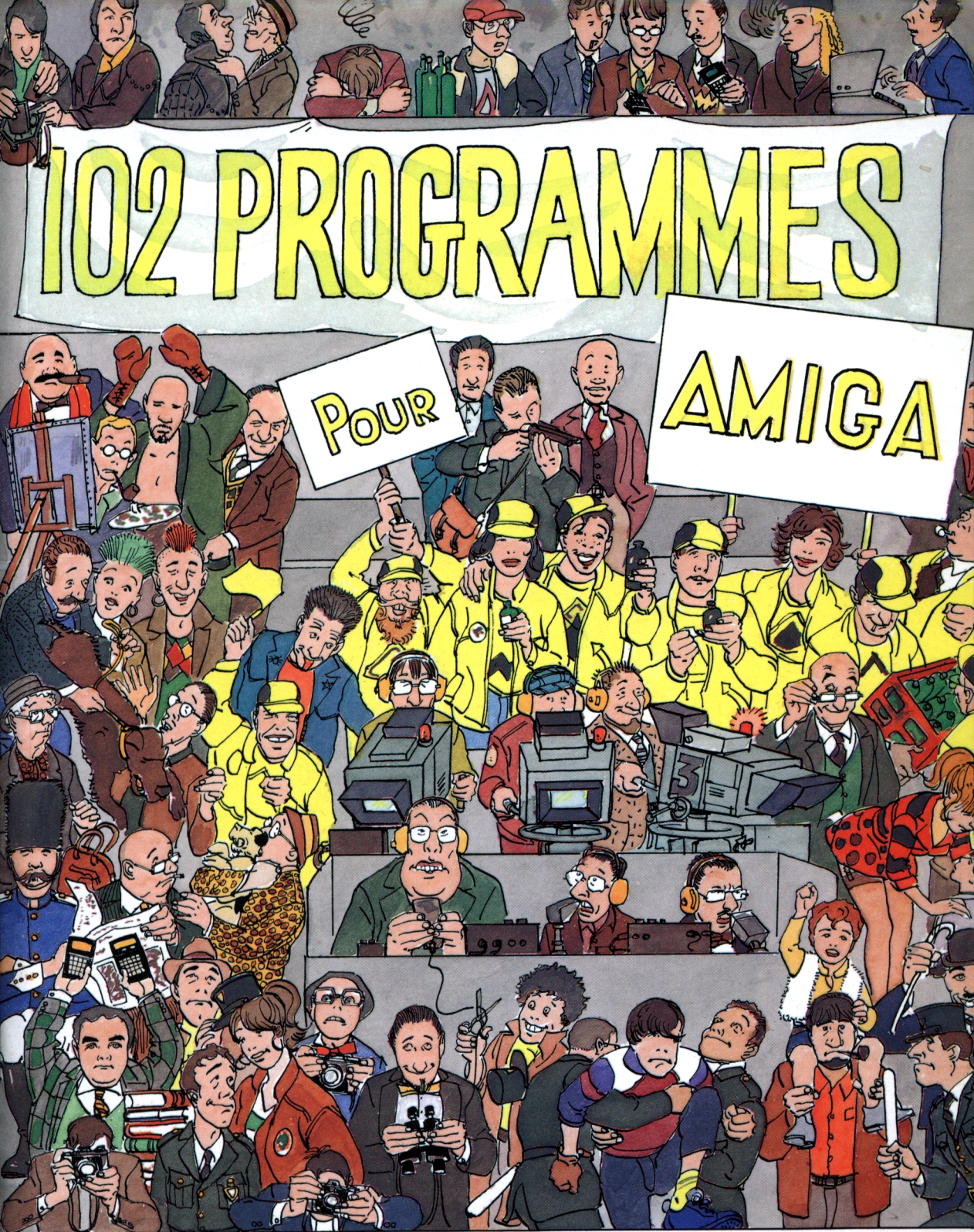




PSi.

PAR
JACQUES
DECONCHAT

PSi.

