.

Les affichages sont effectués aux lignes 23 à 28.

Le retournement est fait, lettre par lettre, dans la boucle I (38-40) ; le résultat est placé dans B\$ en ligne 41, affiché par la ligne 48 et comparé au résultat final en ligne 49.

Le nombre d'essais sera affiché en fin de partie, grâce à la ligne 50.

POUR JOUER :

Après avoir tapé RUN, on obtient l'affichage de la question NOMBRE DE LETTRES (DE 1 A 12) ; on répond en frappant N, suivi de RET. L'affichage suivant présente l'ordre initial (DEBUT) et l'ordre à obtenir (FIN). On joue en indiquant le nombre de lettres à permuter à partir de la gauche, par exemple : sur un affichage initial de GEABDFC, si l'on tape 3 et ENTREE, on obtiendra AEGBDFC.

On poursuivra jusqu'à l'obtention de l'ordre désiré.

Le score sera affiché en fin de partie ainsi qu'une question AUTRE PARTIE (O/N) ; répondre O si l'on désire faire une autre partie.

EXTENSIONS POSSIBLES :

On pourra utiliser le même type de jeu pour, par exemple, reconstituer un dessin caché dont les diverses parties seraient repérées par des lettres. On peut aussi réaliser des permutations plus complexes (par exemple, le fait de permuter deux lettres permute les deux voisins de la lettre la plus à gauche, etc.).

```

1  Rem PERMUTATIONS
2  P=Abs(Xbios(4)-2)
3  Deftext 1,0,0,13+7*(P>0)
4  Fullw 2
5  Repeat
6    Clearw 2
7    Input "NBRE. DE LETTRES (
      1 A 12 ):";N
8    Until N=Int(N) And N>=1 And
      N<=12
9    A$="ABCDEFGHIIJKL"
10   C$=String$(Len(A$)," ")
11   B$=""
12   Clearw 2
13   Repeat
14     X=Random(N)+1
15     If Mid$(A$,X,1)<>" " Then
16       B$=Mid$(A$,X,1)+B$
17       Mid$(C$,X,1)=Mid$(A$,X,
18         1)
19       Mid$(A$,X,1)=" "
20       S=1
21     Endif
22   Until Len(B$)=N
23   C$=Left$(C$,Len(B$))
24   Print At(1,4);"DEBUT   ";
25   Deftext 1,0,0,8+2*(P>0)
26   Gosub Affiche(8,6+2*(P>0),B
27     $)
28   Print At(1,7);"FIN     ";
29   Deftext 1,0,0,8+2*(P>0)
30   Gosub Affiche(8,11+4*(P>0),
31     C$)
32   Repeat
33     Repeat
34       Print At(1,10);"PERMUTA
35       TION NUMERO ";S
36       Print
37       Input "VOTRE QUANTITE "
38       ;L
39       Until L=Int(L) And L>=1 A
40       nd L<=N
41       S=S+1
42   A$=""
43   Print At(1,12);Spc(30)
44   For I=1 To L
45     A$=A$+Mid$(B$,L-I+1,1)
46   Next I
47   B$=A$+Right$(B$,Len(B$)-L
48     )
49   For I=1 To 5
50     Sound 1,10,I,5,8
51   Next I
52   Sound 1,0
53   Print At(1,15);"RESULTAT
54     :";
55   Deftext 1,0,0,8
56   Gosub Affiche(8+6*(P>0),2
57     4+10*(P>0),B$)
58   Until B$=C$
59   Print At(1,17);"TROUVE EN "
60   ;S-1;" ESSAIS"
61   For I=1 To 12 Step 2
62     Sound 1,10,I,4,10
63   Next I
64   Sound 1,0
65   Print
66   Print "AUTRE PARTIE ( O/N )
67     ";
68   R$=Upper$(Input$(1))
69   If R$="O" Then
70     Run
71   Else
72     End
73   Endif
74   Procedure Affiche(L,H,P$)
75     For I=1 To Len(P$)
76       Deftext 1+Random(4^P)
77       Print At(2*I+L,H);Mid$(
78         P$,I,1)
79     Next I
80     Sound 1,10,I,5,1
81     Sound 1,0
82     Deftext 1,0,0,13+7*(P>0)
83   Return

```

Répétitions

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu de mémoire (1 ou plusieurs joueurs).

Difficulté du jeu : moyenne.

LE JEU :

Des nombres vont apparaître un par un.

Le joueur devra à chaque coup les répéter tous, depuis le début (ce qui signifie qu'il en aura un de plus à taper à chaque fois).

Il devra essayer de tenir le plus longtemps possible. Le meilleur score est mémorisé après chaque partie, ce qui permettra éventuellement de jouer à plusieurs.

LE PROGRAMME :

La ligne 8 appelle un sous-programme (lignes 59 à 69) qui donne la règle du jeu.

La ligne 15 fabrique les chiffres, qui sont ensuite stockés dans une variable alphanumérique B\$.

La ligne 20 permet l'impression du dernier chiffre au hasard sur l'écran, et la durée de l'affichage est déterminée par la boucle I (21-22).

La ligne 25 attend la réponse du joueur, qui est comparée à la réponse correcte.

La ligne 37 marque les points et la ligne 38 contrôle le meilleur score.

Les affichages de fin de partie sont effectués de la ligne 41 à la ligne 49.

POUR JOUER :

Le lancement du programme par RUN provoque l'affichage de la règle du jeu, puis d'un chiffre, qui disparaît. L'appareil pose alors la question VOTRE SEQUENCE.

On lui répond en redonnant le chiffre, suivi de RET. Si c'est exact, on a un court instant l'affichage de EXACT. ATTENTION, puis un nouveau chiffre est affiché, qui s'efface, et de nouveau la question VOTRE SEQUENCE. On retape alors les deux chiffres déjà obtenus, suivis de RET.

On continue ainsi (3 chiffres, puis 4, 5, etc.) jusqu'à ce que l'on fasse une erreur, ce qui provoque l'affichage du score (et du meilleur score obtenu jusqu'à présent), suivi de la question AUTRE PARTIE (O/N) ; on tape O pour continuer.

On peut aussi quitter le programme à tout moment, en tapant la combinaison CSA..

EXTENSIONS POSSIBLES :

On pourra réaliser des jeux analogues et plus ou moins perfectionnés en remplaçant les chiffres par des lettres ou des couleurs ou des sons...

```

1  Rem REPETITIONS
2  P=Abs(Xbios(4)-2)
3  Openw 2
4  Fullw 2
5  M=0
6  Debut:
7  Clearw 2
8  Gosub Tex
9  H=0
10 U=240-240*(P<2)
11 V=130-130*(P=0)
12 B$=""
13 Do
14   Clearw 2
15   Z=Random(10)
16   B$=B$+Right$(Str$(Z),1)
17   X=Random(U)+40
18   Y=Random(V)+40
19   Deftext 1,0,0,32+15*(P>0)
20   Text X,Y,Str$(Z)
21   For I=1 To 1500+Rnd*1000
22     Next I
23   Clearw 2
24   Print At(1,8);
25   Input "VOTRE SEQUENCE :";
   R$
26   Exit If R$<>B$
27   Print
28   For I=1 To 7 Step 2
29     Sound 1,10,I,5,10
30   Next I
31   Sound 1,0
32   A$="EXACT. ATTENTION"
33   K=0
34   Gosub Lec(19+10*(P>0))
35   For I=1 To 1200
36     Next I
37     H=H+1
38     M=M+(M-H)*(M<H)
39   Loop
40   Clearw 2
41   A$="VOTRE SCORE EST DE "+Str
   r$(H)
42   K=0
43   Gosub Lec(16+7*(P>0))
44   A$="MEILLEUR SCORE ACTUEL "
   +Str$(M)
45   K=A

46   Gosub Lec(16+7*(P>0))
47   A$="AUTRE PARTIE ( O/N ): "
48   K=2*A
49   Gosub Lec(16+7*(P>0))
50   R$=Upper$(Input$(1))
51   If R$="O" Then
52     Goto Debut
53   Endif
54   For I=1 To 12
55     Sound 1,10,I,5,10
56   Next I
57   Sound 1,0
58   R$=Input$(1)
59   Procedure Tex
60     A$="VOUS DEVEZ REPETER TO
   UTE LA SERIE DES NOMBRES AP
   PARUS DEPUIS "
61     A=62+27*(P=2)
62     K=0
63     Gosub Lec(17+11*(P>0))
64     A$=" LE DEBUT DU JEU. IL
   Y EN A UN DE PLUS A CHAQUE
   FOIS. APPUYEZ "
65     Gosub Lec(17+11*(P>0))
66     A$=" SUR UNE TOUCHE POUR
   COMMENCER LA PARTIE"
67     Gosub Lec(17+11*(P>0))
68     A$=Input$(1)
69   Return
70   Procedure Lec(Ta)
71     For I=1 To Len(A$)
72       Co=1+(I Mod 4^P)
73       K=K+1
74       F=2+(K Mod A)
75       G=4+4*(Int(K/A))
76       Gosub Affiche(Co,Ta,F,G
   ,Mid$(A$,I,1))
77       Sound 1,10,Int(Rnd*12+1
   ),Int(Rnd*4+2),2
78       Sound 1,0
79     Next I
80   Return
81   Procedure Affiche(C,H,F,G,C
   $)
82     Deftext C,0,0,H
83     Print At(F,G);C$
84     Deftext 1,0,0,13+7*(P>0)
85   Return

```


Niveau 4

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Adresse	Pour un seul joueur, très visuel.
Zombies	Un peu farfelu, mais très spectaculaire, pour un joueur.
Diago	Etes-vous vraiment sûr de vos réflexes ?
Le singe	Un jeu de lettres pour deux joueurs, où il faut de bons réflexes.
Affichage publicitaire	Si vous ne savez pas quoi mettre dans votre vitrine.
Chapeaux	Soyez très observateur, et vous réussirez, pour deux joueurs.
Clavier	Apprenez à taper à la machine en vous amusant.
Dessin de dés	Suffisamment souple pour être utilisé partout.
Smokey	Fumez sans risque avec ce programme original pour un joueur.
Captures dans l'espace	Votre mission serait facile, sans ces damnées météorites.
Danger spatial	Serez-vous assez fort pour éviter toutes les météorites ?
Course de voitures	Un jeu de réflexe pas trop difficile pour un joueur.
Obstacles	Beaucoup d'obstacles à éviter dans ce jeu d'adresse, pour un joueur.
Poker	Une base de départ pour mettre au point votre programme.
Course de chevaux	Assez visuel : un jeu de hasard, pour plusieurs joueurs.
Sous-marin	Un classique des jeux vidéo, pour un joueur.
Horloge 1	Ne jetez pas votre vieille télévision : transformez-la en horloge.
Échiquier	Une base graphique, pour vos propres créations.
L'électricien fou	Logique et difficile, pour un seul joueur.
L'indésirable	Vous êtes propriétaire. Arriverez-vous à mettre la main sur ce locataire indésirable ?

Techniques d'animation, Objets graphiques mobiles, Fenêtres et menus

LES TECHNIQUES D'ANIMATION ELEMENTAIRES

Le principe d'un jeu graphique, en animation, est de laisser le jeu se dérouler tout seul, d'une certaine manière, pendant un certain temps, tant qu'aucune intervention n'a été faite par le joueur.

Il faut donc disposer de la possibilité d'observer s'il y a eu ou non intervention du joueur, sans pour cela interrompre le déroulement du programme.

Dans le Basic GFA, cette possibilité est offerte par la fonction **INKEY\$** qui effectue une scrutation des touches du clavier et enregistre le nom de la dernière touche frappée (en réalité, il existe un "buffer" de clavier, qui fait que l'on pourra enregistrer plusieurs frappes effectuées à la suite les unes des autres).

On pourra utiliser **INKEY\$** de deux façons :

- soit dans une boucle (réalisation d'un mouvement). On saisit **INKEY\$** et on le teste pour une éventuelle sortie de la boucle.
- soit dans une attente : tant que **INKEY\$** est vide (rien n'a été frappé), on attend, et l'on sort dès qu'une touche a été frappée.

Certains programmes, utilisant **INKEY\$** dans une boucle d'attente, réalisent en fait une combinaison des deux méthodes.

DEPLACEMENTS D'OBJET DE BAS EN HAUT SUR L'ECRAN : SCROLLING D'ECRAN

Il est difficile, en Basic, de gérer le déplacement simultané sur l'écran d'un nombre élevé d'objets.

Le ST réalise automatiquement un “**scrolling**” (c’est-à-dire un déplacement de l’ensemble de l’image) vers le haut, lorsque l’on écrit sur la dernière ligne en bas de l’écran.

Cette propriété a été mise à profit dans plusieurs jeux : cela permet en effet de déplacer d’une ligne vers le haut les divers objets (textes ou graphiques) présents sur l’écran à un moment donné. La ligne du haut est bien évidemment perdue.

Si un objet doit rester immobile sur l’écran, il sera nécessaire de l’effacer dans sa nouvelle position, et de le redessiner à son emplacement initial.

Le Basic GFA possède en outre une très puissante instruction **BMOVE**, qui permet de déplacer des blocs entiers dans la mémoire. Il sera donc possible, avec cette instruction, de faire des scrollings beaucoup plus doux, et dans toutes les directions.

RECONNAISSANCE D’UN OBJET SUR L’ECRAN : POINT

Il peut être nécessaire, dans de nombreux jeux graphiques en animation, de déterminer s’il y a ou non eu collision entre deux objets présents sur l’écran. Il est possible dans ce cas d’utiliser l’instruction **POINT (X,Y)** qui renvoie la couleur du point se trouvant aux coordonnées X et Y sur l’écran.

LES OBJETS GRAPHIQUES MOBILES : SPRITE

Le Basic GFA permet de définir de façon très simple des SPRITE (on dit aussi lutins). Il s’agit d’objets graphiques, de taille maximum 16*16. Il ne s’agit pas ici de véritables SPRITE au sens habituel (un SPRITE est en général piloté indépendamment du programme : ils restent en mouvement même si le programme s’arrête), mais ils offrent néanmoins de grands avantages pour réaliser des éléments mobiles, par leur facilité d’utilisation. On notera en particulier qu’il n’est possible d’avoir qu’un SPRITE d’un certain type à la fois sur l’écran (on peut néanmoins le dupliquer), ce qui permet de déplacer un SPRITE sans s’occuper des problèmes d’effacements éventuels.

La gestion des SPRITE effectuée par le Basic GFA peut présenter quelques défauts, liés au mode graphique utilisé (**DEFMODE**). On pourra résoudre certains de ces défauts (effacement incomplet, par exemple) en affichant le SPRITE en dehors de l’écran, avant de le faire disparaître.

Le Basic GFA dispose également des deux puissantes instructions **GET** et **PUT**. **GET** permet d’enregistrer dans une variable alphanumérique une zone rectangulaire de l’écran, et **PUT** recopie cette zone à n’importe quel emplacement de l’écran.

LA GESTION DES FENETRES

L'instruction **ALERT** permet une gestion très facile des fenêtres d'alerte.

La gestion des fenêtres proprement dites peut se faire par appel du **GEM**, en utilisant la variable réservée **WINDTAB**, et en effectuant un **DPOKE** à l'adresse convenable.

LES MENUS DEROULANTS : MENU

Le GFA Basic dispose de plusieurs instructions qui permettent une gestion des menus déroulants de façon quasi professionnelle. Les menus sont définis très simplement à l'aide d'un tableau alphanumérique contenant les divers textes à afficher. Le menu est ensuite activé avec l'instruction **MENU A\$**.

Adresse

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexe (pour 1 joueur).

Difficulté du jeu : difficile.

LE JEU :

Des chiffres, entre 0 et 9, vont apparaître au hasard, en divers emplacements de l'écran. Ils restent affichés pendant un temps très court. Le joueur doit réussir à appuyer sur la touche numérique correspondante, au moment où le chiffre est affiché.

Le chiffre tapé par le joueur sera alors affiché en grand format sur la gauche de l'écran ; et chaque réponse exacte rapporte 10 points. La durée de la partie est aléatoire et le score sera affiché en fin de partie.

LE PROGRAMME :

Les lignes 7 et 8 contiennent, dans deux variables alphanumériques C\$ et B\$, les références qui seront utilisées pour dessiner les chiffres en grand format.

L'emplacement du chiffre tiré par le ST est choisi aux lignes 10 et 11 et le chiffre est tiré en ligne 12.

La ligne 14 permet l'affichage du chiffre et la ligne 15 détermine la durée d'affichage. La ligne 22 définit la durée du jeu, et la fin de partie est traitée par la ligne 26.

La boucle I (34-40) réalise le dessin grand format du chiffre frappé par le joueur.

Le contrôle de l'exactitude du résultat est effectué par la ligne 41.

POUR JOUER :

Le programme sera lancé par RUN, puis un premier chiffre apparaît, après une courte pause, vers le milieu de l'écran. Le joueur devra réagir suffisamment vite, tant qu'il reste à l'affichage, et le retaper. Le chiffre qu'il a tapé se dessinera alors en haut à gauche de l'écran, en grand format.

Si le résultat est exact, le joueur marquera 10 points.

Le temps de déroulement de la partie sera fonction des chiffres qui apparaîtront à l'affichage.

En fin de partie, le score sera affiché. Il sera possible de s'arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager de modifier le jeu pour faire participer plusieurs joueurs, en modifiant éventuellement l'affichage et en codant les touches du clavier mises à la disposition de chacun des joueurs.

On pourra aussi enlever des points à chaque résultat faux, modifier les règles de calcul du score et la durée du jeu, etc.

```

1  Rem ADRESSE
2  P=Xbios(4)
3  H=1-(P>0)
4  K=1-(P=2)
5  T=100
6  Z=0
7  C$="111100001101"
8  B$="144413333131211313144133121311214113333141411
   4131"
9  Repeat
10     X=Random(400+200*(P=0))+100
11     Y=Random(200+100*(P<2))+70
12     N=Random(10)
13     Deftext Random(2*Abs(P-2))+1,1,0,32
14     Text X,Y,Chr$(48+N)
15     For I=0 To Rnd*800+600
16         A$=Inkey$
17         Exit If A$<>""
18     Next I
19     If A$<>"" Then
20         Gosub Dessin
21     Endif
22     T=T-N
23     Text X,Y," "
24 Until T<0
25 Deftext 1,0,0,13
26 Print At(1,19);"VOTRE SCORE : ";Z;" SUR 100"
27 For I=1 To 12 Step 2
28     Sound 1,10,I,5,10
29 Next I
30 Sound 1,0
31 R$=Input$(1)
32 Procedure Dessin
33     U=Val(A$)
34     For I=1 To 5
35         A=3*Val(Mid$(B$,5*U+I,1))-2
36         For J=0 To 2
37             Deffill 1,2*Val(Mid$(C$,A+J,1)),8*Val(Mid$(C
$,A+J,1))
38             Pbox 10+16*H*J,16*K*I+30,10+16*H*(J+1),16*K*
(I+1)+30
39         Next J
40     Next I
41     If U=N Then
42         Z=Z+10
43         Sound 1,10,4,5,8
44         Sound 1,0
45     Endif
46 Return

```

Zombies

REFERENCES :

Difficulté de programmation : assez difficile.

Catégorie de jeu : solitaire.

Difficulté du jeu : assez facile.

LE JEU :

Les ZOMBIES sont des êtres malfaisants qui vont essayer de vous détruire ; mais vous allez, vous aussi, essayer de les utiliser à votre profit, pour débarrasser votre espace vital des trous noirs qui l'obscurcissent. Sur l'écran, est tracé un cadre qui définit votre espace vital. Dans ce cadre, vont apparaître de façon aléatoire les ZOMBIES (ce sont des Z), les trous noirs (des carrés noirs) et le joueur (un O). Les ZOMBIES convergent vers le joueur pour essayer de le capturer, mais celui-ci se déplace plus rapidement qu'eux, et il va essayer de leur faire avaler tous les trous noirs, sans se faire prendre.

La partie se termine, soit avec la disparition du joueur, soit par l'élimination de tous les ZOMBIES.

LE PROGRAMME :

La position des ZOMBIES est définie par un tableau **Z** dimensionné en ligne 3. Le tableau **T** contiendra la position des trous.

Les lignes 6 à 19 (boucle I) dessinent le cadre, les ZOMBIES et les trous noirs.

La ligne 20 définit la position initiale du joueur (variable **K**).

Une deuxième boucle I (24-41) réalise le déplacement du joueur.

Les lignes 47 à 54 contrôlent le déplacement du joueur.

POUR JOUER :

Après avoir lancé l'exécution par **RUN**, le joueur devra se déplacer en utilisant les touches 4, 6, 2 et 8 et en veillant à ne pas sortir des limites du terrain.

Il devra manœuvrer de façon à faire venir les ZOMBIES sur les trous noirs, sans se faire lui-même coincer.

Pour arrêter la partie avant la fin, il suffira de taper la combinaison **CSA**.

EXTENSIONS POSSIBLES :

Il serait intéressant d'agrandir le terrain de jeu, d'augmenter le nombre de ZOMBIES ou de trous noirs, d'améliorer les contrôles, de placer le joueur de façon plus aléatoire, éventuellement de modifier la règle.

On pourra aussi, bien sûr, améliorer la qualité du dessin et introduire la couleur.

```

1  Rem ZOMBIES
2  P=Abs(2-Xbios(4))
3  Dim T(22,39),Z(12)
4  Fullw 2
5  Openw 2
6  For I=0 To 19
7    Deftext 2#P+1,1,0,13+7*(P
8    >0)
9    Print At(1,I+2);"#";At(I+
10   2,1);"#";At(I+2,22);"#";
11   If Rnd<0.6 And I<18 Then
12     J=Random(18)+2
13     Deftext P+1+(P<>0)
14     Print At(J,I+1);Chr$(12
15     7);
16     T(I,J)=1
17   Endif
18   S=Int(Rnd*14+2)
19   Z(Int(I/2)+1)=4+S
20   Deftext 2#P+1
21   Print At(22,I+2);"#";
22 Next I
23 K=Random(18)+2
24 Do
25   Deftext 1
26   T=0
27   For I=1 To 4
28     T=T+Z(I)
29     If Z(I)<>0 Then
30       Print At(Z(I+4),Z(I))
31       ;" "
32       Z(I)=Z(I)-Sgn(Z(I)-S)
33       Z(I+4)=Z(I+4)-Sgn(Z(I
34       +4)-K)
35       If T(Z(I),Z(I+4))=1 T
36       hen
37         Z(I)=0
38         T(Z(I),Z(I+4))=0
39       Else
40         Print At(Z(I+4),Z(I
41         ));"Z"
42       Endif
43       If Z(I)=S And Z(I+4)=
44       K Then
45         Print At(1,17);"VOU
46         S AVEZ PERDU"
47       Gosub Fin
48       Endif
49       Next I
50       If T=0 Then
51         Print At(1,17);"BRAVO.
52         TOUS LES ZOMBIES SONT MORTS
53         "
54       Gosub Fin
55       Endif
56       Deftext 3-4*(P=2)
57       Print At(K,S);"0"
58       For J=1 To 2400
59         A$=Inkey$
60         Exit If A$<>" "
61       Next J
62       Print At(K,S);" "
63       K=K-2*((A$="6")-(A$="4"))
64       S=S-2*((A$="2")-(A$="8"))
65       If S<=1 Or S>=22 Or K<=1
66       Or K>=21 Then
67         Print At(1,17);"DOMMAGE
68         "
69       Gosub Fin
70       Endif
71       Loop
72     Procedure Fin
73     For I=12 To 1 Step -2
74       Sound 1,10,I,3,10
75     Next I
76     Sound 1,0
77     End
78     Return

```


Diago

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : réflexe (1 seul joueur).

Difficulté du jeu : difficile.

LE JEU :

On lance le programme par RUN et on obtient un premier affichage, qui vous renseigne sur la performance à réaliser (par exemple, 5 points noirs ou 17 points noirs...), puis le temps est affiché et décompté, à partir de 10 secondes. Le jeu commence lorsque l'on arrive à 0 seconde. Des ronds apparaissent alors en bas et à gauche de l'écran, formant une ligne diagonale, et il vous faudra appuyer sur une touche quelconque pour arrêter l'affichage des points dès que vous estimez que le nombre de points demandé au début est apparu à l'écran.

On obtient alors un affichage BRAVO ou PERDU selon le résultat obtenu.

LE PROGRAMME :

Le nombre de points noirs sera choisi au hasard, en ligne 10.

Le temps est décompté par la ligne 19.

Les points sont imprimés et la ligne diagonale est fabriquée par la ligne 28.

La pression d'une touche est détectée en ligne 31, et la ligne 32 contrôle la fin de partie (aucun réflexe).

Les affichages de fin sont exécutés par les lignes 37 à 51.

POUR JOUER :

On lancera l'exécution par RUN.

On obtient alors l'affichage de la phrase LE BON REFLEXE EST A... qui indique le nombre de points gagnants.

Le temps sera ensuite décompté sous la forme IL RESTE ... SECONDES qui indique le temps restant avant l'apparition de la diagonale.

A l'affichage de 0, la diagonale apparaît et commence à se déplacer sur l'écran.

Si l'on estime que le nombre de points réclamé a été atteint, on devra presser une touche quelconque.

L'affichage indique alors BRAVO si l'on a gagné, ou PERDU dans le cas contraire ; puis la phrase AUTRE ESSAI (O/N) est affichée ; répondre O pour faire une autre partie.

EXTENSIONS POSSIBLES :

On pourra envisager une adaptation du jeu à deux joueurs ou plus, en laissant la diagonale se former complètement et en testant la coïncidence lors de la pression de la touche affectée au joueur.

On peut aussi réaliser un graphisme plus soigné et modifier le principe du déplacement des objets sur l'écran (déplacement horizontal, vertical ou apparition au hasard...).

```

1  Rem DIAGO
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Deffill 1,2,2
6  Fill 1,1
7  Dim A$(2)
8  A$(0)=" "
9  A$(1)="S"
10 X=Random(15)+3
11 S=9
12 Deftext 1,0,0,18
13 P$="LE BON REFLEXE EST A "+
   Str$(X)+" "
14 Gosub Aff(10,80+30*(P>0),P$
   )
15 Repeat
16   Deftext 3,1,0,20
17   P$="IL RESTE "+Str$(S)+"
   SECONDE"+A$(Abs(S>1))+" "
18   Gosub Aff(10,150+40*(P>0)
   ,P$)
19   S=S-1
20   Sound 1,10,4,5,8
21   Sound 1,0
22   Pause 50
23 Until S=0
24 Deftext P+1,0,0,13+7*(P>0)
25 Clearw 2
26 U=0
27 Do
28   Print At(U+1,21);"0"
29   Print
30   For I=1 To 40
31     R$=Inkey$
32     Exit If R$<>""
33   Next I
34   U=U+1
35   Exit If R$<>"" Or U>19
36 Loop
37 If U=X Then
38   Print "BRAVO"
39 Else
40   If U>19 Then
41     Print "PERDU"
42   Else
43     Print "RATE...."
44   Endif
45 Endif
46 For I=1 To 12 Step 2
47   Sound 1,10,I,5,10
48 Next I
49 Sound 1,0
50 Print
51 Print "AUTRE ESSAI ( O/N )"
52 A$=Upper$(Chr$(Inp(2)))
53 If A$="O" Then
54   Run
55 Else
56   End
57 Endif
58 Procedure Aff(A,B,P$)
59   For I=1 To Len(P$)
60     Text A+12*I,B,Mid$(P$,I
   ,1)
61   Next I
62 Return

```

Le singe

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : deux joueurs.

Difficulté du jeu : difficile.

LE JEU :

Il s'inspire de l'idée déjà ancienne du singe qui, frappant au hasard sur une machine à écrire, a des chances non nulles de finir par écrire un texte signifiant.

Le rôle du singe est, ici, tenu par le ST, qui va afficher des groupes de 3 à 6 lettres pendant un temps relativement court.

Si l'un des joueurs estime qu'il peut réussir à faire un mot avec les lettres proposées, il doit taper rapidement sur sa touche de jeu (1 ou Ø).

Le mot devra alors être accepté ou refusé par l'adversaire, qui répondra JUSTE ou FAUX.

Chaque mot juste rapporte 1 point, chaque mot faux enlève 1 point.

LE PROGRAMME :

La ligne 6 contient une liste V\$ des voyelles courantes et la ligne 7 une liste C\$ des consonnes courantes.

Il sera possible de modifier ces deux listes pour augmenter, par exemple, la fréquence d'apparition de certaines lettres (on pourra ainsi mettre deux E dans V\$ ou deux R dans C\$, etc.).

La ligne 10 définit la longueur du mot (en principe, de 3 à 6 lettres, mais on peut également modifier cette longueur).

La boucle I (12-21) permet la confection d'un "mot" en prenant une lettre, soit dans l'alphabet complet, soit dans la liste C\$ (ligne 15), soit dans la liste V\$ (ligne 18).

Les lignes 23 à 30 temporisent l'apparition du résultat.

La ligne 31 gère la fin de partie, avec affichage du score.

La ligne 32 regarde si l'un des joueurs a réagi.

Enfin, les lignes 36 et 37 comptabilisent les points de chaque joueur.

POUR JOUER :

On lance le programme en tapant RUN.

La première série de lettres est alors affichée sur l'écran. Le joueur de gauche devra taper 1 s'il voit qu'il peut réaliser un mot, celui de droite devra taper Ø.

Si rien n'a été frappé, une autre série de lettres sera affichée peu de temps après.

Si l'un des joueurs a frappé Ø ou 1, la question (J)USTE ou (F)AUX est alors affichée. On répond J si le mot est correct, F si le mot est incorrect.

Les points sont alors calculés par le ST, et la série de lettres suivante est alors affichée.

Pour sortir du programme, il faudra taper P au lieu de Ø ou 1 ; le score final est alors affiché.

EXTENSIONS POSSIBLES :

On pourra modifier la longueur des mots, la fréquence d'apparition des lettres, le décompte des points, la règle du jeu, etc.

```

1  Rem LE SINGE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  A=0
5  B=0
6  V$="AEIOU"
7  C$="BCDGLMNPRTV"
8  Do
9      Clearw 2
10     X=Random(4)+3
11     M$=""
12     For I=1 To X
13         E$=Chr$(Random(26)+65)
14         If Rnd<0.4 Then
15             E$=Mid$(C$,Int(Len(C$)*Rnd+1),1)
16         Endif
17         If Rnd<0.5 Then
18             E$=Mid$(V$,Int(Len(V$)*Rnd+1),1)
19         Endif
20         M$=M$+E$
21     Next I
22     Gosub Aff(28+10*(P=2),2,2,M$)
23     For I=1 To Len(M$)
24         Sound 1,10,Int(Rnd*12+1),Int(Rnd*4+3),8
25     Next I
26     Sound 1,0
27     For I=1 To 1600
28         R$=Upper$(Inkey$)
29         Exit If R$<>""
30     Next I
31     Exit If R$="P"
32     If R$="1" Or R$="0" Or R$="&" Or R$="à" Then
33         P$="(J)USTE/(F)AUX  "
34         Gosub Aff(16+9*(P=2),0,7,P$)
35         S$=Upper$(Chr$(Inp(2)))
36         A=A-(S$="J" And (R$="1" Or R$="&"))+(S$="F" An
d (R$="1" Or R$="&"))
37         B=B-(S$="J" And (R$="0" Or R$="à"))+(S$="F" An
d (R$="0" Or R$="à"))
38     Endif
39 Loop
40 For I=1 To 12
41     Sound 1,10,I,5,8
42 Next I
43 Sound 1,0
44 Deftext 1,0,0,13+7*(P>0)
45 Print At(1,15);"SCORE DE A : ";A;"  DE B : ";B
46 R$=Input$(1)
47 Procedure Aff(T,L,H,P$)
48     Deftext 1+Random(4^P),0,0,T
49     For I=1 To Len(P$)
50         Print At(2*I+L,H);Mid$(P$,I,1)
51     Next I
52     Deftext 1,0,0,13
53 Return

```

Affichage publicitaire

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : affichage publicitaire.

Difficulté du jeu : aucune.

LE JEU :

Un texte quelconque devra être introduit en début de programme. Il pourra être relativement long (4 ou 5 lignes) ; on choisira ensuite le sens et la vitesse de défilement, à la suite de quoi le texte va apparaître (une seule ligne à la fois) et commencera à défiler, à la vitesse choisie, et dans le sens choisi.

LE PROGRAMME :

Le texte sera introduit lettre par lettre, grâce aux lignes 8 à 18.

Le sens de défilement est choisi en ligne 24.

Le texte sera affiché par les lignes 36 à 51 et la durée de l'affichage (vitesse de défilement) est choisie par la ligne 27, et contrôlée par 47.

La ligne 48 permettra de changer, si besoin est, les conditions du défilement.

POUR JOUER :

On lancera le programme par RUN, et à l'affichage de la question ENTREZ VOTRE TEXTE, on introduira le texte à faire défiler, suivi de RET.

On peut corriger en utilisant la touche | ← (Backspace). On obtient ensuite l'affichage DÉFILEMENT A DROITE (D) OU A GAUCHE (G) auquel on répondra en tapant, soit D, soit G.

Puis à la question VITESSE, on répondra en introduisant un grand nombre pour une vitesse rapide, et un petit pour une vitesse lente (par exemple, de 0 à 40 ou plus).

Le texte commencera à défiler dès que l'on aura frappé RET.

Si l'on désire changer les conditions du défilement, il suffit d'appuyer sur la touche 0 pour retrouver les questions SENS et VITESSE. On tapera la barre d'espace pour interrompre le programme.

EXTENSIONS POSSIBLES :

On pourra introduire de la couleur, réaliser un affichage ondulant, faire remonter les lignes sur l'écran, etc.

```

1  Rem AFFICHAGE PUBLICITAIRE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Deftext 1,0,0,13+7*(P>0)
6  Print "ENTREZ VOTRE TEXTE"
7  Print At(1,5);"- "
8  Do
9    A$=Upper$(Chr$(Inp(2)))
10   Exit If Asc(A$)=13
11   If Asc(A$)=8 And A$<>" " T
hen
12     B$=Left$(B$,Len(B$)-1)
13   Else
14     B$=B$+A$
15   Endif
16   Print At(1,4);B$;" "
17   Print Tab(Len(B$));"- "
18 Loop
19 B$=" "+B$+" "
20 Do
21   D=B*Len(B$)
22   Deftext 1,0,0,13+7*(P>0)
23   Clearw 2
24   Print At(1,8);"DEFILEMENT
(D)ROITE DU (G)AUCHE :";
25   S$=Upper$(Chr$(Inp(2)))
26   Repeat
27     Print At(1,12);"VITESSE
( DE 0 A 50 ) :";
28     Input V
29     Until V>=0 And V<=50
30     V=51-V
31     R$=B$
32     L=Len(R$)
33     Clearw 2
34     Rbox 20,20,D+80,D+80
35     Rbox 70,70,D+30,D+30
36     Repeat
37       Gosub Tex (48,60,1,0,0)
38       Gosub Tex (D+48,60,0,1,2
700)
39       Gosub Tex (D+48,D+60,-1,
0,1800)
40       Gosub Tex (48,D+60,0,-1,
900)
41       If S$="G" Then
42         R$=Right$(R$,L-1)+Lef
t$(R$,1)
43       Endif
44       If S$="D" Then
45         R$=Right$(R$,1)+Left$(
R$,L-1)
46       Endif
47       For J=0 To 9*V
48         P$=Inkey$
49         Exit If P$<>" "
50       Next J
51       Until P$=" " Or P$="0"
52       Exit If P$=" "
53 Loop
54 Deftext 1,0,0,13
55 For I=1 To 12 Step 2
56   Sound 1,10,I,5,10
57 Next I
58 Sound 1,0
59 Print At(1,18);"C'EST FINI"
;
60 R$=Input$(1)
61 Procedure Tex (B,C,A,D,E)
62   Deftext 1+Random(4^P),0,E
,13+7*(P>0)
63   For I=1+(E=0)*(S$="D") To
Len(R$)
64     Text B+8*A*I,C+8*D*I,Mi
d$(R$,I,1)
65     Sound 1,10,I,5,1
66     Sound 1,0
67   Next I
68 Return

```

Chapeaux

REFERENCES :

Difficulté de programmation : assez facile.

Catégorie de jeu : jeu de réflexe, pour deux joueurs.

Difficulté du jeu : facile.

LE JEU :

L'idée en est très simple : deux personnages se déplacent l'un vers l'autre sur l'écran et se croisent. Au moment où ils se croisent, il est possible qu'ils échangent leurs chapeaux. Il s'agit d'être le premier à dire si oui ou non il y a eu échange de chapeaux lors du croisement.

LE PROGRAMME :

Les lignes 14 et 15 contiennent le motif des deux chapeaux (dans deux variables E et F).

La boucle I (9-12) lit les lignes DATA qui permettront de dessiner les personnages.

La ligne 18 permet, s'il y a lieu, l'échange des chapeaux.

La variable T de la ligne 20 sert à repérer s'il y a eu échange ou non.

Le dessin des deux personnages est réalisé par les lignes 26 à 36, et leur déplacement est réalisé par les lignes 37 et 38.

Le test, permettant ou non le changement des couvre-chefs, est effectué en ligne 45.

Les lignes 42 à 67 déterminent si le premier joueur ayant appuyé sur une touche a gagné ou perdu.

POUR JOUER :

Lancer le programme par RUN.

Les personnages commencent immédiatement à se déplacer l'un vers l'autre.

Le joueur de gauche dispose des touches ESC et 1 et celui de droite, des touches * et /.

S'il y a eu changement de chapeau, les touches à appuyer seront 1 (à gauche) et / (à droite).

S'il n'y a pas eu de changement, il faudra, pour gagner, être le premier à appuyer sur ESC (à gauche) et sur * (à droite).

L'affichage indiquera alors BRAVO GAUCHE ou BRAVO DROITE, ou, encore, PERDU GAUCHE ou PERDU DROITE.

EXTENSIONS POSSIBLES :

Il sera possible de prévoir un affichage couleur, avec une animation plus complexe des personnages.

On pourra aussi compliquer le jeu à plaisir avec plusieurs objets échangés de façon aléatoire (il faudrait alors en trouver la liste complète).

```

1  Rem CHAPEAUX
2  P=Abs(Xbios(4)-2)
3  X=60+42*(P=2)
4  Y=560+302*(P=2)
5  B=189+17*(P>0)
6  Fullw 2
7  Clearw 2
8  Dim X%(32),Y%(32)
9  For I=1 To 31
10   Read X%(I),B%
11   Y%(I)=-B%
12 Next I
13 T=1
14 E=3
15 F=6
16 K=1
17 G=16*(P=0)
18 If Rnd<0.5 Then
19   Swap E,F
20   T=Abs(T-1)
21 Endif
22 Do
23   A=X
24   C=Y
25   Clearw 2
26   Polyline 31,X%(),Y%() Off
27   set A,150
28   Polyline 31,X%(),Y%() Off
29   set C,150
28   Line 0,B-22+G,660,B-22+G
29   Deffill E+1,3,E
30   Fill A+15,B-115+G
31   Deffill F+2,3,F
32   Fill C+15,B-115+G
33   Deffill 2,2,1
34   Fill A+15,B-100
35   Deffill 3,2,14
36   Fill C+15,B-100
37   X=X+K*12
38   Y=Y-K*12
39   If Y<20 Or X<20 Then
40     K=-K
41   Endif
42   Sound 1,10,4,5,5
43   Sound 1,0
44   Pause 40
45   If Abs(X-Y)<10 And Rnd<0.
5   Then
46     T=Abs(T-1)
47     Swap E,F
48   Endif
49   R=Asc(Right$(Inkey$,1))
50   Exit If R=27 Or R=38 Or R
=42 Or R=47
51 Loop
52 If X>Y Then
53   If (R=27 And T=1) Or (R=3
8 And T=0) Then
54     Print "BRAVO GAUCHE"
55     Goto Gagne
56   Endif
57   If (R=47 And T=1) Or (R=4
2 And T=0) Then
58     Print "BRAVO DROIT"
59     Goto Gagne
60   Endif
61 Endif
62 If R=27 Or R=38 Then
63   Print "PERDU GAUCHE"
64 Endif
65 If R=42 Or R=47 Then
66   Print "PERDU DROIT"
67 Endif
68 For I=12 To 1 Step -2
69   Sound 1,10,1,2,10
70 Next I
71 Gagne:
72 For I=1 To 12
73   Sound 1,10,1,5,10
74 Next I
75 Sound 1,0
76 R%=Input$(1)
77 Data 0,0,0,7,7,7,13,55,10,6
5,3,40,0,40,10,70,13,70,12,
77,15,85,0,85,5,100
78 Data 25,100,30,85,15,85,18,
77,17,70,20,70,30,40,27,40,
20,65,17,55,23,7,30,7
79 Data 30,0,20,0,17,40,13,40,
10,0,0,0

```


Clavier

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de réflexe (apprentissage du clavier).

Difficulté du jeu : difficile.

LE JEU :

Une reproduction du clavier de votre ordinateur apparaît sur l'écran ; une des touches apparaît en grisé. Il convient de la frapper au clavier. Plus l'on frappe vite, et plus la touche à frapper se déplace rapidement (le temps écoulé depuis le début s'affiche en fin de partie).

LE PROGRAMME :

Une variable A\$ dimensionnée en ligne 13 contiendra la liste des touches concernées. Les valeurs des touches sont entrées de la ligne 16 à la ligne 35 par appel d'un sous-programme situé en 69-73.

Les boucles emboîtées J et I (36-40) dessinent le clavier.

Les lignes 42 et 43 définissent la position du point imprimé par la ligne 45.

La touche frappée est testée aux lignes 50 à 55 et la touche est redessinée par la ligne 59.

Les affichages de fin sont réalisés aux lignes 61 à 63.

POUR JOUER :

On lance le programme par RUN. L'on dispose alors de quelques secondes pour frapper la touche qui est au-dessus de l'étoile, puis une autre étoile est affichée, etc.

La durée de la partie n'est pas limitée ; il suffit de taper Ø pour obtenir l'affichage du score, et du temps total.

Taper la combinaison CSA pour quitter le programme.

Faire RUN pour une autre partie.

EXTENSIONS POSSIBLES :

On pourra envisager de faire clignoter les lettres, ou de les afficher en couleur, au lieu d'indiquer la lettre par une étoile. On pourra afficher les lettres devinées justes en bas de l'écran, et éventuellement s'entraîner à taper un texte qui défile en haut de l'écran, lettre par lettre.

```

1  Rem CLAVIER
2  P=Abs(Xbios(4)-2)
3  U=48+24*(P=2)
4  V=80+40*(P>0)
5  Deftext 1,,13+7*(P>0)
6  K=2+(P=2)
7  Fullw 2
8  Print "(A)ZERTY OU (Q)WERTY
   "
9  R$=Upper$(Input$(1))
10 T=Timer
11 W=0
12 Z=0
13 Dim A$(10,3)
14 Clearw 2
15 J=1
16 If R$="A" Then
17   S$="AZERTYUIOP"
18 Else
19   S$="QWERTYUIOP"
20 Endif
21 Gosub Tableau
22 J=2
23 If R$="A" Then
24   S$="QSDFGHJKLM"
25 Else
26   S$="ASDFGHJKL;"
27 Endif
28 Gosub Tableau
29 J=3
30 If R$="A" Then
31   S$="WXCVRN,;:="
32 Else
33   S$="ZXCVRNM,./"
34 Endif
35 Gosub Tableau
36 For J=1 To 3
37   For I=1 To 10
38     Gosub Rectangles(I,J,2)
39     Next I
40   Next J
41 Do
42   X=Random(10)+1
43   Y=Random(3)+1
44   Gosub Rectangles(X,Y,8)
45   For F=1 To 700+900*Rnd
46     R$=Upper$(Inkey$)
47     Exit If R$<>" "
48   Next F
49   If R$=A$(X,Y) Then
50     Z=Z+1
51     Sound 1,9,4,5,8
52   Endif
53   Sound 1,0
54   Exit If R$="0" Or R$="a"
55   W=W+1
56   Sound 1,10,7,6,2
57   Sound 1,0
58   Gosub Rectangles(X,Y,2)
59 Loop
60 Print At(1,1);Z;" JUSTES SU
   R ";W;" ESSAIS"
61 Print
62 Print "VOTRE TEMPS : ";Int(
   (Timer-T)/200);" SECONDES"
63 For I=1 To 12
64   Sound 1,10,1,5,10
65 Next I
66 Sound 1,0
67 R$=Input$(1)
68 Procedure Tableau
69   For I=1 To 10
70     A$(I,J)=Mid$(S$,I,1)
71   Next I
72 Return
73 Procedure Rectangles(I,J,C)
74   Deffill 1-(P>0)-(C=8),2,C
75   A=U*I+(16+8*(P=2))*J-54-2
   0*(P=2)
76   B=V*J-20-8*(P>0)
77   Pbox A,B,A+40+16*(P=2),B+
   30+12*(P>0)
78   Print At((3*I+J-2)*K,5*J)
   ;A$(I,J)
79 Return

```

Dessin de dés

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : lancer de dés.

Difficulté du jeu : facile.

LE JEU :

Le programme proposé est surtout intéressant par son côté graphique puisqu'il permet de représenter, de façon très convenable, de 1 à 9 dés sur l'écran, en utilisant une technique de dessin un peu particulière.

Chaque réponse à la question NOMBRE DE DES provoque immédiatement un nouveau lancer du nombre de dés indiqué.

LE PROGRAMME :

La ligne 11 permet de contrôler s'il y a eu ou non action sur une touche de 1 à 9 ou de 1 à 4, selon la taille d'écran.

Les lignes 14 à 17 initialisent les variables de dessin des dés.

Les lignes 19 à 25 dessinent le cadre du dé.

La ligne 26 réalise le tirage, et les sous-programmes 34 à 61 mettent en place les points en relation avec le tirage réalisé.

POUR JOUER :

On lance le programme par l'ordre RUN, et, à la question NOMBRE DE DES, on répond en frappant une touche de 1 à 4, ou de 1 à 9, selon le cas. Les dés demandés sont alors représentés, et la question NOMBRE DE DES est de nouveau posée. Il suffira de taper la combinaison CSA pour sortir du programme.

EXTENSIONS POSSIBLES :

On pourra conserver la partie dessin utilisée dans ce programme pour réaliser toutes sortes de jeux de dés, avec une représentation graphique de qualité convenable et facile à modifier si besoin était.

```

1  Rem DESSIN DE DES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Dim A%(6),B%(6)
6  F=80
7  Do
8      Repeat
9          Print At(1,3);"NOMBRE D
E DES"
10         N=Inp(2)-48
11         Until N>0 And N<=9+5*(P=2
)
12     Clearw 2
13     For I=1 To N
14         X1=70*I-66
15         Y1=80
16         X2=70*I-6
17         Y2=140
18         Co=1+Random(4^P)
19         Deffill Co,2,1-(P=0)
20         Pbox X1,Y1,X2,Y2
21         Color Co
22         Draw X1,Y1 To X1+8,Y1-8
To X2+8,Y1-8 To X2,Y1
23         Fill X1+8,Y1-3
24         Draw X2+8,Y1-8 To X2+8,
Y2-8 To X2,Y2
25         Fill X2+3,Y2-5
26         X=Random(6)+1
27         On X Gosub Tir1,Tir2,Ti
r3,Tir4,Tir5,Tir6
28         Defmark 1,3,20
29         Polymark X,A%(),B%() Of
fset 70*I-64,F
30         Sound 1,10,2*X,5,8
31         Sound 1,0
32     Next I
33 Loop
34 Procedure Tir1
35     A%(0)=30
36     B%(0)=30
37 Return
38 Procedure Tir2
39     A%(0)=10
40     B%(0)=10
41     A%(1)=50
42     B%(1)=50
43 Return
44 Procedure Tir3
45     Gosub Tir2
46     A%(2)=30
47     B%(2)=30
48 Return
49 Procedure Tir4
50     Gosub Tir2
51     A%(2)=10
52     B%(2)=50
53     A%(3)=50
54     B%(3)=10
55 Return
56 Procedure Tir5
57     Gosub Tir4
58     A%(4)=30
59     B%(4)=30
60 Return
61 Procedure Tir6
62     Gosub Tir4
63     A%(4)=10
64     B%(4)=30
65     A%(5)=50
66     B%(5)=30
67 Return

```

Smokey

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : solitaire.

Difficulté du jeu : moyenne.

LE JEU :

Une colonne de fumée grise apparaît sur l'écran. Le joueur, représenté par un symbole %, devient visible en haut de l'écran. Il doit dans un premier temps se positionner sur la colonne de fumée, puis, à partir de là, il va automatiquement avaler la fumée, à condition qu'il réussisse à se maintenir sur sa trace.

LE PROGRAMME :

La ligne 12 fait apparaître la fumée au bas de l'écran ; la ligne 13 permet de changer (vers la droite ou vers la gauche) la direction de la colonne, et la ligne 15 fait monter la colonne.

Le joueur apparaît grâce à la ligne 19, et son déplacement est effectué par les lignes 27 à 29.

Les lignes 44 à 51 se chargent de l'affichage du résultat.

POUR JOUER :

On doit lancer l'exécution du programme par RUN.

La fumée commence à s'élever, tandis que le joueur, apparu sous forme d'un symbole % coloré dans un coin de l'écran, va se déplacer sur elle à l'aide des 4 flèches. Dès qu'il arrive sur la fumée, il va automatiquement commencer à l'avalier, et devra essayer de suivre la colonne sans dévier (grâce aux flèches). Il gagne s'il arrive en bas de l'écran avant que la fumée n'arrive en haut.

L'affichage final indique alors BRAVO ou PERDU, ainsi que le score obtenu.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra, tout en conservant l'idée de départ, imposer d'autres règles : par exemple, avaler toute la fumée, qui pourra aussi repartir d'un autre endroit si l'on dévie de la colonne initiale.

On pourra aussi améliorer les contrôles de fin de partie, et la qualité du graphisme.

```

1  Rem SMOKEY
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Rp=37
5  Z=0
6  M=25
7  N=Random(15)+7
8  U=1
9  V=15
10 Debut:
11 Deftext 2
12 Print At(N,M);Chr$(42);
13 N=N-(Rnd<0.5 And N<24)+(Rnd<0.5 And N>1)
14 T=1
15 M=M-T
16 If M<>0 Then
17   Suite:
18   Deftext 3
19   Print At(V,U);Chr$(37)
20   If T<>0 Then
21     For I=1 To 900
22       R$=Right$(Inkey$,1)
23       Exit If R$<>""
24     Next I
25   Endif
26   Print At(V,U);Chr$(32)
27   U=U-(R$="P" And T=1 And M<24)
28   U=U-(T=0)
29   V=V-(R$="M")+(R$="K")
30   Et=Point(8*V-3,(16+8*(P>0))*(U+1)-3)
31   If Et=0 And U<23 Then
32     Goto Debut
33   Endif
34   T=0
35   Z=Z+1
36   Sound 1,10,Z,5,2
37   Sound 1,0
38   If U>22 Then
39     Goto Fin
40   Else
41     Goto Suite
42   Endif
43 Endif
44 Fin:
45 Deftext 1
46 If Z>8 Then
47   Print At(1,18);"BRAVO..."
48 Else
49   Print At(1,18);"PERDU..."
50 Endif
51 Print "SCORE ";Z
52 For I=1 To 12
53   Sound 1,10,I,5,10
54 Next I
55 Sound 1,0
56 R$=Input$(1)

```

Captures dans l'espace

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexe (1 seul joueur).