102 PROGRAMMES POUR AMIGA

Pour connaître les dernières nouveautés P.S.I.,
ou nous soumettre un problème technique,
nous mettons à votre disposition un service Minitel

Service Minitel

Sur le 3615 tapez OI puis PSI

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., une société de Groupe
5, place du Colonel Fabien - 75491 PARIS Cedex 10
1987

ISBN 2-86595-474-9

**JACQUES
DECONCHAT**

**102 PROGRAMMES POUR
AMIGA**



EDITIONS P.S.I.
1987

**Amiga, AmigaBasic et AmigaDOS sont des marques déposées de
Commodore Computer.**

SOMMAIRE

Avant-propos	8
Organisation de l'ouvrage	10

Niveau 1

Instructions élémentaires	12
Le nombre mystérieux	18
Les allumettes	20
Allumettes à plusieurs	22
Bataille navale	24
Couloirs	26
Calendrier perpétuel	28
Rebonds	30
Devine un chiffre	32
Le jeu de Witthof	34
Pair - Impair	36
Jeu de l'oie aléatoire	38
Jeu de la fève	40
Roulette	42
Golf	44
Jackpot	46
Le jeu du 21	48
Chaud, froid	50

Niveau 2

Dimensionnement d'un tableau - Recherche d'un élément dans un tableau et dans une chaîne - L'instruction DATA - Manipulation des chaînes	55
Le jeu de saute-mouton	58
La course au 20	60
L'électricien fou	62
Le jeu de la vie	64
Le jeu de Marienbad	66
Dé truqué	68
Solfège	70
Jour de la semaine	72
421	74
Musique	76
Machinamot	78
Bridge	80
Espion	82

Pendu inversé	84
Répétitions	86
Tour de cartes	88
Course de chevaux	90
Idem	92
Belote	94
Un taquin de calcul	96
Hanoï	98

Niveau 3

Graphisme et jeux	102
Compose	104
Le jeu du loto	106
Mastermind	108
Nicomaque	110
Cube	112
Von Koch	114
A travers	116
Visions	118
Mathieu	120
Le loup et les agneaux	122
Accrochez les wagons	124
Le pendu	126
Plusieurs en un	128
Chasse au sous-marin	130
Le lézard d'Alice	132
Dessin de dés	134
Chapeaux	136
Labyrinthix	138

Niveau 4

Structures de programmes - Techniques d'animation - Fenêtres et menus	142
Smokey	146
Zombies	148
Clavier	150
Calculatrice	152
La bombe	154
Adresse	156
Le jeu du missile	158
Horloge	160
Mots	162
Diago	164
Decomaque	166
Affichage publicitaire	168

Tirage d'un dé truqué	170
Le singe	172
Lettres	174
Mur de briques	176
Œufs	178
Courses de voitures	180
Obstacles	182
Échiquier	184
Cibles	186

Niveau 5

Souris, Sprites et Bobs	190
Plus grand, plus petit	192
Anagrammes	194
Permutations	196
Le jeu du 15	198
Dessin de cercles	200
Reines sur l'échiquier	202
Gouttes	204
Ardoise magique	206
Tic Tac Toc	208
Quatre à quatre	210
Taquin de lettres	212
Parcours du cavalier	214
Anomalie	216
L'indésirable	218
Ordre	220
Dames	222
Othello	224
Échecs	226
Poker	228
Fais-moi un dessin	231
Animation d'images	234
Captures dans l'espace	236
Danger spatial	238
Combat aérien	240
Sous-marin	242

Avant-propos

Vous trouverez dans ce livre plus de 100 programmes (essentiellement des jeux) rédigés en AmigaBasic pour les ordinateurs Amiga 500, 1000 et 2000.

Pour utiliser ces programmes, il suffit de charger AmigaBasic et de les taper dans la fenêtre LIST. La mise en route peut être faite soit en cliquant sur "start" dans la barre des menus, soit en activant la fenêtre d'exécution et en tapant RUN. Si la fenêtre d'exécution apparaît en grisé il sera nécessaire de l'activer en cliquant à l'intérieur avec la souris (sinon aucune entrée ne sera possible).