Difficulté du jeu : moyenne.

LE JEU :

Votre vaisseau spatial navigue dans un dangereux champ de météorites et votre mission est de vous emparer de la plus grande quantité possible de minerai d'urane, un métal indispensable que l'on ne peut trouver que dans ces lieux particulièrement dangereux.

Vous devez arriver à manœuvrer pour recueillir l'urane, sans vous faire heurter par une météorite.

LE PROGRAMME :

La ligne 12 permet de faire apparaître les météorites.

La ligne 15 fera apparaître le minerai d'urane. La ligne 21 assure le déplacement des météorites sur l'écran.

Les lignes 22 à 35 contrôlent le déroulement du jeu (a-t-on heurté une météorite ou a-t-on capturé un bloc d'urane ?).

La rencontre d'une météorite provoque la fin de partie.

Votre vaisseau spatial est imprimé par la ligne 37 et son déplacement est contrôlé par la ligne 40.

POUR JOUER :

On lance le programme par RUN et les météorites commencent à monter sur l'écran, ainsi que les blocs d'urane.

Les touches de déplacement horizontal (→ et ←) seront utilisées pour déplacer votre vaisseau et récupérer l'urane, mais le moindre heurt avec une météorite provoquera la fin de la partie et l'affichage de la quantité d'urane capturée.

On pourra aussi arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut rendre le jeu de plus en plus difficile en augmentant progressivement le nombre de météorites, ou encore, en minutant la durée de la partie.

On peut aussi prévoir un affichage permanent du score, introduire un deuxième vaisseau spatial manœuvré par une puissance étrangère et qui cherche aussi à s'emparer de l'urane...

```

1  Rem CAPTURES DANS L'ESPACE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  W=20
5  U=W
6  Z=0
7  Clearw 2
8  A=77+39*(P=2)
9  Do
10   H=Random(A)+1
11   Deffill 1+Random(4^P),1,1
12   Pbox 8*H+2,320+160*(P>0),8*H+24,345+172*(P>0)
13   K=Random(A)+1
14   If Rnd<0.8 Then
15     Box 8*K+4,320+160*(P>0),8*K+20,345+172*(P>0)
16   Endif
17   Deftext 1,0,0,13+7*(P>0)
18   Sound 1,10,Random(10),5,1
19   Sound 1,0
20   Print At(1,22)
21   Print At(1,23);Spc(80+40*(P=2));
22   E=8*W+4
23   B=Point(E,166+75*(P>0))
24   E=E+B
25   D=Point(E,166+75*(P>0))
26   E=8*W-(P=1)
27   C=Point(E,166+75*(P>0))
28   E=E+B
29   F=Point(E,166+75*(P>0))
30   Exit If B<>0 Or D<>0
31   If B+C+D+F<>0 Then
32     Z=Z+1
33     Sound 1,10,8,6,4
34     Sound 1,0
35   Endif
36   Print At(U,9);Space$(4)
37   Print At(W+1,11);Chr$(157);Chr$(157)
38   S$=Right$(Inkey$,1)
39   U=W
40   W=W-((S$="M")-(S$="K"))
41 Loop
42 Print At(1,1);"SCORE OBTENU :";Z
43 For I=1 To 12
44   Sound 1,10,I,5,10
45 Next I
46 Sound 1,0
47 R$=Input$(1)

```


Danger spatial

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de réflexe, pour un joueur.

Difficulté du jeu : moyenne.

LE JEU :

Vous êtes aux commandes d'un engin spatial qui vient de pénétrer dans un nuage de météorites. Vous devez essayer de franchir le nuage sans heurter une seule météorite. Votre score sera affiché en fin de partie, lorsque vous entrez en collision avec une météorite.

LE PROGRAMME :

La position de chaque météorite est déterminée par les lignes 15 à 29.

L'avance des météorites est réalisée par la ligne 31.

La ligne 39 vérifie si vous avez ou non heurté une météorite.

La ligne 32 dessine votre vaisseau spatial, et la ligne 38 permet de le déplacer.

Le compteur se trouve en ligne 35.

POUR JOUER :

Le jeu est lancé par l'ordre RUN.

Votre vaisseau apparaît vers le milieu de l'écran, et les météorites viennent à votre rencontre en partant du bas de l'écran. Pour vous déplacer, vous pouvez utiliser les touches de déplacement horizontal (→ et ←).

On peut arrêter la partie à tout moment en tapant la combinaison CSA. Sinon, le jeu s'arrête lorsque vous heurtez une météorite, et le score est alors affiché.

EXTENSIONS POSSIBLES :

Il serait intéressant de rendre le jeu de plus en plus difficile en faisant apparaître de plus en plus de météorites au fur et à mesure du déroulement de la partie.

On peut aussi essayer de détruire les météorites en les pulvérisant au fur et à mesure de leur apparition.

```

1  Rem DANGER SPATIAL
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  A$=Mki$(1)+Mki$(1)+Mki$(0)+Mki$(0)+Mki$(1)
5  For I=1 To 16
6      Read H
7      A$=A$+Mki$(H)+Mki$(H)
8  Next I
9  L=300+150*(P>0)
10 A=77+39*(P=2)
11 F=0
12 Z=0
13 Clearw 2
14 Do
15     H=Random(A)
16     If Rnd<0.6 Then
17         Deffill 1+Random(4^P),2,16
18         Pbox 8*H,336+190*(P>0),8*H+16,352+198*(P>0)
19     Endif
20     M=Random(A)
21     If Rnd<0.6 Then
22         Deffill 1+Random(4^P),2,8
23         Pbox 8*M+60,336+190*(P>0),8*M+76,352+198*(P>0)
24     Endif
25     K=Random(A)
26     If Rnd<0.6 Then
27         Deffill 1+Random(4^P),2,23
28         Pbox 8*K+60,336+190*(P>0),8*K+76,352+198*(P>0)
29     Endif
30     Sprite A$
31     Print At(1,22)
32     Sprite A$,L+50,60+30*(P>0)
33     R$=Right$(Inkey$,1)
34     If R$<>" " Then
35         Z=Z+1
36     Endif
37     Sound 1,10,Random(8),1,1
38     L=L-8*((R$="M")-(R$="K"))
39     Eo=Point(L+52,46+23*(P>0))+Point(L+62,46+23*(P>0)
40 ))
41 Exit If Eo<>0
41 F=F+1
42 Loop
43 Print At(1,17);"C'EST FINI ... "
44 Print "VOTRE SCORE EST DE ";Z;" SUR ";F
45 For I=1 To 12
46     Sound 1,10,I,5,10
47 Next I
48 Sound 1,0
49 R$=Input$(1)
50 Data &h801,&hc03,&h606,&h30c,&h381c,&h1c38,&h1c38,
    &h7ffe
51 Data &h7ffe,&h1c38,&h1c38,&h381c,&h30c,&h606,&hc03
    ,&h801

```

Course de voitures

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de réflexe (1 joueur).

Difficulté du jeu : moyenne.

LE JEU :

Vous circulez sur une route très sinueuse, ne comportant aucun obstacle, et vous devez essayer de la suivre le plus longtemps possible. La trace de votre parcours reste affichée en permanence, et votre score est affiché dès que vous avez un accident.

LE PROGRAMME :

L'impression de la route est assurée par les lignes 17 et 18.

La ligne 20 fait avancer la route, et la ligne 21 imprime la voiture.

La ligne 27 contrôle son mouvement, et la ligne 28 détermine si la route fait ou non des virages.

Les lignes 29 et 30 regardent si l'on a touché le bord de la route.

La ligne 31 incrémente le score.

La ligne 35 affiche le temps avant l'accident.

POUR JOUER :

Lancer l'exécution du programme par l'ordre RUN, le dessin de la route commence à évoluer, et votre véhicule, représenté par une voiture, apparaît.

Pour le déplacer vers la droite ou vers la gauche, vous devrez utiliser les touches → et ← (flèches de déplacement du curseur). Il vous faudra éviter le plus longtemps possible de toucher une bordure noire, ce qui provoquera l'arrêt du programme et l'affichage de la mention ACCIDENT, suivie de l'indication sur le score réalisé.

On pourra arrêter le déroulement de la partie à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager de jouer à plusieurs, chacun son tour, en enregistrant le meilleur score.

On peut aussi envisager d'élargir la piste, et de faire courir plusieurs voitures.

On pourra aussi s'inspirer des jeux d'arcades qui sont particulièrement nombreux sur ce thème.

```

1  Rem COURSE DE VOITURE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  A$=Mki$(1)+Mki$(1)+Mki$(0)+Mki$(0)+Mki$(1)
5  For I=1 To 16
6    Read H
7    A$=A$+Mki$(H)+Mki$(H)
8  Next I
9  T=Timer
10 V=11
11 B=10
12 Z=-14
13 Deffill 1,2,8
14 Deftext 1,0,0,4+2*(P>0)
15 Clearw 2
16 Do
17   Pbox 8*V,354+190*(P>0),8*V+16,360+190*(P>0)
18   Pbox 8*V+60,354+190*(P>0),8*V+76,360+190*(P>0)
19   Sprite A$
20   Print At(1,72)
21   Sprite A$,8*B+50,60+30*(P>0)
22   For I=1 To 50
23     R$=Right$(Inkey$,1)
24     Exit If R$<>" "
25   Next I
26   Sound 1,10,Random(8),1,1
27   B=B-(R$="M")+(R$="K")
28   V=V-(-2*Rnd And V<25)+(-2*Rnd And V>3)
29   Ea=Point(8*B+54,46+23*(P>0))
30   Exit If Ea<>0
31   Z=Z+1
32 Loop
33 Deftext 1,0,0,13+7*(P>0)
34 Print At(20,8);"ACCIDENT. SCORE ";Z
35 Print At(26,10);"TEMPS ";Int((Timer-T)/200)
36 For I=1 To 12
37   Sound 1,10,I,5,10
38 Next I
39 Sound 1,0
40 R$=Input$(1)
41 Data &h240,&h37ec,&h37ec,&h3ffc,&h37ec,&h37ec,&h7e
   0,&h5a0
42 Data &h5a0,&h660,&h3c0,&hbd0,&hff0,&hbd0,&h3c0,&h1
   80

```

Obstacles

REFERENCES :

Difficulté de programmation : assez difficile.

Catégorie de jeu : jeu de réflexe (pour un joueur).

Difficulté du jeu : moyenne.

LE JEU :

Des piétons se déplacent sur l'écran pour venir à la rencontre d'une petite voiture qui dispose de la possibilité de se déplacer vers la gauche ou vers la droite pour les éviter et qui devra tenir le plus longtemps possible.

La distance parcourue sera affichée en permanence, et le jeu se termine lorsque la voiture rencontre, soit le bord du terrain, soit un piéton.

LE PROGRAMME :

La ligne 32 fait avancer les obstacles vers la voiture du joueur qui est imprimée grâce à la ligne 34.

Les lignes 21 à 30 fabriquent la route et les piétons.

La ligne 33 affiche le score.

La ligne 39 permettra de déplacer le véhicule.

Les lignes 40 et 41 permettent de contrôler s'il y a ou non un obstacle et arrêtent le programme s'il y a lieu.

Le score est incrémenté en ligne 42.

POUR JOUER :

On lance le programme en tapant RUN.

La route et les piétons apparaissent en bas de l'écran et montent vers la voiture. La voiture peut se déplacer à l'aide des flèches de déplacement horizontal (touches → et ←).

La rencontre avec un piéton ou la bordure interrompt le déroulement du programme. L'affichage du score est effectué en permanence.

On peut, cependant, sortir du programme à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera assez facile d'améliorer le dessin de la voiture, de la route, des piétons. On pourra améliorer le son et la couleur, admettre un certain nombre d'accidents, enregistrer le meilleur score (pour plusieurs joueurs), éventuellement dessiner plusieurs voitures, etc.

```

1  Rem OBSTACLES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Dim A$(3)
5  For I=1 To 3
6    A$(I)=Mki$(1)+Mki$(1)+Mki$(0)+Mki$(0)+Mki$(I-I*(
   P=2))
7    For J=1 To 16
8      Read H
9      A$(I)=A$(I)+Mki$(H)+Mki$(H)
10     Next J
11  Next I
12  Sprite A$(2),0,40
13  Sprite A$(3),17,40
14  Get 1,40,16,56,A$(2)
15  Get 17,40,33,56,A$(3)
16  Z=0
17  B=5
18  Clearw 2
19  Deffill 1,2,8
20  Do
21    Pbox 0,336+168*(P>0),16,368+168*(P>0)
22    Pbox 152,336+168*(P>0),168,358+168*(P>0)
23    If Rnd<0.7 Then
24      X=Random(8)+1
25      Put 16*X+8,336+168*(P>0),A$(2),3
26    Endif
27    Y=Random(8)+1
28    If Rnd<0.7 And Y<>X Then
29      Put 16*Y+8,336+168*(P>0),A$(3),3
30    Endif
31    Sprite A$(1)
32    Print At(1,22)
33    Print At(24,1);"SCORE :";Z
34    Sprite A$(1),16*B+8,60
35    For I=1 To 200
36      R$=Right$(Inkey$,1)
37      Exit If R$<>" "
38    Next I
39    B=B-(R$="M" And B<9)+(R$="K" And B>0)
40    Eo=Point(16*B+12,60)
41    Exit If Eo<>0
42    Z=Z+1
43  Loop
44  For I=1 To 12
45    Sound 1,10,I,5,10
46  Next I
47  Sound 1,0
48  R$=Input$(1)
49  Data &h240,&h37ec,&h37ec,&h37ec,&h3ffc,&h37ec,&h7e
   0,&h5a0,&h5a0,&h660,&h3c0
50  Data &hbd0,&hff0,&hbd0,&h3c0,&h180,&h7f0,&h7f0,&h1
   ffc,&h7d0,&h7f8
51  Data &h3e0,&h1c0,&h6b0,&hf78,&h1ffc,&h3b6e,&h33e6,
   &h3e0,&h770
52  Data &he38,&hf1c,&h0,&h3c0,&hf40,&h1be0,&h1be0,&h1
   980,&h7e0
53  Data &hff0,&h1bd8,&h33cc,&h7e0,&h7e0,&hff0,&h240,&
   h240,&h360

```

Poker

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu contre le ST.

Difficulté du jeu : facile.

LE JEU :

C'est une simulation d'une partie de poker traditionnelle, effectuée avec un jeu de 32 cartes, dans laquelle le joueur aura pour partenaire le ST.

Le joueur recevra 5 cartes, repérées par des numéros de 1 à 5, et pourra décider de changer certaines cartes en annonçant celles qu'il écarte. Sa nouvelle donne sera alors affichée, et c'est ensuite au tour du ST de jouer.

LE PROGRAMME :

Les lignes de DATA (70 à 76) contiennent les dessins des symboles des cartes. Une variable N est dimensionnée à 16, en ligne 7, pour recevoir 10 cartes joueur et 5 cartes ST.

Le tirage sera effectué par une boucle I (26-31) et le contrôle par une table T, dimensionnée en ligne 7.

L'affichage des cartes est réalisé par le sous-programme 53-69.

Une autre boucle I (38-40) réalise le deuxième tirage du joueur, la ligne 42 permettant le deuxième affichage, et la ligne 44 où l'on arrive directement après, réalisera l'affichage du jeu machine.

POUR JOUER :

L'ordre RUN lance l'exécution du programme, et les 5 premières cartes du joueur sont alors affichées.

A la question CARTES REFUSEES, le joueur peut répondre 0 suivi de RET s'il veut arrêter la partie. Pour jouer, il répondra simplement RET s'il s'estime servi, ou il entrera les numéros des cartes qu'il refuse, suivis de RET.

Par exemple, s'il tape 235 et RET, son nouveau jeu conservera les cartes 1 et 4. Son jeu sera alors affiché ainsi que celui du ST, ce dernier n'ayant pas, dans ce programme, la possibilité de reprendre des cartes.

Il suffira ensuite de taper une touche quelconque pour démarrer une autre partie.

EXTENSIONS POSSIBLES :

Elles seront, bien sûr, très nombreuses, depuis un affichage graphique en couleur des cartes jusqu'à l'amélioration du jeu du ST, l'introduction des règles sur les paris et même, pourquoi pas, des possibilités de bluff.

```

1  Rem POKER
2  P=Abs(Xbios(4)-2)
3  U=7-7*(P<2)
4  Ta=16+8*(P>0)
5  Fullw 2
6  Openw 2
7  Dim N(16),T(33),A$(40),C(4)
8  ,P(4,10)
9  V$=" 7 8 9 10 VAL
ETDAME ROI AS "
10 For I=1 To 4
11   Co=1+(I<3)*(P>0)
12   Sp$=""
13   For J=1 To 16
14     Read A
15     Sp$=Sp$+Mki$(A)+Mki$(A)
16   Next J
17   For K=0 To 9
18     A$(I+4*K)=Mki$(1)+Mki$(
19       1)+Mki$(0)+Mki$(0)+Mki$(Co)
20     +Sp$
21   P(I,K)=I+4*K
22 Next K
23 Next I
24 Repeat
25   Clearw 2
26   For I=1 To 32
27     T(I)=0
28   Next I
29   For I=1 To 16
30     Repeat
31       N(I)=Random(32)
32     Until T(N(I))=0
33     T(N(I))=1
34   Next I
35   A=0
36   B=0
37   Print At(1,10);"CARTES RE
38   FUSEES : "
39   Gosub Tir
40   Print At(18,10);
41   Input R$
42   For I=1 To Len(R$)
43     N(Val(Mid$(R$,I,1)))=N(
44       I+5)
45   Next I
46   B=1
47   Gosub Tir
48   A=A+10
49   Gosub Tir
50   Print At(1,10);"SUITE ?
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76

```


Course de chevaux

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : hasard (plusieurs joueurs).

Difficulté du jeu : aucune.

LE JEU :

On entrera un nombre de chevaux compris entre 1 et 9, et la course débutera immédiatement.

Il pourra y avoir des ex aequo, car l'on tiendra compte du fait que le ST s'occupe de faire avancer les chevaux les uns après les autres, en commençant par le numéro 1 ; le classement sera affiché au fur et à mesure des arrivées (et en tenant compte des ex aequo).

LE PROGRAMME :

Les boucles I (10-13) et (14-16) permettent de définir des SPRITES pour le dessin des chevaux.

Une boucle non formelle (28-47) affiche les chevaux dans leur position de départ.

Une deuxième boucle I (30-56) les fera avancer ; la position de chaque cheval sera repérée par un chiffre ou une lettre dans une variable alphanumérique A\$.

La ligne 34 fera avancer les chevaux.

La ligne 41 les affiche.

La ligne 37 gère les arrivées.

Le sous-programme ARRIVE (69-76) fait avancer le compteur des arrivées, et imprime les résultats.

POUR JOUER :

On lance le programme par RUN.

A la question NOMBRE DE CHEVAUX : on répond par un entier de 1 à 9, suivi de RET.

La course se déroulera ensuite automatiquement.

On devra taper la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra dessiner les chevaux avec un graphisme plus soigné, et les différencier par des numéros.

On pourra aussi imaginer tout un système de paris, qui interviendra avant le départ de chaque course.

```

1  Rem COURSE DE CHEVAUX
2  Fullw 2
3  Repeat
4    Clearw 2
5    Input "NOMBRE DE CHEVAUX
   :";N
6  Until N<11
7  Dim A$(N),X(N),Y(N)
8  Pe=32+16*(Xbios(4)<2)
9  Sp$=""
10 For I=1 To 16
11   Read A
12   Sp$=Sp$+Mki$(A)+Mki$(A)
13 Next I
14 For I=1 To N
15   A$(I)=Mki$(1)+Mki$(1)+Mki$(0)+Mki$(0)+Mki$(I)+Sp$
16 Next I
17 A$=""
18 B$=""
19 F$=Chr$(83)
20 P=0
21 A=0
22 T=1
23 K=0
24 J=T
25 Clearw 2
26 Deffill N+1,2,3
27 Fill 2,2
28 Debut:
29 A$=A$+"0"
30 I=J
31 Goto Affiche
32 Avance:
33 If Mid$(A$,I,1)<>F$ Then
34   B$=Chr$(Asc(Mid$(A$,I,1))+Random(4)+1)
35   Affiche:
36   Mid$(A$,I,1)=B$
37   If B$>=F$ Then
38     Gosub Arrive
39   Endif
40 Endif

41 Sprite A$(I),(16+8*(Xbios(4)=0))*Asc(Mid$(A$,I,1))-46)+Pe,Pe*I+Pe
42 Sound 1,10,I,5,7
43 Sound 1,0
44 If K=0 Then
45   J=J+1
46   If J<=N Then
47     Goto Debut
48   Else
49     K=1
50     I=1
51     Goto Avance
52   Endif
53 Endif
54 I=I+1
55 If I<=N Then
56   Goto Avance
57 Endif
58 P=P+T
59 T=0
60 If A<N Then
61   I=1
62   Goto Avance
63 Endif
64 For I=1 To 12 Step 2
65   Sound 1,10,I,5,8
66 Next I
67 Sound 1,0
68 R$=Input$(1)
69 Procedure Arrive
70   Mid$(A$,I,1)=F$
71   Print At(1,2*I+1);Chr$(48+P);
72   T=1
73   Sound 1,10,T,5,12
74   Sound 1,0
75   A=A+T
76 Return
77 Data 0,0,0,2,3,3,51175,1228
   4,16376,16344,4092,7704
78 Data 13852,27208,27756,5242
   8

```

Sous-marin

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexe (1 seul joueur).

Difficulté du jeu : facile.

LE JEU :

Vous manœuvrez en surface un chasseur de sous-marins, et des sous-marins passent très régulièrement en dessous de votre unité, à différentes profondeurs.

Vous pouvez, à tout moment, larguer des grenades sous-marines et vous pouvez même envisager de toucher plusieurs fois le même sous-marin.

Votre score sera affiché en permanence, sur la gauche de l'écran (rapport nombre de coups au but sur nombre de sous-marins apparus).

LE PROGRAMME :

La boucle J (4-11) dessine des SPRITE.

La ligne 20 permet au ST de choisir la profondeur du sous-marin.

La ligne 25 dessine votre bâtiment.

Les lignes 14 à 17 dessinent la surface de l'eau.

La ligne 26 dessine le sous-marin.

Le tir est effectué par un programme spécifique placé aux lignes 49 à 59.

Le déplacement (très rapide) de votre navire est réalisé par les lignes 42 et 45.

Le test, permettant de savoir si l'on a touché ou non, est effectué en ligne 29.

La ligne 39 s'occupe de la fin de la partie.

POUR JOUER :

Taper RUN pour commencer la partie.

Le bâtiment sera déplacé grâce aux touches 4 et 6.

La touche "Espace" autorise le tir.

Le navire, se déplaçant trois fois plus vite que les sous-marins, il sera possible de toucher plusieurs fois le même bâtiment.

Pour terminer la partie, il conviendra de taper la combinaison CSA ou d'attendre 99 sous-marins.

EXTENSIONS POSSIBLES :

Il sera assez élémentaire d'améliorer la qualité graphique du jeu, d'utiliser la couleur et les sons, de faire apparaître plusieurs sous-marins se déplaçant à des vitesses différentes, éventuellement même de permettre aux sous-marins de tirer...

```

1  Rem SOUS-MARIN
2  P=Abs(Xbios(4)-2)
3  Dim A$(3)
4  For J=1 To 3
5      Sp$=""
6      For I=1 To 16
7          Read A
8          Sp$=Sp$+Mki$(A)+Mki$(A)
9      Next I
10     A$(J)=Mki$(1)+Mki$(1)+Mki$(0)+Mki$(0)+Mki$(1+Random(4^P))+Sp$
11 Next J
12 G=0
13 C=40+20*(P=2)
14 Line 0,59,660,59
15 Line 0,60,660,60
16 Deffill 1,2,4
17 Fill 2,2
18 Do
19     T=78+39*(P=2)
20     S=Random(7)+12
21     G=G+0.01
22     Debut:
23     X=11+4*(P=1)
24     Affiche:
25     Sprite A$(1),8*C,48
26     Sprite A$(2),8*T,8*S
27     Print At(1,1);Using "SCOR
E : ###.###",G
28     Y=Y+(Y-C)*(X=11+4*(P=1))
29     If S=X And (T-1=Y Or T=Y
Or T+1=Y) Then
30         G=G+1
31         For L=1 To 4
32             Text 8*Y,8*X+8,"***"
33             Sound 1,10,3*L,5,6
34             Sound 1,0
35             Text 8*Y,8*X+8," "
36         Next L
37     Endif
38     T=T-1
39     Exit If P>99
40     If T>=0 Then
41         For I=1 To 50
42             R$=Inkey$
43             Exit If R$<>""
44         Next I
45         C=C-3*((R$="6" And C<70
)-(R$="4" And C>1))
46         If R$<>" " And X=11+4*(
P=1) Then
47             Goto Affiche
48         Endif
49         Sprite A$(3),8*Y,8*X
50         Sound 1,10,7,6,2
51         Sound 1,0
52         X=X+1
53         If X>S Then
54             Sprite A$(3)
55             Sprite A$(2)
56             Goto Debut
57         Else
58             Goto Affiche
59         Endif
60     Endif
61 Loop
62 Print At(1,17);"C'EST FINI.
.."
63 For I=1 To 12
64     Sound 1,10,I,5,10
65 Next I
66 Sound 1,0
67 R$=Input$(1)
68 Data 0,0,0,16,16,16,248,115
,49267,65534,32766,16382,81
88,0,0,0
69 Data 0,0,0,0,0,0,0,48,48,10
17,32767,4095,2041,480,0,0
70 Data 0,0,0,1168,672,992,448
,992,992,992,448,448,128,12
8,0,0

```

Horloge 1

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : dessin.

Difficulté du jeu : aucune.

LE JEU :

Transformer votre écran de télévision en horloge numérique à affichage grand format, grâce à ce programme qui vous permettra d'animer une vitrine, de faire une horloge fantaisie, et qui pourra se prêter à bien d'autres usages.

LE PROGRAMME :

La ligne 11 vous permettra d'entrer l'heure sous forme d'un entier de 4 chiffres (par exemple : 1024 signifiera 10 heures 24 minutes).

Les lignes 24 à 36 réalisent le dessin, en décalant l'impression au fur et à mesure (à l'aide de la variable K, incrémentée en ligne 35).

Les lignes 37 à 43 assurent la progression de l'heure.

Les lignes 45 à 55 effectuent l'affichage de deux points clignotants pour les secondes.

On pourra régler l'horloge en modifiant la valeur 164, en ligne 46.

POUR JOUER :

Dès le lancement par RUN, le programme se place en attente de l'heure ; on tape et on attend le top pour appuyer sur RET.

L'heure s'affiche alors assez rapidement et l'horloge est en route. Pour l'arrêter, il faudra taper la combinaison CSA.

L'heure sera entrée sous forme d'un entier ayant impérativement 4 chiffres.

Pour 5 heures 6 minutes, on veillera à taper 0506 suivi de RET.

EXTENSIONS POSSIBLES :

Il sera intéressant de modifier le programme pour en faire un réveil, avec alarme ; on peut aussi améliorer le graphisme, introduire la couleur.

```

1  Rem HORLOGE 1
2  For I=0 To Xbios(4)
3    Read A,B,C,D
4  Next I
5  Dpoke Windtab+16,A
6  Dpoke Windtab+18,B
7  Dpoke Windtab+20,C
8  Dpoke Windtab+22,D
9  Repeat
10   Clearw 2
11   Input "HEURE ( FORME XXXX ) :";A$
12   Openw 2
13   Until Len(A$)=4 And Val(Left$(A$,2))<=24 And Val(Right$(A
$,2))<=59
14   Dim C$(2),B$(2)
15   B$(1)=Space$(3)
16   B$(0)=""
17   C$(0)=""
18   C$(1)=" "
19   Do
20     Deffill Random(4^(2-Xbios(4)))+1,2,5
21     Clearw 2
22     Fill 20,20
23     K=1
24     While K<5
25       A=Val(Mid$(A$,K,1))
26       Print At(6*K-2,1);B$(-(A<>1 And A<>4))
27       For J=1 To 5
28         Print At(6*K-2,J);C$(Abs(A=0 Or (A>3 And A<>7)))
29         Print At(6*K,J);C$(-(A<>5 And A<>6))
30         Print At(6*K-2,J+4);C$(-(A=0 Or A=2 Or A=6 Or A=8))
31         Print At(6*K,J+4);C$(-(A<>2))
32       Next J
33       Print At(6*K-2,5);B$(-(A>1))
34       Print At(6*K-2,9);B$(-(A<>1 And A<>4 And A<>7))
35       K=K+1
36     Wend
37     A=1+Val(A$)
38     A=A+A*(A=2360)
39     A$="000"+Str$(A)
40     A$=Mid$(A$,Len(A$)-3,4)
41     If Right$(A$,2)="60" Then
42       A$=Right$(Str$(Val(Left$(A$,2))+1),2)+"00"
43     Endif
44     H=1
45     For J=1 To 122
46       For I=1 To 164
47         Print At(14,6);
48         If H=1 Then
49           Print " ":"
50         Else
51           Print " "
52         Endif
53       Next I
54       H=Abs(H-1)
55     Next J
56     Sound 1,10,4,5,5
57     Sound 1,0
58   Loop
59   Data 28,30,230,130,190,50,290,170,200,50,230,190

```

Echiquier

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : graphisme.

Difficulté du jeu : variable, selon initialisation.

LE JEU :

Le programme proposé vient en appoint du programme Echecs. Il donne, en effet, la possibilité de réaliser un affichage graphique de bonne qualité pour des jeux utilisant un damier. Le nombre de cases et les couleurs sont choisis avec des menus déroulants.

LE PROGRAMME :

La boucle I (6-8) lit les lignes de menu (DATA des lignes 43 à 48).

La boucle principale (13-25) regarde les choix effectués dans les menus déroulants.

Les lignes 15 à 17 déterminent les choix.

La ligne 19 envoie dans le sous-programme de dessin (26-42).

La fin de partie est traitée en ligne 27.

POUR JOUER :

L'ordre RUN provoque l'affichage du menu en haut de l'écran.

Pour utiliser le programme, il conviendra de cliquer une taille dans le menu TAILLE, puis une première couleur dans le menu COUL1, et une autre dans COUL2.

On dessine l'échiquier en cliquant sur dessin dans le menu ACTION.

On quittera le programme en cliquant sur fin.

Nota : le choix des couleurs dépend évidemment du mode écran utilisé.

EXTENSIONS POSSIBLES :

On pourra utiliser ce programme pour toute sorte de jeux se déroulant sur un damier (échecs, dames anglaises, le loup et les agneaux, etc.).

```

1  Rem CHOIX D'UN ECHQUIER
2  Fullw 2
3  Clearw 2
4  P=Xbios(4)
5  Dim A$(46)
6  For I=0 To 43
7      Read A$(I)
8  Next I
9  H=3
10 Car=0
11 Fon=1
12 Menu A$()
13 Do
14     On Menu
15     D=Menu(0)
16     R=Val(Left$(Str$(D),1))
17     S=Val(Right$(Str$(D),1))-1
18     Menu Off
19     If R=4 Then
20         Gosub Dessin
21     Endif
22     H=H+(H-S-3)*(R=1)
23     Car=Car+(Car-S)*(R=2)
24     Fon=Fon+(Fon-S)*(R=3)
25 Loop
26 Procedure Dessin
27     If S<>1 Then
28         Clearw 2
29         A=156-8*H
30         B=92-8*H
31         For I=0 To H*H-1
32             J=I Mod H
33             K=Int(I/H)
34             C=Car*((J+K) Mod 2)+Fon*((J+K+1) Mod 2)
35             Deffill C,2,4-4*Even(K+J)
36             Box A+16*J,B+16*K,A+16*J+16,B+16*K+16
37             Pbox A+16*J+1,B+16*K+1,A+16*J+14,B+16*K+14
38         Next I
39     Else
40         End
41     Endif
42 Return
43 Data DAMES,"---"," "," "," "," "," "," "," "," "
44 Data TAILLE,"----"," 4"," 5"," 6"," 7"," 8","
45     9"," 10"," "
46 Data COUL1,"-----"," blanc"," noir"," rouge","
47     vert"," bleu"
48 Data " bleu clair"," jaune","","COUL2,"-----","
49     blanc"," noir"," rouge"
50 Data " vert"," bleu"," bleu clair"," jaune","
51     ,ACTION,"-----"
52 Data " dessin"," fin",""," "

```


L'électricien fou

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : solitaire.

Difficulté du jeu : assez difficile.

LE JEU :

Un immeuble comportant 10 fenêtres, numérotées de 0 à 9, apparaît sur l'écran. Le joueur devra à chaque essai manœuvrer l'un des interrupteurs, en indiquant son numéro (de 0 à 9).

En principe, la fenêtre correspondante devrait s'éteindre, mais il est aussi possible qu'une autre s'allume (ou s'éteigne). Il s'agit, bien sûr, d'essayer d'obtenir l'extinction des 10 fenêtres, avec le moins possible d'essais.

LE PROGRAMME :

Une table A, en ligne 8, contiendra l'état (allumé ou éteint) des 10 fenêtres.

La ligne 5 permettra le calcul des positions de remplissage.

La boucle I (12-14) et l'autre boucle I (17-28) dessinent l'immeuble dans son état initial.

La ligne 29 regarde si tout est éteint, auquel cas l'on ira en ligne 48 afficher le nombre d'essais.

La ligne 30 attend l'indication de l'interrupteur à manœuvrer.

Les lignes 32 à 45 définissent le résultat de la manœuvre.

POUR JOUER :

RUN provoque l'affichage de l'immeuble, avec toutes les fenêtres allumées.

A la question INTERRUPTEUR NUMERO, on répond en indiquant le numéro de l'interrupteur à manœuvrer, suivi de RET.

Le résultat de la manœuvre est alors affiché sur l'écran (des fenêtres s'allument ou s'éteignent).

On continue ainsi jusqu'à ce que toutes les fenêtres soient éteintes, et l'on obtient alors l'affichage du nombre d'essais réalisés.

On devra taper la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra envisager un graphisme très amélioré avec si possible l'utilisation de la couleur. On pourra aussi définir un jeu à deux joueurs, augmenter le nombre de fenêtres, modifier la formule définissant l'allumage ou l'extinction...

```

1  Rem L'ELECTRICIEN FOU
2  Fullw 2
3  Clearw 2
4  Alert 0,"ECRAN      : 1:COULEUR 2:NOIR      ",0,"
   1 : 2 ",Re
5  Deffn A(I)=8*I*Re
6  M=0
7  Option Base 0
8  Dim A(11)
9  Clearw 2
10 Deffill 2
11 Fill 1,1
12 For I=1 To 8
13   Print At(5,I);Space$(Re*24)
14 Next I
15 Do
16   T=0
17   For I=1 To 10
18     T=T+A(I)
19     Sound 1,10,4,4,5
20     Sound 1,0
21     If A(I)=0 Then
22       Deffill 1,0,0
23     Else
24       Deffill 3,2,8
25     Endif
26     Pbox 2*Fn A(I)+30,30,2*Fn A(I)+30+10*Re,30+10*
Re
27     Print At(2*Re*I+5,6);I-1
28   Next I
29   Exit If T=10
30   Print At(5,12+2*(Re=1));"INTERRUPTEUR NUMERO :";
31   Input N
32   N=N+1
33   M=M+1
34   A(N)=Abs(1-A(N))
35   Calcul:
36   R=Tan(Rnd+N/Rnd-N)-Sin(Rnd/N)+336*Sin(8*N)
37   N=Int(Abs(10*(R-Int(R))))
38   If N=0 Then
39     N=Random(10)+1
40   Endif
41   If A(N)=1 Then
42     A(N)=0
43     Goto Calcul
44   Endif
45   A(N)=1
46 Loop
47 Print
48 Print "TROUVE EN ";M;" ESSAIS"
49 R$=Input$(1)

```

L'indésirable

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : hasard et réflexion (1 joueur).

Difficulté du jeu : moyenne.

LE JEU :

Vous êtes le propriétaire d'un immeuble de 16 pièces vides, et vous n'arrivez pas à mettre la main sur votre locataire indésirable. Vous visitez les pièces au hasard, mais, si vous visitez la pièce voisine, votre locataire, prévenu par le bruit, va déménager. Il peut même se réinstaller dans une pièce que vous avez déjà visitée, et marquée comme vide. Il vous faudra arriver à le déloger en moins de 9 coups pour gagner.

LE PROGRAMME :

Une table B est dimensionnée à 16, en ligne 4, pour recevoir les numéros des cases déjà visitées.

Les lignes 7 et 8 permettront le calcul des positions d'affichage.

La boucle I (15-17) dessine le terrain de jeu.

La position initiale du locataire est tirée au hasard en ligne 18.

Le numéro de la case à visiter est donné en ligne 26.

Les lignes 29 à 44 contrôlent les diverses réponses possibles.

La boucle I (48-56) dessine les cases visitées.

La ligne 67 dessine le locataire, effacé en ligne 78.

Les lignes 79 à 82 assurent son déplacement.

POUR JOUER :

On lance l'exécution par RUN.

A la question CASE VISITEE, on répond en tapant une lettre, de A à P.

Si on a trouvé le locataire, celui-ci est affiché, et la partie est terminée ; sinon, deux cas peuvent se produire : soit l'on est suffisamment loin, et le locataire ne bouge pas, soit l'on est à une case, il apparaît un court instant, et il se recache, aléatoirement, dans une case voisine. Les cases déjà visitées se colorent en rouge, et vous ne pouvez pas y revenir. Si, au bout de 9 coups, vous n'avez pas trouvé, la position du locataire s'affiche, et vous avez perdu.

On tapera la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra, bien sûr, jouer sur plus de 16 cases, utiliser des caractères graphiques pour représenter le locataire et le propriétaire, utiliser un algorithme plus subtil pour le déplacement du locataire, lui interdire éventuellement de revenir dans des cases déjà visitées, etc.

```

1  Rem L'INDESIRABLE
2  Fullw 2
3  Clearw 2
4  Dim B(16)
5  Alert 1," ECRAN : 1:COUL
   EUR:2:NOIR ",0," 1 : 2
   ",R
6  C$=String$(3-2*(R=2),"X")
7  Deffn A(I)=4+4*(I-1-4*Int((
   I-1)/4))
8  Deffn B(I)=1+4*Int((I-1)/4)
9  Clearw 2
10 Deffill 2,2,20
11 Fill 1,1
12 Print At(1,16);Spc(38);At(1
   ,17);Spc(38)
13 B$=String$(1-1*(R=2)," ")
14 A$=String$(3-2*(R=2)," ")
15 For I=1 To 16
16   Print At(Fn A(I),Fn B(I))
   ;A$;At(Fn A(I),Fn B(I)+1);B
   $;Chr$(64+I);B$;
17 Next I
18 X=Random(16)+1
19 S=1
20 K=0
21 Debut:
22 Print At(1,16);Spc(38);
23 Sound 1,0
24 Deftext 7
25 Print At(1,16);
26 Input "CASE VISITEE :";V$
27 V$=Upper$(V$)
28 C=Asc(V$)-64
29 If B(C)=1 Then
30   Print At(1,17);"DEJA VISI
   TEE";
31   Sound 1,10,4,5,10
32   Goto Debut
33 Endif
34 If C=X Then
35   Print "VOUS L'AVEZ EU EN
   ";S;" COUPS. BRAVO"
36   K=1
37   Goto Testfin
38 Endif
39 If X-5=C Or X-4=C Or X-3=C
   Or X-1=C Or X+1=C Or X+3=C
   Or X+4=C Or X+5=C Then
40   Goto Testfin
41 Endif
42 Print At(1,17);"IL N'EST PA
   S LA";
43 Sound 1,10,4,2,10
44 Sound 1,0
45 Affiche:
46 S=S+1
47 B(C)=1
48 For I=1 To 16
49   Print At(Fn A(I),Fn B(I))
   ;
50   Deftext 3
51   If B(I)=1 Then
52     Print C$
53   Else
54     Print A$
55   Endif
56 Next I
57 If S=9 Then
58   Print At(1,17);"PERDU. IL
   ETAIT EN ";Chr$(64+X);
59   K=1
60   Goto Testfin
61 Endif
62 Pause 70
63 Print At(1,17);Spc(38)
64 Goto Debut
65 Testfin:
66 Color 2
67 Print At(Fn A(X),Fn B(X));B
   $;Chr$(234);B$
68 Sound 1,10,5,5,20
69 Pause 120
70 Sound 1,0
71 If K=1 Then
72   For P=1 To 5 Step 2
73     Sound 1,10,P,5,10
74   Next P
75   Sound 1,0
76   End
77 Endif
78 Print At(Fn A(X)+1,Fn B(X))
   ;" "
79 Repeat
80   Z=Random(10)-4
81   Until Abs(Z)<>2 And X+Z<=16
   And X+Z>=1
82   X=X+Z
83   Goto Affiche

```


Niveau 5

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Horloge 2	Une véritable horloge de précision.
Anagrammes	Un bon jeu de réflexe ; pour deux joueurs.
Cibles	Un peu d'adresse, beaucoup de chance : très visuel.
Gouttes	Un véritable gag visuel, à exploiter.
Combat aérien	Pas si facile de descendre un appareil qui bouge tout le temps.
Anomalie	Exercez votre perspicacité dans la reconnaissance des symboles.
Le jeu du 15	Un jeu de réflexion utilisant des chiffres.
Décomaque	Pour vous éviter de sécher trop longtemps sur Nicomaque.
Mur de briques	Un grand classique, dans une version très complète.
Plusieurs en un	Un terrain de jeu et les pions, pour deux joueurs.
Othello	Pour deux joueurs : il retourne les pions à votre place.
Tic tac toe	Un classique, mais pas sans intérêt, pour deux joueurs.
Dames	Pour deux joueurs : classique.
Parcours du cavalier	Très spectaculaire, et instructif à plus d'un titre.
Quatre à quatre	Un jeu de réflexion un peu difficile, pour deux joueurs.
Dessin de cercles	Une démonstration de la rapidité graphique du GFA.
Taquin de lettres	Un proverbe est caché dedans : à vous de le découvrir.
Œufs	Ne laissez pas faire la pondeuse diabolique ; pour un joueur.
Création de formes	Pour mettre au point vos propres icônes, en 16*16.
Ardoise magique	Laissez tomber le papier et le crayon.
Fais-moi un dessin	Composez vous-même votre paysage, avec des images déjà définies.
Echecs	Un programme assez simplifié, pour deux joueurs.

Utilisation de la souris, Fonctions particulières

La gestion de la souris se fait très facilement par l'utilisation de fonctions assez simples, mais bien conçues. On peut cacher la souris (**HIDEM**), la faire réapparaître (**SHOWM**), lui donner une forme prédéfinie (**DEFMOUSE N**) ou redéfinir soi-même sa forme (**DEFMOUSE A\$**). Trois fonctions particulières permettent de lire l'état de la souris : **MOUSEX** renvoie sa position horizontale, **MOUSEY** sa position verticale, et **MOUSEK** renvoie la valeur 0, 1, 2 ou 3 selon l'état des boutons. On peut d'ailleurs recueillir plus simplement toutes ces informations simultanément avec **MOUSE X,Y,Z**, qui renvoie dans les trois variables X, Y et Z les trois informations précédentes. La seule véritable difficulté est la transformation pratique des valeurs X et Y récupérées, pour tenir compte des emplacements sur l'écran.

Le niveau 5 fait essentiellement appel aux instructions de gestion de la souris, et présente également à l'occasion quelques fonctions supplémentaires non encore abordées. Par exemple **SETTIME A\$,B\$** qui permet de régler l'horloge interne en lui envoyant à la fois l'heure et la date. Ou **SETCOLOR I, N** qui permet de redéfinir soi-même toutes les couleurs (voir le programme Gouttes à titre d'exemple).

Horloge 2

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : simulation.

Difficulté du jeu : aucune.

LE JEU :

Le programme simule le fonctionnement d'une horloge digitale grand format, en utilisant les caractères d'origine du ST, notamment agrandis. Il présente la particularité de faire appel, pour une bonne précision sur l'heure et la date, à l'horloge interne du ST.

LE PROGRAMME :

Les lignes 6 à 19 permettent d'entrer la date (pour mois, année).

Les lignes 22 à 35 permettent d'entrer l'heure en tapant d'abord les heures, puis les minutes, puis les secondes. Le sous-programme 56-99 est utilisé pour définir les zones de saisie, contrôler les entrées et valider la saisie.

La ligne 36 permet d'entrer les informations horaires dans l'horloge interne.

La ligne 45 affiche la date. La ligne 46 affiche l'heure.

La ligne 47 s'occupe de l'arrêt : si on tape une touche quelconque, l'horloge s'arrête.

La ligne 48 permet de boucler.

POUR JOUER :

Faire RUN : la date est entrée sous la forme JJ.MM.AA. Puis la question « Entrez l'heure sous la forme HH : MM : SS » est affichée, et on tape l'heure, puis les minutes, et les secondes.

Le comptage du temps démarre dès la frappe de RET.

On arrête en tapant une touche quelconque.

EXTENSIONS POSSIBLES :

On peut améliorer les divers affichages, ajouter une possibilité de réveil, prévoir un agenda automatique, faire apparaître un coucou chantant au moment du passage de l'heure, etc.

```

1  Rem HORLOGE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Deftext 1,0,0,13+7*(P>0)
6  Print At(5,2);"ENTREZ LA DA
  TE ( FORME JJ.MM.AA )"
7  Deftext 1,0,0,65
8  A$=" _.: _.: _."
9  For I=1 To Len(A$)
10   X=50+20*(P=2)+30*I
11   Y=150+40*(P>0)
12   Text X,Y,Mid$(A$,I,1)
13 Next I
14 Gosub Heure(0,1,80)
15 S$=R$+ "."
16 Gosub Heure(0,2,170)
17 S$=S$+R$+ "."
18 Gosub Heure(0,3,260)
19 S$=S$+R$.
20 Clearw 2
21 Deftext 1,0,0,13+7*(P>0)
22 Print At(5,2);"ENTREZ L'HEU
  RE ( FORME HH:MM:SS )"
23 Deftext 1,0,0,65
24 A$=" _.: _.: _."
25 For I=1 To Len(A$)
26   X=50+20*(P=2)+30*I
27   Y=150+40*(P>0)
28   Text X,Y,Mid$(A$,I,1)
29 Next I
30 Gosub Heure(1,1,80)
31 T$=R$+ ":"
32 Gosub Heure(1,2,170)
33 T$=T$+R$+ ":"
34 Gosub Heure(1,3,260)
35 T$=T$+R$
36 Settime T$,S$
37 Clearw 2
38 Deffill 1+P,2,19
39 Rbox 120+60*(P=2),10,320+80
  *(P=2),90
40 Fill 130,20
41 Deffill 1+2*P,2,12
42 Rbox 130+65*(P=2),100,310+7
  5*(P=2),170
43 Fill 150,120
44 Do
45   Text 140+70*(P=2),60,Date
  $
46   Text 156+70*(P=2),144,Time$
47   Exit If Inkey$<>""
48 Loop
49 For I=1 To 6
50   Sound 1,10,2*I,5,10
51 Next I
52 Sound 1,0
53 Deftext 1,0,0,13+7*(P>0)
54 Print At(1,17);"L'HORLOGE E
  ST ARRETEE"
55 R$=Input$(1)
56 Procedure Heure(H,K,Z)
57   R$=""
58   S=0
59   Repeat
60     Repeat
61       Test=0
62       X=Z+30*S+20*(P=2)
63       Text X,Y,Chr$(63)
64       A$=Chr$(Inp(2))
65       Exit If (K=1 And H=1
  And S=0 And A$<>"0" And A$<
  >"1" And A$<>"2")
66       Exit If (K=1 And H=0
  And S=0 And A$>"3")
67       Exit If (K=2 And S=0
  And A$>"1" And H=0)
68       Exit If ((K=2 Or K=3)
  And H=1 And S=0 And A$<>"0"
  " And Val(A$)>5)
69       If (S=1 And A$<>"0" A
  nd Val(A$)=0) Then
70         S=S-1
71         Goto Fin
72       Endif
73       If A$=" " And R$<>""
  Then
74         Text X,Y," "
75         R$=""
76         S=0
77         Goto Fin
78       Endif
79       R$=R$+A$
80       Text X,Y,A$
81       S=S+1
82       Exit If S<2
83       If (K=2 And Val(R$)>1
  2 And H=0) Then
84         Z=170
85         R$=""
86         S=0
87         Goto Fin
88       Endif
89       If (K=1 And Val(R$)>2
  4 And H=1) Or (K=1 And Val(
  R$)>31 And H=0) Then
90         Z=80
91         R$=""
92         S=0
93         Goto Fin
94       Endif
95       Test=1
96       Fin:
97       Until Test=1
98       Until Test<>0
99   Return

```

Anagrammes

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de réflexion (2 joueurs).

Difficulté du jeu : selon le niveau.

LE JEU :

Des lettres sont choisies au hasard par le ST, le nombre de lettres étant fonction du niveau de difficulté désiré. Puis les lettres du mot ainsi obtenu sont brouillées, et affichées au bas de l'écran ; un nouveau mot est alors confectionné, de façon aléatoire, à partir des mêmes lettres et il est affiché sous le premier, et ainsi de suite ; les deux joueurs (gauche et droit) disposent chacun d'une touche (ESC ou *, ou l'un des boutons de la souris) et doivent la presser dès qu'ils voient s'afficher au bas de l'écran un mot déjà présent sur l'écran ; leur score est alors évalué, et un message correspondant à leur résultat sera délivré.

LE PROGRAMME :

Le niveau du jeu est entré à la ligne 10. La boucle I (14-16) réalise le choix des lettres.

Une deuxième boucle I (20-29) effectue le brouillage, et les lignes 24 et 25 permettent de détecter la pression d'une touche.

La boucle I (34-37) affiche les mots, et fait monter d'une ligne sur l'écran les mots déjà obtenus.

La ligne 47 cherche quel est celui des deux joueurs qui a pressé une touche.

La boucle I (52-54) explore les mots présents sur l'écran ligne par ligne.

On obtient ainsi un nouveau mot, qui est comparé au mot initial en ligne 53 et le résultat BRAVO apparaît si c'est exact.

Sinon, la partie s'arrête sur l'affichage PERDU (ligne 57).

POUR JOUER :

On lance le programme par RUN.

On devra ensuite entrer le niveau de difficulté (en principe de 1 à 3) et faire RET. Les mots s'affichent en partant du bas de l'écran, et montent au fur et à mesure ; si l'un des joueurs croit reconnaître dans le mot qui apparaît un des mots déjà affichés, il devra immédiatement presser sa touche (touches souris, ou ESC et *), et son résultat sera affiché après un court instant.