En AmigaBasic, il n'est pas nécessaire de numéroter les lignes de programme. Dans le livre, les numéros ont été volontairement rajoutés pour faciliter les commentaires des programmes ; **il ne faut pas les taper** (en principe, cependant, aucune erreur ne doit se produire, l'AmigaBasic ne tenant pas compte des numéros dans ce cas).

Pour des raisons de clarté les noms des variables sont donnés en majuscules dans les commentaires, alors que, pour la même raison, ils sont en minuscules dans les programmes. Seuls les mots réservés et les textes sont en majuscules.

Il sera préférable, avant de taper un programme, de prévoir une disquette déjà préparée qui pourra servir à sauvegarder le programme avant sa mise en route. En cas d'erreur grave, avec obligation de redémarrer l'ordinateur, le programme ne sera pas perdu.

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 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.

Note importante :

Dans tous les programmes, les textes apparaissent en caractère TOPAZ, taille 9. Vérifiez, avant de lancer le Basic, que vous êtes bien réglé sur ce type de caractère. Pour l'imposer à l'allumage de l'Amiga, vous pouvez utiliser le programme "Préférences", livré avec le WORKBENCH. Il faut déverrouiller la disquette WORKBENCH, cliquer deux fois sur l'icône préférence, puis cliquer sur **60** dans la rubrique "Tent". On devra ensuite cliquer sur SAVE pour enregistrer ce réglage avant de lancer le Basic (ou sur USE pour l'utiliser provisoirement, sans le sauvegarder, il faudra alors le refaire à chaque démarrage !).

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...ELSE...**, **FOR...NEXT**, **WHILE...WEND**, **GOTO**, **GOSUB**, **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 caractère : **DIM**, **R(I)**, **READ**, **DATA**, **MID\$**, **RIGHT\$**, **LEFT\$**.
- **Au niveau 3**, on s'intéresse plus particulièrement au graphisme et à une instruction permettant la saisie au vol d'un caractère frappé au clavier (**INKEY\$**). On étudiera notamment **PSET**, **LINE**, **CIRCLE**, **PAINT**, **AREA** et **AREAFIL**.
- **Le niveau 4** nous présente de nouvelles instructions pour mieux structurer les programmes, des instructions permettant l'animation d'images de façon simple, l'utilisation de fenêtres multiples et les menus déroulants.
- Enfin, **le niveau 5** est plus particulièrement consacré à l'utilisation de la souris, ainsi qu'à la gestion d'objets graphiques spécifiques (les **PRITES** et les **BOBS**).

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

Les listings des programmes sont imprimés d'après les originaux de l'auteur, sur la même imprimante.

La touche d'entrée, marquée par une flèche sur le clavier , est appelée **RETURN**.

Niveau 1

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

Instructions élémentaires

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

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

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

LET (optionnelle) ; **PRINT** ; **INPUT** ; **GOTO** ; **IF** **THEN** ;
FOR ... NEXT ; **WHILE ... WEND** ; **COSUB ... RETURN**.

soit seulement huit instructions de base (et deux instructions dérivées) que nous allons maintenant examiner plus en détail.

L'INSTRUCTION LET OU INSTRUCTION D'AFECTATION

C'est l'instruction par excellence des langages de programmation évoluée, puisque son rôle essentiel est de permettre, pour le programmeur, une gestion transparente de la mémoire. Cela signifie, en pratique, 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 AmigaBasic, comme dans la plupart des Basic, le LET est devenu optionnel, aucune ambiguïté n'étant possible, on écrira donc : A = 12 ; NOM\$ = "TOTO" ; etc.

L'INSTRUCTION PRINT OU INSTRUCTION DE SORTIE (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 individuel pour faire connaître ses résultats. Ainsi, PRINT A1 provoque l'affichage de 12 (si l'on a fait auparavant 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 1^{re} 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 *énième* colonne à partir de la gauche (de 1 à 40 ou de 1 à 80, 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 pré-définie.