La partie sera alors terminée.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On pourra utiliser d'autres affichages que des mots, modifier les temps d'affichage, faire participer plus de deux joueurs, améliorer le son et la couleur, etc.

```

1  Rem ANAGRAMMES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Deftext 1,0,0,13+7*(P>0)
5  Dim T$(21),S$(2)
6  S$(0)="PERDU"
7  S$(1)="BRAVO"
8  Repeat
9    Clearw 2
10   Input "NIVEAU ( DE 1 A 3 ) : ";N
11   Until N=Int(N) And N>=1 And N<=3
12   Clearw 2
13   C$=""
14   For I=1 To N+3
15     C$=C$+Chr$(Random(26)+65)
16   Next I
17   Repeat
18     B$=""
19     T$=C$
20     For I=1 To Len(C$)
21       Repeat
22         N=Random(Len(C$))+1
23         Until Mid$(T$,N,1)<>"-"
24         Bo=Mousek
25         R=Asc(Right$(Inkey$,1))
26         Exit If R=27 Or R=42 Or Bo=1 Or Bo=2
27         B$=B$+Mid$(T$,N,1)
28         Mid$(T$,N,1)="-"
29       Next I
30       Sound 1,8,Int(Rnd*12+1),5,7
31       Sound 1,0
32       Co$=Chr$(33+Random(4^P))
33       T$(19)=Co$+B$
34       For I=19 To 1 Step -1
35         Deftext Asc(Left$(T$(I),1))-31,0,0,13+7*(P>0)
36         Print At(1,I);Mid$(T$(I),2)
37       Next I
38       For I=0 To 19
39         T$(I)=T$(I+1)
40       Next I
41       For J=1 To 7
42         Bo=Mousek
43         R=Asc(Right$(Inkey$,1))
44         Exit If R=27 Or R=42 Or Bo=1 Or Bo=2
45       Next J
46     Until R=27 Or R=42 Or Bo=1 Or Bo=2
47     If Bo=2 Or R=1 Then
48       H$="JOUEUR DE GAUCHE"
49     Else
50       H$="JOUEUR DE DROITE"
51     Endif
52     For I=0 To 17
53       Exit If B$=Mid$(T$(I),2)
54     Next I
55     Deftext 1,0,0,13+7*(P>0)
56     Fin:
57     Print At(8,I+1);S$(-(I<=17));" ";H$
58     For I=1 To 12
59       Sound 1,10,I,5,10
60     Next I
61     Sound 1,0
62     R$=Input$(1)

```

Cibles

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : hasard.

Difficulté du jeu : facile.

LE JEU :

Il consiste simplement à tirer sur des cibles, en essayant de marquer le maximum de points ; le nombre de cibles (de 1 à 4, ou 8, selon l'écran) sera choisi en début de partie, et le score réalisé sera affiché en permanence. Si l'on touche le bord d'une cible, la partie s'arrête. On marquera d'autant plus de points que le nombre sur lequel on tombe sera plus grand.

LE PROGRAMME :

Le nombre de cibles est entré en ligne 9. La ligne 12 permet de choisir entre 2 types de jeux. La boucle I (24-29) effectue l'impression des cibles.

La ligne 34 affichera le score en permanence.

La balle est affichée par la ligne 37.

La boucle I (42-48) réalise l'impression des chiffres contenus dans les cibles, et permet le déplacement de la boule (ligne 47).

La ligne 52 regarde si l'on a tiré, et si la boule est encore sur la ligne de départ.

La ligne 53 fait progresser la boule d'une case vers la droite, et la ligne 54 vérifie si l'on a ou non atteint les cibles ; puis la ligne 58 regarde ce que l'on a touché.

La ligne 59 incrémente le score.

Les lignes 64 et 65 s'occupent de la fin de partie.

POUR JOUER :

Lancer le programme par RUN, et taper le nombre de cibles désiré suivi de RET. On choisira ensuite le type de jeu en tapant 0 ou 1. Les cibles s'affichent, et la partie commence. Votre balle, représentée par un O, est affichée en bas à gauche ; vous pouvez la déplacer avec les touches de la souris et vous pouvez tirer à tout moment en appuyant simultanément sur les 2 touches.

Le score sera modifié en fonction de votre résultat, et une nouvelle balle apparaît. Il est possible d'arrêter à tout moment en tirant à côté ; sinon, le jeu s'arrête au bout d'un certain nombre de tirs.

On tape la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra sans difficulté augmenter le nombre de cibles, modifier la règle d'apparition des nombres dans les cibles pour la rendre moins hasardeuse, s'arranger pour que la balle elle-même soit représentée par un chiffre qui devrait coïncider à l'arrivée avec celui de la cible.

```

1  Rem CIBLES
2  P=Abs(Xbios(4)-2)
3  U=2-2*(P>0)
4  Fullw 2
5  Deftext 1,0,0,13+7*(P>0)
6  Repeat
7    Clearw 2
8    Print "NOMBRE DE CIBLES :
9      ";
9    N=Val(Chr$(Inp(2)))
10   Until N<=8+4*(P>0)
11   Print N
12   Print At(1,5);"0:FACILE 1:D
13     IFFICILE ?"
13   Op=Val(Chr$(Inp(2)))
14   If Op>1 Then
15     Run
16   Endif
17   Clearw 2
18   Dim A(N),U(N)
19   Z=0
20   B=1
21   Nb=10
22   Color 3
23   Defline 1,14,2,0
24   For I=1 To N
25     A(I)=9
26     Line 290,32*I-22,320,32*I
27       -22
27     Line 316,32*I-22,316,32*I
28       +10
28     Line 290,32*I+10,320,32*I
29       +10
29   Next I
30   Repeat
31     K=1
32     A=17
33     Debut:
34     Print At(1,19);"SCORE : "
35       ;Z;" - BALLE : ";B
35     Bo=0
36     Repeat
37       Print At(K,A);"0"
38       If K>2 And Op=1 Then
39         Hidem
40       Endif
41       D=A
42       For I=1 To N
43         U(I)=Random(A(I))
44         Deftext A(I)
45         Print At(38,U*I);Chr$
46           (48+U(I))
46         Bo=Mousek
47         A=A-(Bo=1 And A<18)+(
48           Bo=2 And A>1)
48       Next I
49       Pause 3
50       Deftext 1
51       Print At(K,D);" "
52       Until (Bo=3 Or K<>1)
53       K=K+1
54       If K<38 Then
55         Goto Debut
56       Endif
57       Exit If D Mod U<>0 Or D<1
58       Or D>U*N
58       F=D/U
59       Z=Z+U(F)
60       Sound 1,10,4,5,4
61       Sound 1,0
62       A(F)=A(F)-2
63       Nb=Nb-1
64     Until Nb=1
65     Print "C'EST FINI....."
66     For I=1 To 12
67       Sound 1,10,I,5,8
68     Next I
69     Sound 1,0
70     R$=Input$(1)

```

Gouttes

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : dessin.

Difficulté du jeu : aucune.

LE JEU :

C'est un véritable programme de dessin animé que l'on peut considérer, en fait, comme un interlude ; un récipient est dessiné sur l'écran, vu en coupe, et il contient déjà des obstacles. Une goutte d'eau, symbolisée par un rond gris, apparaît au-dessus du récipient et tombe en remplissant le récipient au fur et à mesure. Sa trajectoire, lorsqu'elle rencontre un obstacle, est entièrement déterminée par le hasard : elle part à droite ou à gauche, jusqu'à ce qu'il n'y ait plus aucune possibilité. Alors, une autre goutte tombe, et ainsi de suite, jusqu'au débordement final.

LE PROGRAMME :

Les lignes 11 à 35 permettent de choisir les couleurs de la balle et des obstacles en utilisant les touches de la souris.

Les boucles emboîtées I et J (40-55) dessinent le récipient et les obstacles.

La ligne 57 tire la position de départ de la goutte.

La ligne 64 affiche la goutte, effacée par la ligne 68.

La ligne 69 contrôle la fin de partie.

Les lignes 70 à 89 traitent les différents cas.

POUR JOUER :

En mode noir et blanc, le RUN déclenche le dessin du récipient, et la goutte commence à tomber ; sinon, on commencera par choisir les couleurs, en cliquant sur une fenêtre d'alerte pour choisir les intensités des 3 couleurs de base et sur fin pour passer au choix suivant.

Le programme s'arrête de lui-même lorsque le récipient déborde ; on peut l'arrêter à tout moment en tapant la combinaison CSA..

EXTENSIONS POSSIBLES :

On peut, bien entendu, améliorer la couleur et le son et aussi envisager un jeu dans lequel un petit personnage devrait échapper aux gouttes ou à la noyade.

```

1  Rem GOUTTES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Li=16+8*(P>0)
6  Deftext 1,0,0,13+7*(P>0)
7  Dim A(3),T(17,20),A$(2),B$(
3),Var(3)
8  If P=0 Then
9    Goto Debut
10 Endif
11 A$(1)="BALLE"
12 A$(2)="OBSTACLES"
13 B$(1)="ROUGE"
14 B$(2)="VERT"
15 B$(3)="BLEU"
16 For I=1 To 2
17   J=1
18   M$=A$(I)
19   Repeat
20     N$=M$+"
;
;INTENSITE DU "+B$(J)
21     Alert 2,N$,2," + - : SU
ITE : FIN ",Var
22     J=J-(Var=2)
23     R=(R+(J=1)*(Var=1)) Mod
8
24     V=(V+(J=2)*(Var=1)) Mod
8
25     B=(B+(J=3)*(Var=1)) Mod
8
26     J=J+(J-1)*(J>3)
27     N=R*256+V*16+B
28     Setcolor I,N
29     Print At(16+16*(2-P),1)
; "ROUGE VERT BLEU"
30     Print At(17+16*(2-P),2)
; R; At(24+16*(2-P),2); V; At(3
1+16*(2-P),2); B
31     Deftext I+1
32     Print At(22+16*(2-P),4)
; "*****"
33     Deftext 1
34     Until Var=3
35 Next I
36 Debut:
37 Deffill 0,0,0
38 Pbox 0,0,640,400
39 Deftext 1,1
40 For I=9 To 17
41   For J=2 To 15
42     Deffill 1,2,16
43     Pbox 8,Li*I-Li,16,Li*I
44     Pbox 128,Li*I-Li,136,Li
*I
45     Pbox 8*J,Li*16,8*J+8,Li
*16+Li
46     If I>10 And I<>17 And R
nd<0.35 Then
47       Deffill 3,2,7
48       Pbox 8*J,Li*I-Li,8*J+
8,Li*I
49       T(J,I)=1
50       Endif
51       T(1,I)=1
52       T(16,I)=1
53       T(J,17)=1
54     Next J
55   Next I
56   Do
57     X=Random(7)+6
58     Y=5
59     Cas1:
60     K=-2*(Rnd<0.5)-1
61     I=0
62     Cas2:
63     Deffill 2,2,8
64     Pbox 8*X,Li*Y-Li,8*X+8,Li
*Y-1
65     Sound 1,9,4,5,1
66     Sound 1,0
67     Deffill 0,0,0
68     Pbox 8*X,Li*Y-Li,8*X+8,Li
*Y-1
69     Exit If Y>16
70     J=T(X,Y+1)
71     If J=0 Then
72       Y=Y+1
73     Goto Cas1
74   Endif
75   Cas3:
76   J=T(X+K,Y)
77   If J=0 Then
78     X=X+K
79     I=1
80     Goto Cas2
81   Endif
82   If I=0 Then
83     K=-K
84     I=1
85     Goto Cas3
86   Endif
87   Deffill 2,2,8
88   Pbox 8*X,Li*Y-Li,8*X+8,Li
*Y-1
89   T(X,Y)=1
90 Loop
91 Print At(1,19); "C'EST FINI"
92 For I=1 To 12
93   Sound 1,10,I,5,8
94 Next I
95 Sound 1,0
96 R$=Input$(1)

```


Combat aérien

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexe (1 joueur).

Difficulté du jeu : difficile.

LE JEU :

Au centre de l'écran, se trouve le viseur de votre appareil, et l'avion ennemi se déplace devant vous, d'une façon qui pourra vous sembler, au début, incohérente.

Vous pouvez vous déplacer dans toutes les directions, en bougeant la souris, et vous allez essayer d'amener l'appareil ennemi au centre de votre viseur.

Vous tirez sur lui en frappant la barre d'espace et vous voyez vos projectiles qui convergent sur lui. Mais il continue à se déplacer, même pendant votre tir, et il n'est pas du tout évident que vous réussirez à l'avoir.

La partie continuera à se dérouler jusqu'à ce que vous l'ayez touché (affichage de BRAVO).

LE PROGRAMME :

La ligne 24 dessine l'appareil ennemi, dont le déplacement aléatoire sera réalisé par les lignes 25 et 30.

Les lignes 22 et 23 vous permettent le pilotage de votre appareil.

La ligne 21 dessine le viseur de votre appareil.

Le sous-programme, gérant le tir, va de la ligne 38 à la ligne 46, et le test de fin de partie est effectué en ligne 47.

La ligne 32 regarde si vous avez tiré ou non.

POUR JOUER :

Dès que le programme a été lancé par l'ordre RUN, vous devez manœuvrer pour essayer de faire en sorte que l'appareil ennemi soit cadré dans votre viseur.

Pour cela, vous disposez de la souris, qu'il faudra déplacer très doucement, et vous pouvez tirer à tout moment en pressant la barre d'espace.

L'appareil ennemi peut continuer à se déplacer de manière aléatoire, même lorsque vous avez tiré.

La partie se terminera avec l'affichage de BRAVO, si vous touchez l'avion ennemi.

On peut aussi arrêter à tout moment en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera assez facile de faire un affichage au graphisme plus soigné, utilisant la couleur. On pourrait envisager de stabiliser la position de l'avion ennemi pendant le tir.

On peut aussi introduire plusieurs avions, compter les points ou le temps, tenir compte de sa propre réserve de carburant, etc.

```

1  Rem COMBAT AERIEN
2  P=Abs(Xbios(4)-2)
3  C=300+150*(P>0)-140*(P=1)
4  D=200+100*(P>0)
5  M=21+4*(P>0)
6  F=70+35*(P>0)
7  Fullw 2
8  Clearw 2
9  G=28+19*(P=2)
10 A$=Mki$(8)+Mki$(8)+Mki$(0)+Mki$(1)+Mki$(1+Random(4
    ^P))
11 For I=0 To 15
12   Read X
13   A$=A$+Mki$(X)+Mki$(X)
14 Next I
15 Hidem
16 A=C
17 B=D
18 Do
19   N=M
20   Debut:
21   Circle C,D-F,16
22   U=Mousex
23   V=Mousey
24   Sprite A$,A,B
25   A=Abs(A+2*(-(U>A)+(U<A)))
26   A=Abs(A+(2*Random(2)-1))
27   A=A Mod (2*C)
28   B=Abs(B+2*(-(V>B)+(V<B)))
29   B=Abs(B+(2*Random(2)-1))
30   B=B Mod (2*D)
31   R$=Inkey$
32   If R$="" Then
33     If N>=M Then
34       Goto Debut
35     Endif
36   Endif
37   Deftext 2
38   Repeat
39     Print At(G+20-N,N);Chr$(249)
40     Print At(G+N,N);Chr$(249)
41     Sound 1,9,8,6,2
42     Sound 1,0
43     Print At(G+20-N,N);" "
44     Print At(G+N,N);" "
45     N=N-1
46   Until N<=9
47   Exit If Abs(Mousex-C)<=4 And Abs(Mousey-D+F/2)<=
4
48 Loop
49 Deftext 1
50 Print "BRAVO"
51 For I=1 To 12
52   Sound 1,10,1,5,8
53 Next I
54 Sound 1,0
55 R$=Input$(1)
56 Data 2016,3120,6168,6168,16380,65535,65535,65535,6
57 Data 32766,16380,8184,8184,6120,10212,17346

```

Anomalie

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : solitaire.

Difficulté du jeu : moyenne.

LE JEU :

Il repose essentiellement sur une bonne capacité d'observation. Deux grilles de 25 cases chacune sont dessinées sur l'écran. Des symboles identiques sont affichés en des endroits différents des deux grilles, mais l'un des symboles a été modifié sur la grille de droite. Le jeu consiste à trouver lequel en cliquant sur la bonne case dans la grille de droite.

LE PROGRAMME :

La boucle I (8-16) imprime les deux grilles.

La boucle principale I (19-28) imprime les symboles des deux grilles, dans un ordre différent.

La ligne 20 choisit le caractère à imprimer.

Le sous-programme (55-62) s'occupe de faire l'impression à un emplacement libre (on l'appelle deux fois pour les deux grilles).

Le sous-programme (63-73) s'occupe de l'affichage proprement dit.

La ligne 30 change, de façon aléatoire, l'un des caractères.

Les coordonnées du point essayé sont calculées aux lignes 36 à 41, et le contrôle est effectué par la ligne 43.

Les lignes 46 et 48 s'occupent de l'affichage du résultat.

POUR JOUER :

Après avoir lancé le programme par l'instruction RUN, on obtient l'affichage des deux grilles. On devra alors chercher, par comparaison, quel est le caractère qui a été modifié (sur la grille de droite).

On clique sur la case contenant le caractère modifié.

Le programme répondra alors BRAVO ou indiquera la lettre qu'il fallait découvrir.

On termine en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

Il sera possible d'utiliser des symboles graphiques de diverses couleurs, ce qui pourra simplifier la recherche.

On peut aussi envisager d'utiliser des dessins plus sophistiqués, améliorer le contrôle du résultat, admettre la bonne réponse sur la grille de droite ou celle de gauche, introduire un minutage, etc.

```

1  Rem ANOMALIE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Dim T(6,12)
5  U=32+16*(P>0)
6  H=-7*(P>0)
7  L=24+12*(P>0)
8  For I=0 To 5
9    Print At(2*I+3,1);Chr$(48
+I);At(1,2*I+2);Chr$(48+I)
10   Print At(2*I+19,1);Chr$(4
8+I);At(17,2*I+2);Chr$(48+I
)
11   For J=0 To 5
12     B=U*I-16*(P=0)
13     Box 16*I+12,B+H,16*I+26
,B+L+H
14     Box 16*I+140,B+H,16*I+1
54,B+L+H
15   Next J
16 Next I
17 Print At(1,14);"VEUILLEZ AT
TENDRE UN INSTANT"
18 Pause 10
19 For I=1 To 30
20   D=Random(26)+1
21   L=0
22   S=6
23   Gosub Tir
24   L=6
25   Gosub Tir
26   Sound 1,10,Random(12)+1,R
andom(4)+2,2
27   Sound 1,0
28 Next I
29 S=0
30 Gosub Tir
31 T=Timer
32 Gosub Vis
33 Print At(1,14);"CLIQUEZ SUR
LA GRILLE DE GAUCHE"
34 Print "SUR LA LETTRE A DECO
UVRIR"
35 Print At(1,16);"COLONNE
LIGNE      "
36 Repeat
37   Bo=Mousek
38   Print At(10,16);Y;"  ";At
(23,16);X;"  "
39   Y=Int((Int((Mousex-4)/8)-
1)/2)-8
40   X=Int((Int((Mousey+2)/16)
-1)/(1-(P=0)))-(P>0)
41 Until Bo=1 And X>=0 And X<=
5 And Y>=0 And Y<=5
42 E=T(Y,X+6)
43 If E=D Then
44   Print At(1,18);"BRAVO
"
45 Else
46   Print At(1,18);"LA LETTRE
A TROUVER ETAIT ";Chr$(D+6
4)
47 Endif
48 Print "VOTRE TEMPS :";Int((
Timer-T)/200)
49 For I=1 To 12
50   Sound 1,10,I,5,8
51 Next I
52 Sound 1,0
53 R#=Input$(1)
54 Z240:
55 Procedure Tir
56   Repeat
57     F=Random(6)
58     G=Random(5)+L
59     E=T(F,G)
60     Until E=0 Or S=0
61     T(F,G)=D
62   Return
63 Procedure Vis
64   For I=0 To 5
65     For J=0 To 5
66       Deftext Random(4^P)+1
67       C=T(I,J)+64+32*(T(I,J
)=0)
68       Print At(2*I+3,2*J+2)
;Chr$(C)
69       C=T(I,J+6)+64+32*(T(I
,J+6)=0)
70       Print At(2*I+19,2*J+2
);Chr$(C)
71     Next J
72   Next I
73   Return

```

Le jeu du 15

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : moyenne.

LE JEU :

C'est un jeu pour deux joueurs, assez ancien, qui était autrefois pratiqué dans les foires. On dispose d'un tapis de jeu de 9 cases, et chacun mise à son tour un jeton de sa couleur, sur une case disponible.

Le premier des deux joueurs qui, avec exactement 3 jetons, arrive à faire un total de 15, a gagné.

LE PROGRAMME :

La boucle I (13-18) dessine le terrain de jeu, et la ligne 23 s'occupe de la gestion du tour des joueurs (jeton O ou X correspondant).

La boucle (27-31) attend le coup joué.

La ligne 34 vérifie la validité du coup.

La ligne 39 teste la fin de partie, en cas de match nul, et la ligne 40 refuse les coups non conformes (avec retour à la situation initiale, ligne 19).

La ligne 49 s'occupe de l'affichage du coup, et la ligne 58 l'enregistre dans une variable R\$. La partie la plus intéressante du programme, qui pourrait servir à définir une stratégie machine, va de la ligne 62 à la ligne 79 : on utilise 3 boucles emboîtées pour vérifier si l'un des joueurs a obtenu, avec 3 pions, un total de 15. Le changement de joueur s'effectue en ligne 81.

POUR JOUER :

Lancer le jeu en tapant RUN.

Le terrain de jeu s'affiche, puis la question JOUEUR X, et le joueur qui a choisi les X devra cliquer sur la case où il désire jouer. Si elle est libre, son coup y sera enregistré.

On obtient, ensuite, l'affichage JOUEUR O et le deuxième joueur procède de la même façon. Si l'un des joueurs gagne (total de 15 sur 3 coups), il obtient l'affichage BRAVO et la partie s'arrête.

En cas de match nul, la machine affiche C'EST FINI.

On fera une autre partie en tapant la combinaison CSA pour arrêter, puis RUN.

EXTENSIONS POSSIBLES :

Le plus intéressant sera sans doute d'essayer de définir une stratégie machine (en essayant d'éviter que la machine ne joue toujours le 5 lorsqu'elle doit commencer).

On peut aussi, bien sûr, améliorer le graphisme et introduire des effets sonores pour les différents messages.

```

1  Rem LE JEU DU QUINZE
2  P=Abs(Xbios(4)-2)
3  Deftext 1,0,0,13
4  Fullw 2
5  Dim S$(2)
6  H=Random(2)
7  R$="123456789"
8  O$=""
9  X$=""
10 U=32
11 S$(0)="X"
12 S$(1)="O"
13 For I=1 To 9
14   Box U*I-10,30,U*I+18,66
15   Deffill 1,2,8
16   Pbox U*I-8,70,U*I+16,108
17   Print At(4*I+1,6);Chr$(48
    +I)
18 Next I
19 Do
20   Deftext 1,0,0,13+7*(P>0)
21   Print At(1,15);"TOUR DU J
    QUEUR : ";
22   Deftext H+2
23   Print S$(H)
24   Sound 1,10,5*H+2,5,8
25   Print At(1,16);"CLIQUEZ S
    UR LE NUMERO CHOISI"
26   Sound 1,0
27   Repeat
28     Bo=Mousek
29     X=Int(Mousex/16)
30     Y=Mousey
31   Until Not (Bo<>1 Or Y<71
    Or Y>108 Or (X Mod 2)<>0 Or
    X>18 Or X<2)
32   A$=Right$(Str$(Int(X/2)),
    1)
33   I=Instr(0,R$,A$)
34   If I<>0 Then
35     Sound 1,10,5*I+2,4,6
36     Sound 1,0
37     Gosub Test
38   Endif
39   Exit If Val(R$)=0
40 Loop
41 Print At(1,17);"C'EST FINI"
42 For I=1 To 12
43   Sound 1,10,I,5,9
44 Next I
45 Sound 1,0
46 R$=Input$(1)
47 Procedure Test
48   Deftext -(H+1)*(P>0)+1,0,
    0,20
49   Text 32*I,58,Chr$(88-9*H)
50   Deftext 1,0,0,13
51   If H=0 Then
52     X$=X$+A$
53     T$=X$
54   Else
55     O$=O$+A$
56     T$=O$
57   Endif
58   Mid$(R$,I,1)="0"
59   If Len(T$)<3 Then
60     Goto Change
61   Endif
62   For I=1 To 3
63     Exit If I=Len(T$)
64     For J=I+1 To Len(T$)
65       Exit If J=Len(T$)
66       For K=J+1 To Len(T$)
67         If Val(Mid$(T$,I,1)
    )+Val(Mid$(T$,J,1))+Val(Mid
    $(T$,K,1))=15
68         Deftext 1,0,0,13+
    7*(P>0)
69         Print At(1,17);"B
    RAVO, JOUEUR ";S$(H)
70         For I=1 To 12 Ste
    p 2
71           Sound 1,10,I,5,
    10
72         Next I
73         Sound 1,0
74         R$=Input$(1)
75         End
76       Endif
77     Next K
78   Next J
79 Next I
80 Change:
81 H=Abs(H-1)
82 Sound 1,10,6*H+2,4,8
83 Sound 1,0
84 Return

```

Décomaque

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : solution du programme Nicomaque.

Difficulté du jeu : aucune.

LE JEU :

C'est une réplique au programme Nicomaque, proposé dans les pages précédentes ; vous entrez les restes successivement dans les divisions par 3, 5 et 7 d'un entier compris entre 1 et 100, et le ST calcule et affiche cet entier. Le procédé retenu peut d'ailleurs être généralisé (à vous de trouver comment).

LE PROGRAMME :

La boucle I (9-12) affiche quelques explications ; en appelant le sous-programme (58-67).

Les lignes 28 à 36 attendent que vous cliquiez sur un reste.

La ligne 37 emmagasine les restes dans une variable dimensionnée à 3 A (S).

La ligne 41 réaffiche les restes entrés.

Les lignes 46 à 56 gèrent la fin de partie.