Lors de la frappe, l'**instruction PRINT** pourra être abrégée en tapant ? (**point d'interrogation**). Lors d'une impression, tout nombre est précédé d'un signe ; mais le signe positif n'est pas affiché, un espace précèdera donc automatiquement l'impression de tout résultat positif.

L'INSTRUCTION LOCATE

L'AmigaBasic 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**. Il s'agit de l'instruction LOCATE X, Y. X correspond à une tabulation verticale (de 1 à 21) et Y à une tabulation horizontale (de 1 à 31 ou 62) en mode texte.

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

LOCATE 8,5 : PRINT "BONJOUR"

Le choix de la largeur d'écran peut être imposé par l'instruction WIDTH.

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 l'AmigaBasic rencontre une instruction INPUT, il s'arrête et vous rend la main ; on peut alors lui entrer une information de nature convenable (nombre ou texte), et lui signaler que cette entrée est terminée à l'aide de la touche RETURN, ce qui lui rendra la main.

Si l'information entrée n'est pas de nature convenable (vous tapez, par exemple, un texte sur un INPUT X, qui attend un nombre), le message **REDO FROM START** est affiché, et l'appareil se remet en attente.

Il est possible, comme pour le PRINT, d'afficher des messages avant l'entrée de la variable, et d'utiliser l'instruction LOCATE.

On pourra, par exemple, écrire :

INPUT "VOTRE NOM, S'IL VOUS PLAÎT" ; A\$

On pourra aussi faire appel à 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\$.

L'INSTRUCTION GOTO OU INSTRUCTION DE RENVOI INCONDITIONNEL

L'AmigaBasic, contrairement aux Basics standards, peut se passer de numéros de lignes ; les lignes étant de toute façon exécutées les unes après les autres, dans l'ordre de leur écriture, même si elles sont numérotées. Il peut être nécessaire, au cours d'un programme, de ne pas suivre cet ordre séquentiel très rigide ; on utilisera alors *l'instruction GOTO, qui permet de détourner le fonctionnement du programme de son cours normal* quitte d'ailleurs à le reprendre ensuite par un autre GOTO. Le renvoi doit, par ailleurs, impérativement se faire sur un numéro ou une étiquette existant dans le programme (GOTO début, par exemple).

Pour donner un peu de souplesse à cette instruction de renvoi, il existe une instruction très puissante, qui permet de renvoyer à n'importe quel numéro de ligne, selon la valeur d'une variable numérique : **c'est l'instruction**

tion ON__ GOTO__ ; on écrira par exemple :

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

L'INSTRUCTION IF__ THEN__ OU INSTRUCTION DE TEST

On peut l'utiliser de deux façons : d'abord, en rupture de séquence conditionnelle, sous la forme :

IF condition = vraie THEN étiquette

C'est l'usage le plus courant de ce test, qui permet de renvoyer à une autre partie de programme, si par exemple $X = 0$, ou si $A = B$, etc. Les **opérateurs logiques NOT, OR** ou **AND** peuvent d'ailleurs être utilisés dans l'écriture de la condition.

On pourra aussi utiliser le test sous la forme :

IF condition THEN instruction ;

l'instruction présentée à la suite de THEN peut être une affectation $A = 12$, une impression PRINT A, une entrée INPUT B \$, ou même un autre test IF __ THEN __.

On peut par exemple écrire :

IF $A > 3$ THEN IF $B < 7$ THEN PRINT $A + B$

L'AmigaBasic permet, en outre, de prévoir une autre instruction en cas de non-réalisation du test. C'est la forme dite IF condition THEN instruction 1 ELSE instruction 2.

L'INSTRUCTION FOR...NEXT OU INSTRUCTION DE BOUCLES

On est très souvent conduit, en programmation, à vouloir exécuter un nombre entier de fois une certaine suite d'instructions (par exemple, faire avancer un mobile de la case 6 à la case 20, etc.). Plutôt que d'utiliser un test et un renvoi, il sera préférable de faire appel à une **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 aussi 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 $I = 0$, pour $I = 2, \dots$ et ceci jusqu'à $I = 20$, soit 11 fois. La variable I peut être utilisée pour faire un calcul dans la boucle, mais ne devra pas être modifiée dans la boucle. On peut, bien sûr, sortir de la boucle, avant $I = 20$, par un test portant une quantité quelconque.