POUR JOUER :

On lance le programme par RUN. Un texte de présentation est affiché, puis le message "RESTE DANS LA DIVISION PAR 3" s'affiche.

On doit cliquer sur 0, 1 ou 2 pour 3, puis de 0 à 4 pour 5 et de 0 à 6 pour 7.

La solution est affichée dès que l'on a donné les 3 restes.

Pour une autre partie, taper O à la question VOULEZ-VOUS UN AUTRE ESSAI (O/N).

Tapez N pour finir.

EXTENSIONS POSSIBLES :

On peut étendre le programme aux entiers de 1 à 1000, en utilisant les restes par 3, 5, 7 et 11 ; on pourra améliorer le son et la couleur pour rendre le programme plus attrayant.

```

1  Rem DECOMAQUE
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Dim A(3)
6  Deftext 1,0,0,13+7*(P>0)
7  Ae=62+25*(P=2)
8  K=0
9  For I=0 To 2
10   Read A$
11   Gosub Lec(A$)
12 Next I
13 A=Inp(2)
14 Repeat
15   K=64+34*(P=2)
16   F=32+17*(P=2)
17   G=18+9*(P=2)
18   Clearw 2
19   Restore Debut
20   For J=1 To 3
21     Read A$
22     Deftext J+1,0,0,G
23     For I=1 To Len(A$)
24       Sound 1,10,Int(Rnd*12
25 )+1,Int(Rnd*4+2),2
26       Text G*I-10,40,Mid$(A
27 $,I,1)
28       Sound 1,0
29     Next I
30     Repeat
31       Bo=Mousek
32       For I=0 To 2*J
33         Deftext 1,-18*(2*I+
34 1=A And Yo>73 And Yo<103),0
35 ,K
36       Text K*I+30+20*(P=2
37 ),100,Chr$(48+I)
38     Next I
39     Yo=Mousey
40     A=Int((Mousex+2)/F)
41     Until Bo<>0 And Odd(A)
42     And Yo>73 And Yo<103
43     A(J)=(A-1)/2
44     A$="PAR"+Str$(2*J+1)+":
45 "+Str$(A(J))
46   Deftext J+1,0,0,G
47   For I=1 To Len(A$)
48     Text G*I+450+236*(P=2
49 ),20*J+50,Mid$(A$,I,1)
50     Sound 1,8,I,5,4
51     Sound 1,0
52   Next I
53   Next J
54   G=A(1)*70+A(2)*21+A(3)*15
55   While G>105
56     G=G-105
57   Wend
58   Deftext 1,0,0,10+4*(P=2)
59   Text 130+100*(P=2),140,"L
60 A SOLUTION EST "+Str$(G)
61 A$="AUTRE ESSAI (O/N )"
62   Deftext 1,0,0,13+7*(P>0)
63   K=192+32*(P>0)
64   Gosub Lec(A$)
65   A$=Upper$(Input$(1))
66   Until A$="N"
67   Procedure Lec(A$)
68     Local I
69     For I=1 To Len(A$)
70       K=K+1
71       Deftext Random(4^P)+1
72       Print At(2+(K Mod Ae),4
73 +4*(Int(K/Ae)));Mid$(A$,I,1
74 );
75       Sound 1,10,Int(Rnd*12+1
76 ),Int(Rnd*4+2),2
77       Sound 1,0
78     Next I
79   Return
80 Data VOUS DEVEZ ENTRER AVEC
81 LA SOURIS LES TROIS RESTES
82 DANS LES DIVISIONS
83 Data " PAR TROIS, PAR CINQ
84 ET PAR SEPT D'UN NOMBRE ENT
85 IER INFÉRIEUR A 100."
86 Data " APPUYEZ SUR UNE TOUC
87 HE POUR COMMENCER LA PARTIE
88 "
89 Debut:
90 Data RESTE DANS LA DIVISION
91 PAR 3,RESTE DANS LA DIVISI
92 ON PAR 5
93 Data RESTE DANS LA DIVISION
94 PAR 7

```


Mur de briques

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexe (pour 1 joueur).

Difficulté du jeu : difficile.

LE JEU :

S'il est un jeu que l'on peut présenter comme un grand classique des jeux d'arcades pour micro-ordinateur, c'est bien le mur de briques (en anglais BREAKOUT).

Il est offert, ici, dans une version simplifiée, mais offre néanmoins de bonnes performances : affichage du score en fin de partie, rebonds aléatoires, etc.

La règle en est très élémentaire : vous disposez d'une raquette qui peut être déplacée horizontalement pour renvoyer une balle (matérialisée par O) qui va démolir les briques dessinées en haut de l'écran.

Votre problème est d'en démolir le plus grand nombre possible, sans rater la balle, ce qui provoquerait la fin de partie et l'affichage du score.

LE PROGRAMME :

Les lignes 8 à 11 dessinent le terrain de jeu fictif et les briques sont mises en place par les boucles emboîtées I et J (18-31).

La ligne 42 dessine la raquette dont la position est contrôlée par les lignes 43 à 49.

Les lignes 58 à 78 regardent si l'on a touché le mur, les briques ou la raquette.

Les rebonds sont gérés en conséquence avec d'éventuels aléas, par les lignes 89 à 102.

La ligne 85 affiche le score en fin de partie.

POUR JOUER :

Le programme sera lancé par l'instruction RUN, ce qui provoque l'affichage du terrain de jeu puis, très rapidement, l'envoi de la balle. Le joueur devra alors essayer de déplacer sa raquette à l'aide des touches de la souris. Il sera préférable de la tenir dans sa main.

Si, à un moment quelconque, sa balle ne touche pas la raquette, le score est affiché, et la partie s'arrête.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut envisager d'agrandir le terrain de jeu, d'augmenter le nombre de briques, de prévoir plusieurs balles, d'introduire des effets sonores.

On peut aussi, en s'inspirant des jeux d'arcades, introduire de petits ouvriers qui viendront reconstruire votre mur de briques, au fur et à mesure que vous le démolissez.

```

1  Rem MUR DE BRIQUES
2  P=Abs(Xbios(4)-2)
3  Deftext 1,0,0,13+7*(P>0)
4  Fullw 2
5  Dim T(28,32)
6  F=320+160*(P>0)
7  E=32+16*(P>0)
8  Define 1,32
9  Line 29,0,29,F
10 Line 29,0,288,0
11 Line 288,0,288,F
12 For J=0 To 30
13   For I=0 To 23
14     T(I,J)=-1+(I=0)
15   Next I
16 Next J
17 Define 1,0
18 For I=1 To 23
19   For J=1 To 28
20     If I<9 Then
21       A=8*J+38
22       B=(E*I/2)
23       Deffill (I Mod 4^P)+1
24       Pbox A,B,A+B,B+(E/2)-
25       T(I,J)=I+(I+1)*(J=1)
26     Endif
27     If J>1 And I>=9
28       T(I,J)=0
29     Endif
30   Next J
31 Next I
32 Z=0
33 T=0
34 H=-1
35 K=-0.5
36 J=14
37 I=12
38 Y=8*J
39 Debut:
40 Color 1
41 Define 1,14
42 Line Y-15,F+7,Y+15,F+7
43 Bo=Mousek
44 If Bo<>0 Then
45   Y0=Y
46   Y=Y+(8*(Bo=1) And Y>30)-(
47     8*(Bo=2) And Y<290)
48   Color 0
49   Line Y0-20,F+7,Y0+20,F+7
50   Goto Debut
51 Pause 2
52 If J>2 And J<30 And I>1
53   Print At(J+5,I);" "
54 Endif
55 T=0
56 I=I+H
57 J=J+K
58 C=T(I-1,J)
59 If I=21 Then
60   Goto Test
61 Endif
62 If C=0
63   If J>2 And J<30 And I>1 T
64     hen
65       Print At(J+5,I);Chr$(79
66     )
67   Endif
68   Goto Debut
69 Endif
70 If C=-1 Then
71   T=1
72   Goto Renvoi1
73 Endif
74 If C>=1 Then
75   T(I-1,J)=0
76   T=0
77   Goto Renvoi2
78 Endif
79 Test:
80 If I>=21 And Abs(8*(J+5)-Y)
81   >40 Then
82     Color 1
83     For I=1 To 12
84       Sound 1,9,I,5,9
85     Next I
86     Sound 1,0
87     Deftext 1
88     Print At(1,1);"SCORE : ";
89     Z
90     R$=Input$(1)
91     End
92 Endif
93 H=-H
94 K=K+(Rnd-Rnd)*0.5
95 Renvoi1:
96 K=-K
97 Sound 1,9,2,4,2
98 Sound 1,0
99 Goto Debut
100 Renvoi2:
101 K=-(K-(2*(Rnd<0.5)+1)*Rnd/3
102 )
103 Z=Z+C
104 Sound 1,9,5,6,4
105 Sound 1,0
106 H=-H
107 Goto Debut

```

Plusieurs en un

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : deux joueurs.

Difficulté du jeu : variable, selon l'utilisation.

LE JEU :

Plusieurs jeux sont en réalité possibles avec ce programme qui va se comporter comme un crayon électronique vous permettant d'inscrire des symboles de couleur, en n'importe quel emplacement de votre feuille-écran de télévision. On pourra, par exemple, l'utiliser pour un jeu de morpion (le premier à réaliser un alignement de 5 pions identiques gagne) ou pour un GO (on réalise des encadrements des pions adverses) ou tout autre jeu, selon son imagination.

LE PROGRAMME :

Le terrain de jeu est délimité par un cadre mis en place par les lignes 19 à 27.

Le changement de joueur est effectué par la ligne 58.

Le joueur qui a le trait voit le symbole qu'il a choisi (X ou O) apparaître à l'emplacement de la souris (ligne 29).

Les lignes 39 et 40 permettent aux deux joueurs de choisir une case.

Les lignes 41 à 51 contrôlent si le déplacement est possible (case libre et non dépassement des limites).

La ligne 56 permet d'écrire sur une case libre.

POUR JOUER :

RUN provoque l'affichage des limites du terrain de la grille de jeu et du numéro du joueur, ainsi que des symboles correspondants que l'on devra déplacer sur l'écran à l'aide de la souris.

Il n'est pas possible de venir sur une case déjà occupée et il faudra cliquer sur la touche de gauche pour inscrire son propre symbole dans une case disponible.

Dès que cela a été fait, le symbole du second joueur apparaît et le joueur procédera de la même manière que son adversaire.

On peut cliquer sur la touche de droite pour changer de joueur.

Pour sortir du programme, il suffira de cliquer sur les 2 touches.

EXTENSIONS POSSIBLES :

On pourra utiliser l'écran tout entier comme terrain de jeu et réaliser des contrôles en fonction des différents jeux que l'on désire pratiquer : contrôle d'alignements de 5 pièces pour le morpion, contrôle du nombre de pièces capturées pour le GO.

```

1  Rem PLUSIEURS EN UN
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  Titlew 2,"PLUSIEURS EN UN"
6  A=48+28*(P=2)
7  B=16+16*(P=2)
8  L=460+230*(P=2)
9  H=320+160*(P>0)
10 Dim T(L/16,H/16),A$(2)
11 K=Random(2)
12 For I=0 To 1
13   A$(I)=Mki$(B)+Mki$(B)+Mki
$(0)+Mki$(0)+Mki$(1)
14   For J=0 To 15
15     Read F,G
16     A$(I)=A$(I)+Mki$(F)+Mki
$(G)
17   Next J
18 Next I
19 Define 1,16
20 Rbox A=8*(P=2),B=10*(P>0),A
+L,B+H-6*(P=2)
21 Define 1,1
22 For I=A To A+L Step 16
23   Line I,B+8,I,B+H+8
24 Next I
25 For J=B-16*(P>0) To B+H Ste
p 16
26   Line A,J,A+L,J
27 Next J
28 Deftext 1,1,0,7
29 Defmouse A$(K)
30 Debut:
31 Repeat
32   Bo=Mousek
33 Until Bo<>0
34 If Bo=3 Then
35   Goto Fin
36 Endif
37 Sound 1,10,3+4*K,5,8
38 Sound 1,0
39 U=Int(Mousex/16)-4-2*(P=2)
40 V=Int(Mousey/16)-2-(P=2)
41 If U<0 Or V<0 Or U>=Int((L-
16)/16) Or V>=((H-16)/16) T
hen
42   Goto Debut
43 Endif
44 If Bo=1 Then
45   If T(U,V)<>0 Then
46     Sound 1,9,2,2,8
47     Sound 1,0
48     Goto Debut
49   Else
50     T(U,V)=1
51   Endif
52   C=79+9*K
53   X=16*U+70+32*(P=2)
54   Y=16*V+44+16*(P=2)
55   Deftext 2-K
56   Text X,Y,Chr$(C)
57 Endif
58 K=Abs(K-1)
59 Color 2*K+2
60 Defmouse A$(K)
61 Goto Debut
62 Fin:
63 Deftext 1,0,0,13+7*(P>0)
64 Color 1
65 For I=1 To 6
66   Sound 1,9,2*I,5,9
67 Next I
68 Sound 1,0
69 R#=Input$(1)
70 Data 384,2016,4080,8184,818
4,8184,8184,4080,2016,384,0
,0,0,0,0,0
71 Data 0,384,960,1632,1632,16
32,1632,960,384,0,0,0,0,0,0
,0
72 Data 8060,16382,8188,4088,2
032,2032,4088,8188,16382,80
60,0,0,0,0,0,0
73 Data 0,3640,1584,864,448,44
8,864,1584,3640,0,0,0,0,0,0
,0

```

Othello

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : difficile.

LE JEU :

Devenu un grand classique des jeux sur micro-ordinateur, le jeu d'Othello ou Reversi gagne, même si comme ici il est pratiqué à deux joueurs, à être géré sur ordinateur, avec visualisation : en effet, le jeu consistant essentiellement en des retournements des pions de l'adversaire, ceci peut se révéler, à la longue, très fastidieux, surtout si vous êtes un joueur assidu. Le principe est que le joueur possédant les X doit pour pouvoir jouer, réussir à encadrer une ou plusieurs des pièces de son adversaire, dans une ou plusieurs directions ; s'il y réussit, les O seront alors retournés pour devenir des X (le retournement pourra être fait horizontalement, verticalement ou suivant les diagonales).

Le joueur ayant les O procédera de la même façon, et ceci jusqu'à ce que le damier soit rempli ou que la personne ne puisse plus jouer (si un joueur ne peut retourner des pions, il doit passer son tour).

LE PROGRAMME :

Les boucles emboîtées P (17-31) et K (20-30) dessinent le terrain de jeu et la ligne 44 affiche un X ou un O, selon le joueur qui commence.

La boucle P (81-112) contrôle la validité du coup, en utilisant une partie de programme qui se trouve aux lignes 95 à 105 (a-t-on pris un ou plusieurs ennemis en ligne 95 et retournement éventuel autorisé par la ligne 96 : on refait un balayage dans toutes les directions, si l'encadrement s'est révélé possible).

Les lignes 85 et 86 calculent les déplacements à effectuer pour explorer les 8 directions.

Les lignes 124 à 131 réalisent les déplacements.

La ligne 107 assure le changement de joueur, s'il y a eu retournement.

Les lignes 53 à 69 assurent la gestion de la souris

La ligne 70 contrôle la fin de partie.

POUR JOUER :

Après RUN, chaque joueur cliquera à son tour sur la case où il désire jouer ; si le coup est accepté, les pions sont retournés et l'on change de joueur ; si un coup n'est pas possible, on clique sur la touche de droite pour le changement de joueur. On peut arrêter à tout moment en cliquant simultanément sur les 2 touches.

EXTENSIONS POSSIBLES :

Avant tout, bien sûr, améliorer l'affichage avec des couleurs, une grille plus aérée.

On pourra aussi imaginer un jeu machine pour le ST.

Après avoir lancé le programme par RUN, il convient d'entrer la dimension de la grille (4, RET ou 6, 8 ...).

```

1 Rem OTHELLO
2 Pe=Abs(Xbios(4)-2)
3 Ph=1-(Pe>0)
4 Fullw 2
5 Deftext 1,0,0,13+7*(Pe>0)
6 Repeat
7   Clearw 2
8   Input "DIMENSION DE LA GR
   ILLE :";A
9   Until A<=10+(Pe=2) And A Mo
   d 2=0
10  Clearw 2
11  Dim T(A+2,A+2),A$(2)
12  U=32+16*(Pe>0)
13  For I=1 To A+1
14    Line 16,U*I-4,16+32*A,U*I
    -4
15    Line 32*I-16,U-4,32*I-16,
    U+U*A-4
16  Next I
17  For P=1 To A
18    Deftext 1
19    Print At(4*P+1,1);Chr$(47
    +P);At(1,2*P+1);Chr$(47+P)
20    For K=1 To A
21      If K=A/2+1 Then
22        If (P=A/2) Or (P=A/2+
        1) Then
23          Deftext 2-(P<>A/2)
24          Print At(4*K-3,2*P+
        1);Chr$(79-9*(P<>A/2))
25          Deftext 2-(P<>A/2+1
        )
26          Print At(4*K+1,2*P+
        1);Chr$(88+9*(P<>A/2))
27        Endif
28      Endif
29      T(P,K)=0
30    Next K
31  Next P
32  T(A/2,A/2)=79
33  T(A/2+1,A/2)=88
34  T(A/2,A/2+1)=88
35  T(A/2+1,A/2+1)=79
36  K=Random(2)
37  For I=1 To 2
38    A$(I)=Mki$(4)+Mki$(4)+Mki
    $(0)+Mki$(0)+Mki$(I+1)
39    For J=0 To 31
40      Read H
41      A$(I)=A$(I)+Mki$(H)
42    Next J
43  Next I
44  Defmouse A$(K+1)
45  So=2
46  Sx=2
47  Do
48    Deftext 1
49    Print At(6*A+12*(Pe=2),4)
    ;"O ";So;" "
50    Print At(6*A+12*(Pe=2),6)
    ;"X ";Sx;" "
51  Debut:
52  Sound 1,0
53  Repeat
54    Bo=Mousek
55    Until Bo<>0
56    If Bo=1 Then
57      Sound 1,10,3+4*K,5,8
58      Sound 1,0
59      U=Ph*(Int((Mousex-8)/(3
        2+16*(Pe=0)))+1)
60      V=Ph*(Int((Mousey-6)/16)
        -2*(Pe>0))
61      If Odd(U) Or Odd(V) Or
        Uk2 Or V<2 Or U>2*A Or V>2*
        A Then
62        Goto Debut
63      Endif
64      If T(U/2,V/2)<>0 Then
65        Sound 1,9,2,2,8
66        Goto Debut
67      Endif
68      Gosub Test(1,1,0)
69      Endif
70      Exit If Bo=3 Or So+Sx=A*A
71      K=Abs(K-1)
72      Defmouse A$(K+1)
73      Loop
74      For I=1 To 6
75        Sound 1,9,2*I,5,9
76      Next I
77      Sound 1,0
78      Print "C'EST FINI..."
79      R$=Input$(1)
80      Procedure Test(H,P,S)
81        Suite:
82        W=0
83        T=U
84        R=V
85        F=(P>3 And P<7)-(P<3 Or P
        >7)
86        B=(P>1 And P<5)-(P>=6)
87        Q=32-56*(S=1)-47*(S=2)
88        If S<>0 Then
89          Gosub Affiche(1)
90        Endif
91        Essai:
92        T=T+2*F
93        R=R+2*G
94        Co=T(T/2,R/2)
95        If Co=79 Or Co=88 Then
96          If (Co=79 And K=1) Or (
        Co=88 And K=0) Then
97            W=1
98          Endif
99          If S<>0 Then
100            Gosub Affiche(-1)
101          Endif
102          If W=1 And ((Co=79 And
        K=1) Or (Co=88 And K=0)) Th
        en
103            Goto Essai
104          Endif
105          If Not (W=0 Or S<>0) Th
        en
106            S=1-(K=0)
107            H=-H*(S=0)
108            Goto Suite
109          Endif
110        Endif
111        P=P+1
112        If P<9 Then
113          S=0
114          Goto Suite
115        Else
116          If H<>0 Then
117            Goto Fin
118          Endif
119        Endif
120        So=So-(K=0)
121        Sx=Sx-(K=1)
122        Fin:
123        Return
124      Procedure Affiche(M)
125        Deftext 2-(Q=88)
126        C=Q
127        Print At(2*T+1,R+1);Chr$(
        Q)
128        T(T/2,R/2)=C
129        So=So+M*((Co=88)-(Co=79))
130        Sx=Sx+M*((Co=79)-(Co=88))
131      Return
132      Data 384,2016,4080,8184,818
        4,8184,8184,4080,2016,384,0
        ,0,0,0,0,0
133      Data 0,384,960,1632,1632,16
        32,1632,960,384,0,0,0,0,0,0
        ,0
134      Data 8060,16382,8188,4088,2
        032,2032,4088,8188,16382,80
        60,0,0,0,0,0,0
135      Data 0,3640,1584,864,448,44
        8,864,1584,3640,0,0,0,0,0,0
        ,0

```

Tic tac toe

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion (deux joueurs).

Difficulté du jeu : facile.

LE JEU :

Un grand classique des jeux d'écoliers, proposé ici dans une version ne comportant pas le jeu machine ; chaque joueur devra placer à son tour un O ou un X dans une case libre de la grille de jeu, et le premier qui réussit un alignement de 3 symboles identiques a gagné (l'alignement peut être horizontal, vertical, ou suivant l'une des deux diagonales).

LE PROGRAMME :

Le terrain de jeu est dessiné à l'aide de la boucle I (14-17).

La ligne 20 s'occupe d'afficher le symbole correspondant au joueur qui a le trait.

Les lignes 24 à 30 contrôlent le déplacement de la souris sur le terrain de jeu.

La ligne 33 vérifie si la case où l'on veut jouer est libre, et si l'on désire jouer.

La ligne 37 enregistre le coup, et la ligne 42 réalise le changement de joueur.

POUR JOUER :

Le terrain de jeu sera affiché dès la frappe de RUN.

Le curseur de la souris apparaît alors sur l'écran ; le joueur peut le déplacer sur le terrain de jeu avec la souris.

Lorsqu'il arrive sur la case où il désire jouer, il suffit qu'il appuie sur la touche de droite, le coup sera refusé si la case n'est pas libre.

Si la case est libre, le symbole qui correspond à ce joueur est inscrit, et on change de joueur.

Lorsque la partie est finie, il convient de cliquer sur les 2 touches pour sortir du programme. On pourra ensuite faire une autre partie avec RUN.

EXTENSIONS POSSIBLES :

On pourrait réaliser après chaque coup un contrôle pour vérifier si l'un des joueurs a gagné. On peut aussi réaliser un jeu machine, mais il faudrait éviter qu'il soit par trop systématique (par exemple, la machine jouant toujours son premier coup sur la case centrale).

```

1  Rem TIC TAC TOE
2  P=Abs(Xbios(4)-2)
3  S=0
4  Dim T(3,3)
5  Fullw 2
6  Deftext 1,0,0,10+4*(P>0)
7  Print At(1,19);"CLIQUEZ SUR";At(1,21);"LA CASE CHOISIE"
8  K=Random(2)
9  Pe=32+16*(P>0)
10 Se=2-3*(P>0)
11 Te=30+15*(P>0)
12 Re=64+32*(P=2)
13 Defline 1,Pe
14 For I=1 To 4
15   Line Re*I+80-32*(P=2),29-30*(P>0),Re*I+80-32*(P=2),251+90*
(P>0)
16   Line 144,2*Pe*I-20-50*(P>0),336+100*(P=2),2*Pe*I-20-50*(P>
0)
17 Next I
18 Debut:
19 Text 0,6*Pe,"JOUEUR "
20 Deftext 2+K,0,0,Te
21 C=79+9*K
22 Text 86,6*Pe+4,Chr$(C)
23 Entree:
24 Bo=Mousek
25 If Not (Bo=3 Or S=9) Then
26   If Bo=1 Then
27     U=Int((((Mousex+Re)/(Re/2))-7)/2)+2*(P=2)
28     V=Int((((Mousey+32)/Pe)-Se)/2)
29   Endif
30   If Bo<>1 Or U>2 Or V>2 Or U<0 Or V<0 Then
31     Goto Entree
32   Endif
33   If T(U,V)<>0 Then
34     Goto Entree
35   Else
36     S=S+1
37     T(U,V)=C
38     Text Re*U+168+12*(P=2),2*Pe*V+88+3*(P=2),Chr$(T(U,V))
39   Endif
40   Sound 1,9,5*K+2,5,15
41   Sound 1,0
42   K=Abs(K-1)
43   Deftext 1,0,0,10+(P>0)
44   Goto Debut
45 Endif
46 Deftext 1,0,0,13+7*(P>0)
47 Print At(1,19);"C'EST FINI..."
48 For I=1 To 12
49   Sound 1,10,I,5,9
50 Next I
51 Sound 1,0
52 R$=Input$(1)

```


Dames

REFERENCES :

Difficulté de programmation : très difficile.

Catégorie de jeu : deux joueurs.

Difficulté du jeu : difficile.

LE JEU :

C'est le jeu classique de dames à la française, mais un peu incomplet puisqu'il ne prévoit pas la possibilité de faire des dames. La règle du jeu reste classique, mais les contrôles ont été un peu simplifiés. Les mouvements des pièces sont entrés en deux manœuvres (position de départ et direction du déplacement) et les prises sont réalisées automatiquement, si elles sont possibles.

En revanche, le changement de joueur n'étant pas effectué après une prise, il faudra éventuellement cliquer pour pouvoir rejouer s'il y avait eu saut ou si le coup n'était pas possible.

LE PROGRAMME :

Les boucles I (6-17) et J (8-16) dessinent le damier et définissent les positions initiales des pions noirs et des pions blancs. Les positions et la couleur sont dans une table C (X, Y).

L'impression des pions proprement dite est effectuée par un sous-programme aux lignes 97 à 100, et 101 à 109.

La ligne 26 indique le joueur qui a le trait.

Le coup est entré dans les lignes 28 à 72.

La ligne 61 réalise le changement de joueur (on clique sur les 2 touches).

La ligne 69 regarde si le déplacement est possible.

La ligne 72 traite le cas d'un déplacement et la ligne 80 celui d'un saut.

La ligne 74 assure le changement de joueur.

POUR JOUER :

Dès le lancement du programme par RUN, le damier est affiché avec ses pions, et un cercle de couleur apparaît sur le côté pour indiquer que le trait appartient au joueur correspondant.

On joue en cliquant sur la pièce à déplacer, qui s'affiche légèrement agrandie.

Puis on clique sur la flèche indiquant la direction de déplacement.

Le changement de joueur étant effectué automatiquement il faudra cliquer pour rejouer si le coup est refusé ou s'il y a eu prise. Le programme ne contrôle pas l'arrivée à dame, et n'interdit pas les déplacements en arrière.

EXTENSIONS POSSIBLES :

Il sera assez facile de commencer par améliorer les contrôles (changement de joueur, coup en arrière qui ne correspondrait pas à une prise, sortie de terrain). On pourra envisager la création de dames, leur déplacement, et éventuellement écrire un programme permettant au ST de se mêler avec de bonnes chances de réussite à vos passionnantes parties.

```

1  Rem DAMES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  F=6+3*(P=2)
5  Dim C(10,10)
6  For I=1 To 10
7    Print At(1,2*I-(P>0));Chr$(47
+I);At(F*I,1);Chr$(64+I)
8    For J=1 To 10
9      Gosub Rec
10     C(I,J)=-2*(J<5)-5*(J>6)
11     H=-(J<5)
12     If ((I+J) Mod 2)=1 And C(I,
J)<>0 Then
13       C=C(I,J)
14       Gosub Cer(8*I+1,2*J,8,C)
15     Endif
16   Next J
17 Next I
18 H=Random(2)
19 Repeat
20   I=Random(10)+1
21   J=Random(10)+1
22 Until C(I,J)<>0
23 H=(I+J) Mod 2
24 Debut:
25 C=5-3*H
26 Gosub Cer(96,5,8,C)
27 D=65+32*(P=2)
28 Print At(D,10);"CLIQUEZ";At(D,1
1);"LE PION"
29 Print At(D,12);"PUIS LA";At(D,1
3);"FLECHE"
30 Choix:
31 Bo=Mousek
32 If Bo=3 Then
33   H=1-H
34   Goto Debut
35 Endif
36 I=Int((Int((Mousex-2)/(4*F))-1)
/2)+1
37 J=Int((Int((Mousey-25-5*(P>0))/
16))/(1-(P=0)))
38 If Bo<>1 Or J<0 Or I<1 Or J>9 O
r I>10 Or ((I+J+1) Mod 2)=0 The
n
39   Goto Choix
40 Endif
41 If C(I,J+1)=0 Or C(I,J+1)=2+3*H
Then
42   Goto Choix
43 Endif
44 J=J+1
45 R=11+(P>0)
46 Gosub Cer(8*I+1,2*J,R,C)
47 Sound 1,10,3+3*H,5,8
48 Sound 1,0
49 Define 1,F,0,1
50 U=95*F
51 V=160+10*(P>0)
52 Line U,V,U-20,V-20
53 Line U,V,U-20,V+20
54 Line U,V,U+20,V-20
55 Line U,V,U+20,V+20
56 Sens:
57 Repeat
58   Bo=Mousek
59 Until Bo<>0
60 If Bo=3 Then
61   H=Abs(H-1)
62   Gosub Rec
63   Gosub Cer(U,V,8,C)
64   Pause 5
65   Goto Debut
66 Endif
67 S=2*(Mousex<U)+1
68 T=2*(Mousey<V)+1
69 If I+S<1 Or I+S>10 Or J+T<1 Or
J+T>10 Then
70   Goto Sens
71 Endif
72 If C(I+S,J+T)=0 Then
73   Gosub Ecris
74   H=Abs(H-1)
75   Goto Debut
76 Endif
77 If I+2*S<1 Or I+2*S>10 Or J+2*T
<1 Or J+2*T>10 Then
78   Goto Sens
79 Endif
80 If C(I+S,J+T)=2+3*H And C(I+2*S
,J+2*T)=0 Then
81   C(I,J)=0
82   Gosub Rec
83   I=I+S
84   J=J+T
85   Gosub Ecris
86   Goto Debut
87 Endif
88 Gosub Rec
89 Gosub Cer(8*I+1,2*J,8,C)
90 Goto Debut
91 Procedure Rec
92   Deffill 3+5*((I+J) Mod 2),2,2
+16*((I+J) Mod 2)
93   A=(48+24*(P=2))*I-20-12*(P=2)
94   B=(32+16*(P>0))*J-7-5*(P>0)
95   Pbox A,B,A+48+24*(P=2),B+32+1
6*(P>0)
96 Return
97 Procedure Cer(U,V,R,C)
98   Deffill C,H+1,2
99   Pcircle F*U-2,(16+8*(P>0))*V+
6,R
100 Return
101 Procedure Ecris
102   C(I,J)=0
103   Gosub Rec
104   I=I+S
105   J=J+T
106   C(I,J)=5-3*H
107   C=C(I,J)
108   Gosub Cer(8*I+1,2*J,8,C)
109 Return

```

Parcours du cavalier

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion.

Difficulté du jeu : difficile.

LE JEU :

Il se présente, ici, comme un jeu de hasard total, mais provient, en réalité, d'un problème assez difficile à résoudre : comment déplacer un cavalier sur un échiquier de façon qu'il passe une fois, et une seule, sur chacune des 64 cases du damier.

Le cavalier se déplace suivant les règles communément admises aux échecs, et sa position de départ est définie par le joueur ; en revanche, ses déplacements sont effectués au hasard. Arrivera-t-il à parcourir les 64 cases ?

LE PROGRAMME :

Les lignes 7 à 9 définissent les divers déplacements possibles, dans deux variables alphanumériques A\$ et B\$; les boucles emboîtées I, J (11-19) dessinent le terrain de jeu.

On clique sur la case de départ aux lignes 20 à 26.

La ligne 36 imprime le numéro du coup.

La ligne 40 permet le tirage au hasard d'un déplacement du cavalier.

La boucle (41-49) cherche si le déplacement est possible ; si ce n'est pas le cas, on essaie tous les autres (toujours dans le même ordre).

Les lignes 42 et 43 effectuent le déplacement.

La ligne 44 contrôle que l'on ne sorte pas du terrain de jeu.

La ligne 50 gère la fin de partie (cavalier bloqué) et affiche le score.

Les lignes 59 à 62 réalisent le coup et incrémentent le score.

POUR JOUER :

On lance le programme par l'ordre RUN. Le terrain de jeu s'affiche ; on choisit alors la position de départ en cliquant sur la case de départ. Le jeu commence immédiatement et il se déroule de façon entièrement automatique jusqu'à ce que le cavalier soit bloqué.

On quitte le programme en tapant la combinaison CSA.

EXTENSIONS POSSIBLES :

On peut imaginer de nombreuses variantes sur ce thème, ne serait-ce qu'en commençant par faire varier la taille des échiquiers ; on peut aussi améliorer considérablement le graphisme. Mais, le plus intéressant est d'arriver à définir un algorithme donnant la ou les solutions du problème initial...

```

1  Rem PARCOURS DU CAVALIER
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  F=8+4*(P=2)
5  S=1
6  T=0
7  A$="56653223"
8  B$="653223"+A$+"56"
9  A$=A$+A$
10 Dim T(8,8)
11 For I=1 To 8
12   Print At(1,2*I+1-(P>0));Chr$(48+I);At(F*I,1);Chr$(48+I)
13   For J=1 To 8
14     Deffill 2+3*((I+J) Mod 2),2,8*((I+J) Mod 2)
15     A=(64+32*(P=2))*I-30-17*(P=2)
16     B=(32+16*(P>0))*J+6+4*(P>0)
17     Pbox A,B,A+64+32*(P=2),B+32+16*(P>0)
18   Next J
19 Next I
20 Print At(1,20);"CLIQUEZ SUR LA CASE - COL LIGNE"
21 Repeat
22   Print At(27,20);Y;" ";At(38,20);X;" ";
23   Bo=Mousek
24   Y=Int((Int((Mousex-2)/(4*(F)-1)/2)+1)
25   X=Int((Int((Mousey-8-5*(P>0))/16))/(1-(P=0)))
26 Until Not (Bo<>1 Or X<1 Or Y<1 Or X>8 Or Y>8)
27 I=X
28 J=Y
29 T(I,J)=1
30 Do
31   If S>9 Then
32     S=0
33     T=T+1
34   Endif
35   Deftext 3-5*(P=2)
36   Print At(F*J-(P=2),2*I+2);Chr$(S+48)
37   If T<>0 Then
38     Print At(F*J-1-(P=2),2*I+2);Chr$(T+48)
39   Endif
40   Z=Random(8)+1
41   For H=Z To Z+8
42     A=I+Val(Mid$(A$,H,1))-4
43     B=J+Val(Mid$(B$,H,1))-4
44     If A<1 Or B<1 Or A>8 Or B>8 Then
45       Goto Fin
46     Endif
47     Exit If T(A,B)=0
48   Fin:
49   Next H
50   If H=Z+9 Then
51     Deftext 1
52     Print At(1,19);Spc(70);At(1,19);"TOTAL : ";T*10+S
53     For I=1 To 12
54       Sound 1,10,I,5,10
55     Next I
56     Sound 1,0
57     R$=Input$(1)
58   Endif
59   I=A
60   J=B
61   S=S+1
62   T(A,B)=1
63   Sound 1,10,4,5,6
64   Sound 1,0
65 Loop

```

Quatre à quatre

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion (un ou deux joueurs).

Difficulté du jeu : difficile.

LE JEU :

Il pourra être pratiqué à un ou plusieurs joueurs. Dans les deux cas, le but est d'arriver à échanger les pions X avec les pions O ; si l'on joue en solitaire, cet échange devra être réalisé avec le moins de coups possible. Les pions sont déplacés alternativement en respectant les règles de déplacement du cavalier aux échecs ; il n'y a pas de prise, et on ne peut passer son tour.

LE PROGRAMME :

Les lignes 8 à 11 définissent le terrain de jeu, selon l'écran choisi.

La boucle I (12-15) dessine le terrain de jeu, et les lignes 16 et 17 définissent les positions de départ.

La ligne 37 permet d'afficher O ou X, selon le joueur qui a le trait.

Les lignes 39 à 45 contrôlent le déplacement du curseur.

La ligne 48 regarde si l'on a choisi le symbole correct.

La ligne 63 efface la figure qui était sous le curseur.

La ligne 58 réalise le contrôle d'arrivée. Le résultat est imprimé par la ligne 67, le changement de joueur est réalisé par la ligne 64.

Les lignes 73 à 80 s'occupent de la fin de la partie.

POUR JOUER :

On lance le programme par RUN. Le terrain de jeu est affiché, ainsi que le symbole permettant d'identifier le joueur qui a le trait. Le joueur peut choisir le pion qu'il désire déplacer en manœuvrant le curseur grâce à la souris.

Pour déplacer son pion, il suffit de cliquer lorsque le curseur est sur ce pion.

Puis on clique de nouveau sur la case d'arrivée.

Le coup sera refusé si la case de départ n'est pas conforme ou si la case d'arrivée est occupée. Sinon, le coup est réalisé, et le trait passe à l'autre joueur, qui déplace le curseur pour l'amener sur l'un des pions, et joue de la même façon.

Le joueur gagnant est celui qui réussit à amener le premier tous ses pions dans les cases initialement occupées par l'adversaire. On peut interrompre la partie à tout moment en cliquant sur les 2 touches simultanément.

EXTENSIONS POSSIBLES :

On peut améliorer les contrôles, prévoir un test de fin, utiliser un terrain de jeu différent ; on pourra aussi envisager l'écriture d'un programme permettant au ST de participer au jeu avec de bonnes chances de l'emporter.

```

1  Rem QUATRE A QUATRE
2  P=Abs(Xbios(4)-2)
3  S=0
4  Fullw 2
5  Dim T(6,6)
6  E=32
7  Deftext 1,0,0,10+5*(P=2)
8  Defline 1,8,0,0
9  Pe=64+32*(P=2)
10 Pf=48+24*(P>0)
11 Te=30+16*(P>0)
12 For I=0 To 6
13   Line Pe*I,0,Pe*I,291+146*
(P>0)
14   Line 0,Pf*I,380+186*(P=2)
,Pf*I
15 Next I
16 Deffn A(I)=Pe*I+24+12*(P=2)
17 Deffn B(J)=Pf*J+38+20*(P>0)
18 Print At(E,2);"CLIQUEZ";At(
E,3);"LE PION"
19 K=Random(2)
20 For I=0 To 5
21   For J=0 To 5
22     T(I,J)=-79*(I<2 And J<2
)-88*(I>3 And J>3)
23     Deftext 2-(T(I,J)=79),0
,0,Te
24     X=Fn A(I)
25     Y=Fn B(J)
26     C=T(I,J)
27     Text X,Y,Chr$(C)
28   Next J
29 Next I
30 Do
31   Deftext 1,0,0,10+5*(P=2)
32   Sound 1,10,3+4*K,5,12
33   Sound 1,0
34   Print At(E,9+4*(P=1));"JO
UEUR "
35   Deftext 3-K,0,0,Te
36   C=79+9*K
37   Text 8*Pe+6,2*Pf+20,Chr$(
C)
38   Debut:
39   Bo=Mousek
40   If Bo=3 Then
41     Goto Fin
42   Endif
43   U=Int((((Mousex+94)/(Pe/2
))-3+3*(P=2))/2)
44   V=Int((((Mousey+64)/(Pf/2
))-3)/2)+(P>0)
45   If Bo=0 Or U>5 Or V>5 Or
U<0 Or V<0 Then
46     Goto Debut
47   Endif
48   If S=0 And T(U,V)=79+9*K
Then
49     Deftext 2-(T(U,V)=79),2
,0
50     X1=Fn A(U)
51     Y1=Fn B(V)
52     W=U
53     Z=V
54     Text X1,Y1,Chr$(T(U,V))
55     S=1
56     Goto Debut
57   Endif
58   If Not ((Abs(W-U)=1 And A
bs(Z-V)=2) Or (Abs(W-U)=2 A
nd Abs(Z-V)=1)) Then
59     Goto Debut
60   Endif
61   If S=1 And T(U,V)=0 Then
62     T(U,V)=C
63     Text X1,Y1,Chr$(32)
64     K=1-K
65     T(W,Z)=0
66     Deftext 2-(T(U,V)=79),0
67     Text Fn A(U),Fn B(V),Ch
r$(T(U,V))
68     S=0
69   Endif
70   Sound 1,9,4*S,2,15
71   Sound 1,0
72 Loop
73 Fin:
74 Deftext 1,0,0,13
75 Print At(1,19);"C'EST FINI"
76 For I=1 To 12
77   Sound 1,10,1,5,9
78 Next I
79 Sound 1,0
80 R$=Input$(1)

```

Dessin de cercles

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin avec tirage aléatoire.

Difficulté du jeu : aucune.

LE JEU :

Le principe est très élémentaire. On dessine des cercles dont le motif est aléatoire, en choisissant leur taille, et en précisant la position de leur centre à l'aide de la souris. Le programme est bâti suivant une méthode particulièrement intéressante sur ce type de machine, en utilisant seulement deux lignes.

LE PROGRAMME :

La ligne 7 affiche la taille du cercle (de 1 à 128) et la ligne 8 détecte les actions sur la souris.

La ligne 9 envoie dans les sous-programmes correspondants, et la ligne 10 reboucle sur la ligne 6.

Le dessin des cercles est fait par appel de la procédure Des (lignes 15 à 20).

POUR JOUER :

On lance le programme par RUN. L'écran s'efface et la taille initiale des cercles (rayon 1) apparaît en haut à gauche.

On clique sur le bouton de gauche pour changer de taille, on déplace la souris pour choisir la position du centre, et on clique à droite pour dessiner le cercle. On peut quitter le programme à tout moment en cliquant sur les 2 boutons.

EXTENSIONS POSSIBLES :

On pourra envisager des possibilités d'effacement, permettre le choix d'autres motifs, etc.

```

1  Rem DESSIN DE CERCLES
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Clearw 2
5  J=1
6  Do
7      Print At(1,1);"TAILLE :";J;"  "
8      Mouse X,Y,Bo
9      On Bo GOSUB Des,Tai,Fin
10 Loop
11 Procedure Tai
12     J=J+1+J*(J>127)
13     Pause 2
14 Return
15 Procedure Des
16     S=Random(2)+2
17     Deffill 1+Random(4^P),S,Int(Rnd*(23+12*(S=1))+1)
18     Bo=0
19     Pcircle X,Y,J
20 Return
21 Procedure Fin
22     For I=1 To 12
23         Sound 1,10,I,5,10
24     Next I
25     Sound 1,0
26     R$=Input$(1)
27     End
28 Return

```


Taquin de lettres

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : réflexion.

Difficulté du jeu : difficile.

LE JEU :

Qui d'entre nous n'a pas, un jour ou l'autre, eu dans les mains un de ces petits puzzles où l'on essaie de déplacer d'une case à une autre des petits carrés de plastique colorés, jusqu'à l'obtention d'un résultat bien précis ? En son temps le 14-15 de Sam Loyd, a eu autant de succès que notre célèbre RUBIK'S.

Le taquin qui vous est proposé ici utilise 35 lettres, disposées au hasard dans un carré de 6 cases sur 6 cases, et ces lettres peuvent, replacées dans leur ordre initial, vous permettre de reconstituer un proverbe très connu. Une case vide vous permet de déplacer les lettres, et un compteur affiche en permanence le nombre d'essais : arriverez-vous à retrouver le proverbe ?

LE PROGRAMME :

La phrase à reconstituer (qui pourra bien sûr être modifiée) est placée dans une variable R\$ en ligne 8.

Le mélange des lettres et l'impression sont faits par les boucles I (12-22) et (23-29).

Les lignes 35 à 44 regardent si l'on a cliqué sur une lettre.

Les lignes 51 à 78 réimpriment les lettres déplacées, si le coup est possible.

La ligne 79 comptabilise le nombre d'essais qui est imprimé grâce à la ligne 40.

POUR JOUER :

On lance le programme par RUN.

La partie sera évidemment d'autant plus difficile que l'on ne connaît pas le proverbe à reconstituer. On peut déplacer les lettres et les faire passer à l'emplacement du carré vide à l'aide de la souris (on clique sur la lettre concernée).

Le nombre d'essais est affiché en permanence, à côté du puzzle.

On peut arrêter à tout moment en cliquant sur les 2 touches.

EXTENSIONS POSSIBLES :

Le même principe peut être appliqué à toute sorte de taquins (manipulation de nombres, de couleurs, de sons éventuellement). On peut aussi prévoir plusieurs phrases, avec un choix effectué par un générateur aléatoire, etc.

```

1 Rem TAQUIN DE LETTRES
2 P=Abs(Xbios(4)-2)
3 Defext 1,0,0,13+7*(P>0)
4 Fullw 2
5 Openw 2
6 Dim T(6,6)
7 S=1
8 R$="DNASOUVENTBESOIND'UNPLU
  SPETITQUESOI "
9 A$=""
10 Pe=32+8*(P>0)
11 Se=2-(P>0)
12 For I=0 To 35
13   X=Random(Len(R$))+1
14   A$=A$+Mid$(R$,X,1)
15   F$=Left$(R$,X-1)
16   R$=F$+Mid$(R$,X+1)
17   T((I Mod 6)+1,Int(I/6)+1)
    =Asc(Mid$(A$,I+1,1))
18   If Asc(Mid$(A$,I+1,1))=32
      Then
19     U=(I Mod 6)+1
20     V=Int(I/6)+1
21   Endif
22 Next I
23 For I=1 To 6
24   For J=1 To 6
25     Rbox 24*I+24,Pe*J-9*(P=
      0),24*I+48,Pe*J+Pe-9*(P=0)
26     Defext 1+(Asc(Chr$(T(I
      ,J))) Mod 4^P)
27     Print At(3*I+5,Se*J+2);
      Chr$(T(I,J))
28   Next J
29 Next I
30 Debut:
31 For I=1 To 4
32   Sound 1,9,3*I,6,1
33 Next I
34 Sound 1,0
35 Lis_souris:
36 Mouse Xo,Yo,Bo
37 If Bo=3 Then
38   Goto Fin
39 Endif
40 Print At(28,7);"ESSAI :";S
41 X=Int((Xo-24)/24)
42 Y=Int((Yo-8)/Pe)
43 If Bo=0 Or X<1 Or Y<1 Or X>
  6 Or Y>6 Then
44   Goto Lis_souris
45 Endif
46 If X<>U And Y<>V Then
47   Sound 1,9,2,2,10
48   Sound 1,0
49   Goto Debut
50 Endif
51 If U=X And V<Y Then
52   For I=V To Y-1
53     T(U,I)=T(U,I+1)
54   Gosub Exchange(U,I,Se,Ch
    r$(T(U,I)))
55   Next I
56 Endif
57 If U=X And Y<V Then
58   For I=V To Y+1 Step -1
59     T(U,I)=T(U,I-1)
60   Gosub Exchange(U,I,Se,Ch
    r$(T(U,I)))
61   Next I
62 Endif
63 If V=Y And U<X Then
64   For I=U To X-1
65     T(I,V)=T(I+1,V)
66   Gosub Exchange(I,V,Se,Ch
    r$(T(I,V)))
67   Next I
68 Endif
69 If V=Y And X<U Then
70   For I=U To X+1 Step -1
71     T(I,V)=T(I-1,V)
72   Gosub Exchange(I,V,Se,Ch
    r$(T(I,V)))
73   Next I
74 Endif
75 T(X,Y)=32
76 U=X
77 V=Y
78 Gosub Exchange(U,V,Se,Chr$(T
  (X,Y)))
79 S=S+1
80 Goto Debut
81 Fin:
82 For I=1 To 12
83   Sound 1,10,I,5,10
84 Next I
85 Sound 1,0
86 Print At(1,19);"C'EST FINI"
87 R$=Input$(1)
88 Procedure Exchange(A,B,C,T$)
89   Defext 1+(Asc(T$) Mod 4^
    P)
90   Print At(3*A+5,C*B+2);T$
91 Return

```

Œufs

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : solitaire.

Difficulté du jeu : moyenne.

LE JEU :

Un oiseau mystérieux se déplace en battant des ailes au hasard sur l'écran. Il pond des œufs (représentés par des O) que vous (un carré de couleur) devez essayer de récupérer. Votre score (nombre d'œufs récupérés) sera affiché en permanence, et la partie se déroule en un laps de temps défini.

LE PROGRAMME :

L'oiseau est dessiné en deux parties (pour le mouvement) aux lignes 19 et 43 ; son déplacement aléatoire est géré par le sous-programme 56-69 (en particulier les lignes 65 à 68).

La ligne 45 permet de reboucler le programme sur la ligne 17.

Le déplacement du joueur est régi par les lignes 22 à 25, et le test permettant de détecter la présence d'un œuf est effectué en ligne 37.

La ligne 54 permet l'affichage du score.

La fin de jeu est détectée en ligne 27 ou 58.

POUR JOUER :

On lancera le programme par RUN.

L'oiseau commence alors à se déplacer au hasard sur l'écran, et se met à pondre ses œufs. Lorsque le symbole qui représente le joueur (un carré de couleur) apparaît sur l'écran, celui-ci peut se déplacer en utilisant la souris. Il vous faut dès lors essayer de récupérer un maximum d'œufs avant la fin de la partie, en cliquant lorsque le carré est sur un œuf.

On pourra arrêter de jouer à tout moment en cliquant sur les 2 touches.

EXTENSIONS POSSIBLES :

Sur le même principe, on pourra facilement imaginer d'autres thèmes de jeu. On pourra bien sûr améliorer la couleur, les sons, le graphisme, et éventuellement, définir un jeu à plusieurs joueurs, chacun essayant de récupérer un maximum d'œufs.

```

1  Rem DEUFS
2  Fullw 2
3  Clearw 2
4  Dim T(80,25)
5  P=500
6  A$=Mki$(B)+Mki$(B)+Mki$(1)+
   Mki$(0)+Mki$(3)
7  For I=0 To 15
8    A$=A$+Mki$(0)
9  Next I
10 For I=0 To 15
11   Read H
12   A$=A$+Mki$(H)
13 Next I
14 Defmouse A$
15 U=Random(30)+5
16 V=Random(13)+6
17 Do
18   Deftext 2
19   Print At(U,V);Chr$(95)+Ch
   r$(234)+Chr$(95)
20   For J=1 To 5
21     B=Mousek
22     X=Mousex
23     Y=Mousey
24     Xo=Abs(Int(X/B)*(X<640)
   )
25     Yo=Abs(Int(Y/(16+B*(Xbi
   os(4)<2)))*(Y<400))
26   Deftext 1
27   If B=3 Then
28     Print "C'EST FINI"
29     For I=1 To 9
30       Sound 1,9,I,5,8
31     Next I
32     Sound 1,0
33     R$=Input$(1)
34     End
35   Else
36     If B<>0
37       Gosub Tes
38     Endif
39   Endif
40 Next J
41 Gosub Mou
42 Deftext 2
43 Print At(U,V);Chr$(92)+Ch
   r$(234)+Chr$(47)
44 Gosub Mou
45 Loop
46 Procedure Tes
47   If T(Xo,Yo)=42 Then
48     T=T+1
49     Sound 1,9,4,5,3
50     Sound 1,0
51   Endif
52   Print At(Xo,Yo);Chr$(32);
53   T(Xo,Yo)=0
54   Print At(1,19);"SCORE : "
   ;T
55 Return
56 Procedure Mou
57   P=P-1
58   If P=0 Then
59     End
60   Endif
61   Print At(U,V);"  "
62   Deftext 3
63   Print At(U+1,V);Chr$(42)
64   T(U+1,V)=42
65   U=Abs(U+Int(1+2*(Rnd<0.5)
   ))
66   V=Abs(V+Int(1+2*(Rnd<0.5)
   ))
67   U=U*(U<40)*(U>1)-(U>39)-3
   9*(U<2)
68   V=V*(V<21)*(V>4)-5*(V>20)
   -20*(V<5)
69 Return
70 Data &h1c1c,&h1ffc,&hffb8,&h
   7f0,&h6b0,&h7f0,&h360,&h1c0
71 Data &he38,&h1774,&h17f4,&h
   3776,&h7f0,&h220,&ha28,&he3
   8

```

Création de formes

REFERENCES :

Difficulté de programmation : assez facile.