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 de noms différents, et d'omettre l'indication de la variable dans le NEXT.

UNE DEUXIEME FORME DE BOUCLE PLUS STRUCTUREE : L'INSTRUCTION WHILE...WEND

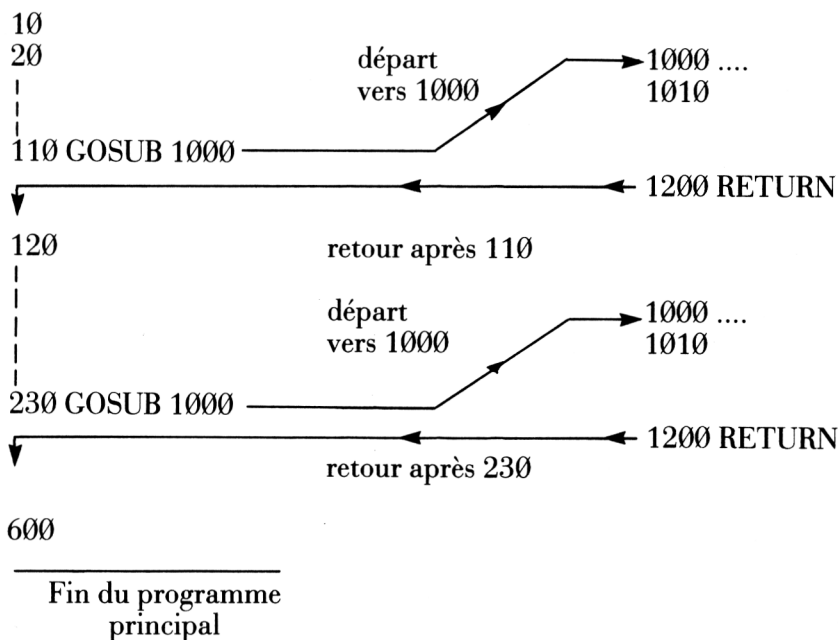
L'instruction FOR...NEXT souffre d'un excès de rigidité puisque le compteur de boucle (en général une variable entière I ou J) ne peut progresser que d'une quantité constante (de 1 en 1, ou de 2 en 2, etc.). La structure WHILE...WEND est en général plus souple : le test de fin est fait en début de boucle (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 dès le départ.

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 (en mettant, par exemple, après la dernière ligne du programme). Indépendamment des difficultés que l'on pourra rencontrer au niveau du nom des variables, lorsque l'on sera désireux d'utiliser cette séquence, il suffira de l'appeler par **l'instruction GOSUB étiquette**. Le programme ira dès lors "faire un détour" dans la partie du programme appelé, mais il ne pourra revenir et poursuivre son déroulement normal que si il rencontre dans le programme appelé une instruction **RETURN**, ou instruction de retour.

Voici le schéma théorique de fonctionnement :

Programme principal



1000
1200 RETURN

Le sous-programme n'est écrit qu'une fois, mais appelé aussi souvent que l'on veut.

Sous-programme 1

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

On peut aussi appeler un sous-programme de façon conditionnelle, en utilisant l'instruction `ON [ ] GOSUB [ ]`.

Par exemple `ON X GOSUB 1000, 2000, 3000` permettra d'exécuter le sous-programme placé en 1000 si $X = 1$, en 2000 si $X = 2$, en 3000 si $X = 3$.

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 5 permet de fabriquer le nombre à deviner.

Les entrées sont faites par les lignes 8 et 9.

La ligne 7 teste si le résultat est exact.

La ligne 10 signale un essai trop petit, et la ligne 11 un essai trop grand.

Le compteur d'essais est en ligne 12.

Les lignes 15 à 17 s'occupent de l'affichage du résultat, et de la gestion de la partie suivante.

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 RETURN.

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

On entre un nombre, et l'on presse la touche RETURN ou .

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.