Catégorie de jeu : préparation d'un dessin 16×16 et calcul des codes.

Difficulté du jeu : aucune.

LE JEU :

Le programme facilite la création de formes, en utilisant un maximum de 16 cases sur 16, ce qui correspond à la définition de la forme de la souris. On pourra bien entendu aussi l'utiliser pour redéfinir les caractères, en travaillant en 8×8 ou en 8×16 . Les codes correspondant seront affichés en fin de création.

LE PROGRAMME :

La ligne 9 dessine un carré destiné à l'affichage de la forme en taille réelle.

La boucle I (11-17) affiche la grille de dessin.

Les lignes 19 à 23 contrôlent l'utilisation de la souris.

Les lignes 25 et 26 écrivent ou effacent une case, selon le bouton de la souris sur lequel on appuie.

La ligne 35 calcule les codes, affichés en ligne 39.

POUR JOUER :

On lance le programme par RUN. Puis on déplace la souris dans la case concernée, et on clique à droite ou à gauche, pour écrire ou effacer.

On clique sur les 2 boutons simultanément pour afficher les codes et quitter le programme.

EXTENSIONS POSSIBLES :

On pourra ajouter la possibilité de choisir la couleur de chaque case, travailler sur des grilles plus grandes, ajouter le dessin d'un masque pour la souris, sauvegarder l'icône obtenue.

```

1  Rem CREATION DE FORMES 16*16
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  A=263
5  B=73
6  Dim T(16,16),A(16)
7  Clearw 2
8  Print At(32,2);"FORME"
9  Box A-2,B-2,A+19,B+19
10 Deffill 1,2,1
11 For I=1 To 16
12   A(I)=0
13   For J=1 To 16
14     Pbox 11*I+11,11*J-11,11*I+22,11*J
15     T(I,J)=0
16   Next J
17 Next I
18 Do
19   Bo=Mousek
20   Exit If Bo=3
21   I=Int(Mousex/11)-1
22   J=Int(Mousey/11)+1
23   If Not (Bo=0 Or I<1 Or J<1 Or I>16 Or J>16) Then
24     C=Bo+Bo*(Bo=2)
25     Deffill 1,2,8*C
26     Pbox 11*I+11,11*J-11,11*I+22,11*J
27     Color C
28     Plot A+I,B+J
29     T(I,J)=C
30   Endif
31 Loop
32 Print At(1,14);
33 For I=1 To 16
34   For J=1 To 16
35     A(I)=A(I)+T(J,I)*(2^(16-J))
36   Next J
37   Print Using "LIGNE ##",I;
38   Print " :";
39   Print Using "#####",A(I);
40   Print " - ";
41   If P=2 And I Mod 2=0 Then
42     Print
43   Endif
44 Next I
45 For I=1 To 12
46   Sound 1,10,I,5,10
47 Next I
48 Sound 1,0
49 R$=Input$(1)

```

Ardoise magique

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin.

Difficulté du jeu : facile.

LE JEU :

Comme le titre peut sembler l'indiquer, il s'agit tout simplement d'utiliser l'écran du téléviseur comme une ardoise, en déplaçant une main dans toutes les directions, avec la souris.

On dispose aussi de la possibilité de déplacer un trait sur l'écran sans laisser de trace en cliquant sur la touche de gauche.

On peut choisir la couleur du point en appuyant sur la touche ESC et il est possible de remplir une surface fermée en tapant Ø.

L'appui sur la touche 1 permet de tracer un rectangle à partir de la position souris, et 2 permet d'obtenir un cercle dont le rayon est donné par la position souris.

LE PROGRAMME :

La ligne 10 permet de choisir la main comme curseur.

La ligne 18 affiche les choix de couleur.

Les lignes 21 et 37 contrôlent les entrées.

La ligne 24 affiche une ligne.

La ligne 28 affiche un point.

La ligne 40 effectue le remplissage s'il y a lieu.

La ligne 51 s'occupe du choix de la couleur.

La fin du programme est détectée en ligne 53.

La ligne 45 dessine un rectangle, et la ligne 48 un cercle.

POUR JOUER :

On lance le programme par RUN ; une main apparaît sur l'écran. On devra cliquer à droite pour marquer un point, et à gauche pour tracer un trait.

Si on frappe la touche ESC, la couleur du point change.

On peut sortir à tout moment, en tapant la barre d'espace.

On change la position du trait en maintenant appuyées les 2 touches. On tape 1 pour tracer un rectangle, et 2 pour un cercle.

EXTENSIONS POSSIBLES :

On pourra, bien sûr, envisager d'écrire un programme identique en utilisant d'autres types de tracés, prévoir un sous-programme qui permette la sauvegarde du dessin réalisé pour une utilisation ultérieure, etc.

```

1  Rem ARDOISE MAGIQUE
2  Fullw 2
3  Clearw 2
4  P=Abs(Xbios(4)-2)
5  C=2-2*(P>0)-12*(P=2)
6  U=640+320*(P=2)
7  V=400+200*(P>0)
8  Co=1
9  Cf=0
10 Defmouse 3
11 Color 1
12 Line 0,10-10*(P=0),640,10-1
   0*(P=0)
13 Deftext 1,0,0,13+7*(P>0)+2*
   (P=2)
14 Print At(19,1);"O:REPLIR
   1:RECTANGLE 2:CERCLE"
15 X=100
16 Y=100
17 Do
18   Print At(1,1);"ENCRE ";Co
   ;"   ";At(10,1);"FOND ";Cf
19   Cf=Point(X,Y)
20   Test:
21   Mouse X,Y,Z
22   If Z=1 Or Z=3 Then
23     Color Co
24     Line X0,Y0,X,Y
25   Endif
26   If Z=2 Then
27     Color Co
28     Line X,Y,X,Y
29     X0=X
30     Y0=Y
31   Endif
32   If Z=3 Then
33     Color Cf
34     Line X0,Y0,X,Y
35   Endif
36   For I=1 To 20
37     R=Asc(Right$(Inkey$,1))
38     Exit If R<>0
39   Next I
40   If R=48 Then
41     Deffill Co
42     Fill X,Y
43   Endif
44   If R=49 Then
45     Box X0,Y0,X,Y
46   Endif
47   If R=50 Then
48     Circle X0,Y0,Sqr((X-X0)
   ^2+(Y-Y0)^2)
49   Endif
50   If R=27 Then
51     Co=(Co+1) Mod C
52   Endif
53   Exit If R=32
54   If Z<>0 Then
55     Goto Test
56   Endif
57 Loop
58 Fin:
59 Defmouse 0
60 For I=1 To 12
61   Sound 1,10,I,5,10
62 Next I
63 Sound 1,0
64 R$=Input$(1)

```


Fais-moi un dessin

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : création d'un dessin.

Difficulté du jeu : aucune.

LE JEU :

Le programme permet de confectionner son propre paysage, à partir d'icônes prédéfinies apparaissant dans des cases en haut de l'écran. Le programme initial propose 8 ou 10 icônes, avec la possibilité de les colorer et de changer leur taille.

LE PROGRAMME :

La ligne 25 permet de dessiner les 8 ou 10 icônes, en utilisant un sous-programme, en lignes 80 à 91.

Les lignes 30 à 34 gèrent la position de la souris et l'état des touches.

La ligne 35 contrôle le choix de l'icône.

Les lignes 47 à 71 permettent de la choisir, et l'affiche.

La ligne 57 permet de la déposer.

Les dessins des icônes sont contenus dans des DATA, de la ligne 93 à la ligne 114.

POUR JOUER :

On lance le programme par RUN.

Les icônes sont affichées dans des carrés en haut à gauche : on clique sur le bouton de droite pour choisir l'icône. On clique ensuite sur les symboles + ou - pour choisir largeur, hauteur et couleur. Puis on la dépose à l'emplacement choisi en cliquant à gauche. On tapera la combinaison CSA pour quitter le programme.

EXTENSIONS POSSIBLES :

On pourra augmenter le nombre d'icônes disponibles, améliorer les possibilités de déformation, etc.

```

1 Rem FAIS MOI UN DESSIN
2 P=Abs(Xbios(4)-2)
3 Pe=1-(P=2)
4 Fullw 2
5 Clearw 2
6 Titlew 2,"FAIS MOI UN DESSIN"
7 Dim T(10,37),M(3),A(16),B(16),
  M$(3)
8 E=0
9 Line 0,40,640,40
10 Deftext 1,0,0,6
11 For I=2 To 4
12   Read M$(I-1),M(I-1)
13   Print At(59+29*(Pe=2),I);"+
  - ";Mid$(M$(I-1),1,7+4*(Pe=2))
14 Next I
15 For I=0 To 10+2*(Pe=2)
16   For J=1 To 34
17     Read T(I,J)
18   Next J
19 Next I
20 For K=0 To 10+2*(Pe=2)
21   X=- (40*(K+24)*(Pe=1)-(24*(K+12)
  )*(Pe=2))
22   Y=22
23   Box X-8/Pe,Y-15,X+22+4*(Pe=2)
  ,Y+5
24   I=K
25   Gosub Dessin
26 Next K
27 Debut:
28 Print At(78/Pe,2);M(1);" ";At(
  78/Pe,3);M(2);" "
29 Print At(78/Pe,4);M(3);" "
30 Repeat
31   Bo=Mousek
32 Until Bo<>0
33 Xo=Int((Mousex-11)/(10+4*(Pe=2)
  ))-(Pe=2)
34 Yo=Int(Mousey/9)
35 If (Xo=45+7*(Pe=2) Or Xo=47+6*(
  Pe=2)) And (Yo=1 Or Yo=2 Or Y
  o=3) Then
36   G=Xo-45-7*(Pe=2)
37   M(Yo)=Abs(M(Yo)+1 And G=0)-
  (1 And (G=2 Or G=3))
38   If Yo=3 Then
39     M(Yo)=M(Yo) Mod 16
40   Endif
41 Endif
42 If E=0 And ((Xo Mod 4)=0 Or Yo
  <1 Or Yo>4 Or Xo>44+7*(Pe=2))
  Then
43   Goto Debut
44 Else
45   Xo=Int(Xo/4)
46 Endif
47 If E=1 And Bo=2 Then
48   Y=Mousey
49   If Y<40 And Mousex<480+258*(
  Pe=2) Then
50     E=0
51     X=P1
52     Gosub Param
53   Endif
54   If Y>40 Then
55     E=0
56     X=Mousex
57     Gosub Dessin
58     X=P1
59     Gosub Param
60     Goto Debut
61   Endif
62 Endif
63 If E=0 And Y<40 Then
64   X=- (40*(Xo+24)*(Pe=1)-(24*(Xo+
  12)*(Pe=2))
65   P1=X
66   I=Xo
67   M(3)=2
68   Gosub Dessin
69   E=1
70 Endif
71 Goto Debut
72 Procedure Param
73   X=P1
74   Y=22
75   M(1)=1
76   M(2)=1
77   M(3)=4
78   Gosub Dessin
79 Return
80 Procedure Dessin
81   M=M(1)
82   P=-M(2)
83   Deffill M(3)
84   A(0)=X
85   B(0)=Y
86   For J=1 To 16
87     A(J)=A(J-1)+M*T(I,2*(J-1))
88     B(J)=B(J-1)+P*T(I,2*(J-1))
89   Next J
90   Polyfill 17,A(),B()
91 Return
92 Data Largeur,1,Hauteur,1,Coule
  ur,4
93 Data 2,1,0,5,-3,-3,-1,1,4,-2
  ,3,8,0,-2,-3,4,-4,-1,-1,-3,3,0
94 Data -5,2,-1,-3,0,-1,3,-1,-3,-
  3,0
95 Data 2,3,0,4,-3,-2,2,4,-2,-2,3
  ,3,3,-3,-2,2,2,-4,-3,2,0
96 Data -4,2,-3,-4,0,0,0,0,0,0,0
  ,0
97 Data 4,4,3,-3,1,1,-1,3,3,3,-
  3,-3,2,-3,-3,1,-1,-5,0
98 Data 1,1,-3,3,-3,-3,0,0,0,0,0
  ,0,0
99 Data 0,6,2,2,5,0,0,2,2,0,0,-2,
  2,0,2,-2,0
100 Data -6,-3,0,0,4,-3,0,0,-4,-7,
  0,0,0,0,0,0,0
101 Data 3,7,6,0,2,-3,3,-1,3,-2,-3
  ,0,-2,-2,-2,2,-5,0
102 Data -2,-2,-2,2,-2,0,0,0,0,0,0
  ,0,0,0,0,0
103 Data 0,9,1,1,1,-1,0,-2,5,0,0,-
  2,-5,0,0,-3,5,0
104 Data 0,-2,-5,0,0,-1,-2,0,0,0,0
  ,0,0,0,0,0
105 Data 0,8,11,0,0,-4,4,0,0,-4,-2
  ,0,-2,-2,-1,0
106 Data -2,2,-2,0,-2,-2,-1,0,-2,2
  ,-1,0,0,0,0,0,0,0
107 Data 1,3,-3,2,2,0,2,1,2,2,-3,0
  ,3,2,1,2,1,-4
108 Data 3,-1,-3,-1,-3,-3,-2,-3,2,
  0,-3,0,0,0,0,0,0
109 Data 3,2,-2,3,3,-2,3,2,-1,-3,2
  ,-2,-3,0,-2,-2,0
110 Data 2,-2,0,0,0,0,0,0,0,0,0,0,
  0,0,0,0,0
111 Data 0,2,14,0,0,-2,-14,0,0,0,0,
  0,0,0,0,0,0
112 Data 0,0,0,0,0,0,0,0,0,0,0,0,0
  ,0,0,0,0
113 Data 0,10,3,0,0,-10,-3,0,0,0,0,
  0,0,0,0,0,0
114 Data 0,0,0,0,0,0,0,0,0,0,0,0,0
  ,0,0,0,0

```

Echecs

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de réflexion, pour deux joueurs.

Difficulté du jeu : difficile.

LE JEU :

Le programme propose seulement à deux joueurs de disputer une partie en se servant de l'écran du téléviseur comme d'un plateau de jeu. Il n'y a pas de contrôle de la validité des coups, mais seulement gestion de la position des pièces sur l'échiquier. Les pions et les pièces sont dessinés à l'aide de la redéfinition des caractères. La couleur et le dessin des pièces permettent de distinguer chaque camp.

LE PROGRAMME :

Les lignes 28 à 38 dessinent l'échiquier. Les lignes 32, 33, 36 et 37 définissent les conditions initiales, à l'aide de deux tableaux : R (I, J) pour les codes des pièces, et C (I, J) pour la couleur.

Les boucles emboîtées I et J (39-45) dessinent les pièces.

Les lignes 52 à 75 permettent l'entrée de la ligne et de la colonne de départ, avec contrôle en ligne 75.

La ligne 79 choisit la pièce et la ligne 80 efface la case.

La ligne 71 assure le changement de joueur.

POUR JOUER :

On lance l'exécution par RUN.

L'échiquier est dessiné, puis les pièces, et un message indique le joueur qui a le trait ; on place le curseur sur la pièce à déplacer, on clique à droite ; puis on choisit la position d'arrivée et on clique à gauche. Le déplacement est alors effectué, et c'est au tour de l'autre joueur.

Pour arrêter, il suffit de cliquer simultanément sur les 2 touches.

EXTENSIONS POSSIBLES :

On pourra chercher à réaliser un graphisme plus précis, introduire le contrôle du mouvement des pièces. Et pourquoi pas, écrire un programme qui permette au ST de se comporter en partenaire honorable. Bon courage !

```

1  Rem ECHECS
2  P=Abs(Xbios(4)-2)
3  Fullw 2
4  Deftext 1,0,0,13+7*(P>0)
5  F=8+4*(P=2)
6  Dim A$(2),P$(32),R(8,8),C(8,8)
7  For K=0 To 1
8      For I=1 To 6
9          P$(I+16*K)=Mki$(0)+Mki$(0)
          +Mki$(0)+Mki$(0)+Mki$(1-K*(P=1
          )-3*K*(P=2))
10         For J=0 To 15
11             Read H
12             P$(I+16*K)=P$(I+16*K)+Mk
            i$(65535)+Mki$(H)
13         Next J
14     Next I
15     For I=9+16*K To 16+16*K
16         P$(I)=P$(6+16*K)
17     Next I
18     P$(6+16*K)=P$(3+16*K)
19     P$(7+16*K)=P$(4+16*K)
20     P$(8+16*K)=P$(5+16*K)
21     If P>0 Then
22         Restore
23     Endif
24 Next K
25 Restore Ordre
26 A$(0)="BLANC"
27 A$(1)="NOIR"
28 For I=1 To 8
29     Print At (2+(P=2),2*I);Chr$(4
    8+I);At((8+4*(P=2))*I,1);Chr$(
    64+I)
30     For J=1 To 8
31         Gosub Rec(I,J)
32         R(I,J)=-(I+8)*(J=2)-(I+24)
        *(J=7)
33         C(I,J)=-3*(J>6)-2*(J<3)
34     Next J
35     Read H
36     R(I,1)=H
37     R(I,8)=R(I,1)+16
38 Next I
39 For J=1 To 8
40     For I=1 To 8
41         If R(I,J)<>0
42             Sprite P$(R(I,J)),8*(F*I
            -1),(16+8*(P>0))*(2*I+2)

```

```

43      Endif
44      Next I
45  Next J
46  H=0
47  G=0
48  E=0
49  Debut:
50  Print At(1,19);"JR ";A$(G);" -
      CLIQUEZ SUR LA PIECE  "
51  Do
52      Repeat
53          Bo=Mousek
54          If Bo=3 Then
55              Goto Fin
56          Endif
57          Y=Int((Int((Mousex-2)/(4*(F
      ))-1)/2)+1
58          X=Int((Int((Mousey+16+12*(
      P>0))/16))/(1-(P=0)))
59          If E=1 Then
60              Sprite P$(K),Mousex-10,M
      ousey+24+12*(P>0)
61          Endif
62          If E=1 And Bo=2 Then
63              If R(Y,X)<>0
64                  Sprite P$(R(Y,X)),600+
      (330-16*(H\8))*(P=2),30+20*(H
      Mod 8)
65              H=H+1
66          Endif
67          E=0
68          R(Y,X)=K
69          C(Y,X)=S
70          Sprite P$(K),8*(F*Y-1),(
      16+8*(P>0))*(2*X+2)
71          G=1-G
72          Showm
73          Goto Debut
74      Endif
75      Until Not (Bo=0 Or X<1 Or Y<
      1 Or X>8 Or Y>8 Or E=1)
76      Exit If E=0 And ((C(Y,X)=2 A
      nd G=1) Or (C(Y,X)=3 And G=0)
      Or R(Y,X)=0)
77  Loop
78  If E=0 And R(Y,X)<>0 Then
79      Sprite P$(R(Y,X))
80      Gosub Rec(Y,X)
81      K=R(Y,X)
82      S=C(Y,X)

```

```

83   C(Y,X)=0
84   R(Y,X)=0
85   E=1
86   Hidem
87 Endif
88 Goto Debut
89 Fin:
90 For I=1 To 12
91   Sound 1,10,I,5,10
92 Next I
93 Sound 1,0
94 R$=Input$(1)
95 Procedure Rec(I,J)
96   Deffill 1+2*((I+J) Mod 2),2,
     2-6*((I+J) Mod 2)
97   A=8*F*I-30-8*(P=2)
98   B=(32+16*(P>0))*J-16-8*(P>0)
99   Pbox A,B,A+8*F,B+32+16*(P>0)
100 Return
101 Data &h100,&h100,&h7c0,&h100,&
     h380,&h7c0,&hfe0,&hfe0
102 Data &hfe0,&h7c0,&h380,&h100,&
     h380,&h7c0,&h820,&h1ff0
103 Data &h80,&h9c8,&h1c9c,&h888,&
     hffb,&hffb,&hffb,&h410
104 Data &h3e0,&h1c0,&h1c0,&h1c0,&
     h3e0,&h7f0,&hc18,&h1ffc
105 Data &h0,&h36d8,&h36d8,&h3ffb,&
     h3ffb,&h3ffb,&h1010,&h1ff0
106 Data &h820,&hfe0,&hfe0,&hfe0,&
     hfe0,&hfe0,&h1010,&h3ffb
107 Data &h100,&h180,&h7e0,&hbf0,&
     hdb0,&hbf8,&hdf8,&hbf8
108 Data &hddc,&h7c8,&h3c0,&h3c0,&
     h3c0,&h3c0,&h420,&hff0
109 Data &h3e0,&h7e0,&hfe0,&h1fd8,&
     h1fb8,&hf70,&h7e0,&h3c0
110 Data &h3c0,&h5a0,&h3c0,&h3c0,&
     h3c0,&h7e0,&h810,&h1ffb
111 Data &h0,&h3c0,&h7e0,&hff0,&hf
     f0,&h7e0,&h3c0,&h180
112 Data &h7e0,&h180,&h180,&h3c0,&
     h3c0,&h7e0,&hff0,&h1ffb
113 Data &h100,&h100,&h7c0,&h100,&
     h380,&h440,&h820,&h820
114 Data &h820,&h440,&h380,&h100,&
     h380,&h7c0,&h820,&h1ff0
115 Data &h80,&h9c8,&h1c9c,&h888,&
     hffb,&h808,&hffb,&h410

```

116 Data &h360,&h140,&h140,&h140,&
 h3e0,&h410,&h808,&h1ffc
 117 Data &h0,&h36dB,&h36dB,&h3ffB,
 &h301B,&h301B,&h1ff0,&h1B30
 118 Data &hfe0,&h820,&h820,&h820,&
 h820,&h820,&h1ff0,&h3ffB
 119 Data &h100,&h180,&h6e0,&hc10,&
 ha50,&hd08,&ha0B,&hd24
 120 Data &ha54,&h54B,&h240,&h240,&
 h240,&h3c0,&h420,&hff0
 121 Data &h3e0,&h410,&h810,&h102B,
 &h104B,&h890,&h420,&h240
 122 Data &h240,&h420,&h240,&h240,&
 h240,&h7e0,&h810,&h1ffB
 123 Data &h0,&h3c0,&h420,&h810,&hB
 10,&h420,&h3c0,&h1B0
 124 Data &h7e0,&h180,&h180,&h3c0,&
 h240,&h7e0,&h810,&h1ffB
 125 Ordre:
 126 Data 3,4,5,1,2,8,7,6



A VOTRE SERVICE

DISQUETTE D'ACCOMPAGNEMENT

Certains ouvrages font l'objet d'une disquette d'accompagnement reprenant les programmes contenus dans le livre ou les applications associés. Pour les obtenir, reportez-vous à la page "disquette d'accompagnement" insérée au début de chaque ouvrage possédant une disquette.

EN COMPOSANT LE 3615 CODE OI * LIV NOTRE SERVICE MINITEL VOUS PROPOSE

- de vous renseigner sur notre catalogue et toutes nos nouveautés
- de vous indiquer le plus proche point de vente
- de répondre à vos questions techniques concernant nos ouvrages grâce à la messagerie P.S.I.

CATALOGUES ET "LIVRES MICRO"

Vous pouvez recevoir chez-vous les catalogues complets de nos ouvrages et être abonné gratuitement à la revue "Livres Micro". Pour ce faire, envoyez le coupon ci-dessous à :

Editions P.S.I. 6-10 boulevard Jourdan - 75014 PARIS



VOTRE AVIS NOUS INTERESSE

Je désire recevoir gratuitement : ☐ vos catalogues ☐ la revue "Livres Micro"

Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques et vos suggestions sur le présent ouvrage.

Titre de l'ouvrage : _____

- Ce livre vous donne t-il toute satisfaction ?

- Y a t-il un aspect du problème que vous auriez aimé voir aborder ?

Nom _____ Prénom _____ Age _____

Adresse _____

Profession _____

Centre d'intérêt : ☐ PC

☐ Macintosh

☐ Autre

● Achevé d'imprimer par l'imprimerie Tardy Quercy S.A. - Bourges
Dépôt légal : Octobre 1988 - N° d'Imprimeur : 14716
N° d'Édition : 86595-483-2



Ce livre, idéal pour le débutant, va au fil de ses 102 programmes de jeu, vous guider dans l'exploration du Basic GFA sur Atari ST.

Les programmes courts et faciles à recopier sont classés par niveau, chacun d'eux faisant appel à de nouvelles connaissances et à une plus grande maîtrise du Basic GFA.

Chaque niveau commence par une présentation des nouvelles instructions utilisées. Tous les jeux sont décrits et les programmes abondamment commentés.

**DES LIVRES
POUR VOTRE
ORDINATEUR**

PSA

ISBN : 2-86595-483-8
135 FF 5000121



Couverture GIBRAT



102 PROJETS POPULAIRES POUR ATARI ST

PAR JACQUES DECOURCEL