En fin de partie, on obtient la question AUTRE PARTIE (O/N) ; on répond O pour faire une autre partie (suivi de RETURN).

On peut arrêter à tout moment en tapant CTRL et C.

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 LE NOMBRE MYSTERIEUX
2  CLS:RANDOMIZE TIMER
3  INPUT"NOMBRE DE CHIFFRES :";n
4  IF n>7 THEN RUN
5  x=INT(RND*10^n+1)
6  j=0
7  WHILE m<>x
8  PRINT:PRINT"ESSAI NO. ";j+1
9  INPUT"VOTRE NOMBRE :";m
10 IF m<x THEN PRINT"TROP PETIT"
11 IF m>x THEN PRINT"TROP GRAND"
12 j=j+1
13 WEND
14 CLS
15 PRINT"VOUS AVEZ TROUVE ";x;" EN ";j;" ESSAIS"
16 INPUT"AUTRE PARTIE (O/N)";r$
17 IF r$="O" OR r$="o" THEN RUN

```

Les allumettes

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : contre l'Amiga

Difficulté du jeu : moyenne.

LE JEU :

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

LE PROGRAMME :

Le jeu du joueur commence en ligne 16.

Celui de l'Amiga commence en ligne 22.

La ligne 17 contrôle la légalité du coup du joueur.

La ligne 7 gère le changement de joueur.

Le coup de l'Amiga est calculé dans une boucle WHILE...WEND (lignes 23 à 26).

Le résultat est affiché par la ligne 28.

Si la partie est terminée, la ligne 6 le détecte et envoie en ligne 12 ou 13, pour le message de fin.

POUR JOUER :

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

Après chaque coup, le nombre d'allumettes restantes est affiché, et le PC 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 de laisser le hasard choisir le nombre d'allumettes et la prise maximum, etc.

```

1  REM LES ALLUMETTES
2  COLOR 1,2:CLS
3  INPUT"NOMBRE TOTAL D'ALLUMETTES :";n
4  INPUT"PRISE MAXIMUM PAR COUP :";i
5  INPUT"QUI COMMENCE (VOUS:0 / MOI:1) :";d
6  WHILE n<>0
7  IF d=1 THEN GOSUB machine ELSE GOSUB joueur
8  n=n-p
9  PRINT "IL RESTE ";n;" ALLUMETTES"
10 WEND
11 PRINT
12 IF d=1 THEN COLOR 3:PRINT"J'AI GAGNE":BEEP
13 IF d<>1 THEN COLOR 0:PRINT"BRAVO":BEEP:BEEP:BEEP
14 END
15 joueur:
16 p=0
17 WHILE p<>INT(p) OR p<1 OR p>i OR p>n
18 COLOR 0:INPUT"VOUS RETIREZ :";p :COLOR 1
19 WEND
20 d=1:RETURN
21 machine:
22 s=i+d:t=.5
23 WHILE t<>INT(t)
24 t=(n-s)/(i+d)
25 s=s-d
26 WEND
27 p=ABS(s):IF p=0 THEN p=d
28 COLOR 3:PRINT "J'EN PRENDS ";p
29 COLOR 1:d=0:RETURN

```

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 PC 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 3 et 4 permettent les différentes entrées.

Les lignes 9 à 11 réalisent les impressions.

La ligne 12 contrôle le coup de chaque joueur (par rapport au coup précédent).

La ligne 7 gère les tours (remise à 1 du numéro du joueur).

La ligne 6 réalise le test de fin, et la ligne 18 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 RETURN, puis le nombre de joueurs (et RETURN).

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 maximum autorisé est à chaque fois affiché par l'Amiga.

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

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  COLOR 2,1:CLS:DEFINT a-z
3  INPUT"NOMBRE TOTAL D'ALLUMETTES :";n
4  INPUT"NOMBRE DE JOUEURS :";j
5  f=1:k=0
6  WHILE n<>0
7    c=0:k=k+1:IF k>j THEN k=1
8    CLS
9    PRINT"TOUR DU JOUEUR :";k
10   PRINT"TOTAL RESTANT :";n
11   PRINT"MAXIMUM A ENLEVER :";f
12   WHILE c<=0 OR c<>INT(c) OR c>f OR c>n
13     INPUT"VOTRE COUP :";c
14   WEND
15   n=n-c:f=2*c
16 WEND
17 COLOR 1,3:PRINT
18 PRINT"LE JOUEUR NUMERO ";k;" A GAGNE"
19 BEEP:BEEP:BEEP:END

```

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 l'Amiga 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 :

La boucle I (4-7) dessine le terrain de jeu.

Les lignes 10 et 11 définissent l'emplacement du navire.

Les coordonnées du tir sont entrées aux lignes 14 et 15, et la ligne 17 affiche le coup sur l'écran. La distance est calculée en ligne 16, et la ligne 12 regarde si l'on a ou non fait mouche.

La ligne 19 intervient pour déplacer le navire si les coups sont rapprochés.

La ligne 18 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 RETURN et, à la question COLONNE, on répond en donnant le numéro de la colonne verticale suivi de RETURN.

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.

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 l'Amiga, mettre plusieurs navires, des sous-marins, etc.

```

1  REM BATAILLE NAVALE
2  COLOR 2,1:CLS:RANDOMIZE TIMER:DEFINT a-z
3  PRINT"    0 1 2 3 4 5 6 7 8 9"
4  FOR i=0 TO 9
5  PRINT i;" . . . . .";
6  IF i<9 THEN PRINT:PRINT
7  NEXT i
8  s=0:d=1
9  debut:
10 x=INT(RND*10)
11 y=INT(RND*10)
12 WHILE d<>0
13 LOCATE 8,49:PRINT "    ":LOCATE 10,49:PRINT "    "
14 COLOR 3:LOCATE 8,40:INPUT"LIGNE    ":";h
15 LOCATE 10,40:INPUT"COLONNE ":";v
16 d=INT(SQR((x-h)*(x-h)+(y-v)*(y-v)))
17 s=s+1:LOCATE 2*h+2,3*v+4:COLOR 0:PRINT s;:COLOR 2
18 LOCATE 12,40:PRINT"DISTANCE    ":";d;"    ":BEEP
19 IF d=1 THEN LOCATE 12,40:PRINT"PFFF...
    ":GOTO debut
20 WEND
21 COLOR 3:PRINT"TROUVE EN ":"s;" COUPS":END

```

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 3 à 7 dessinent les couloirs (et les trois tourniquets).

La ligne 10 attend votre choix.

Le décodage du chemin est fait par l'Amiga en utilisant trois variables logiques I, J et K, représentant les états des trois tourniquets (lignes 12 à 28).

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 de l'Amiga.

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. Ceci présente l'avantage de permettre à plusieurs joueurs d'exercer leur sagacité en essayant de prévoir, pour une entrée donnée, ce que va répondre l'Amiga.

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  DEFINT a-z:CLS
3  PRINT"-----"
4  PRINT:PRINT"B          ----- X 2"
5  PRINT:PRINT"----- X -----"
6  PRINT:PRINT"A          ----- X 1"
7  PRINT:PRINT"-----"
8  i=0:j=0:k=0:t$=""
9  entree:
10 COLOR 1:LOCATE 13,1:INPUT"ENTREE 1 OU 2 :";c$
11 IF c$<>"1" AND c$<>"2" THEN GOTO entree
12 IF c$="1" THEN GOTO labi
13 labj:
14 j=NOT(j)
15 IF j<>0 THEN GOTO labk
16 s$="B":GOTO sortie
17 labi:
18 i=NOT(i)
19 IF i<>0 THEN GOTO labk
20 s$="A":GOTO sortie
21 labk:
22 k=NOT(k)
23 IF k<>0 THEN s$="B":GOTO sortie
24 s$="A"
25 sortie:
26 t$=t$+s$:COLOR 2:LOCATE 15,1:PRINT"SORTIE EN ";s$
27 COLOR 3:PRINT:PRINT t$
28 GOTO entree

```

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 3 et 4 vous permettront d'entrer les mois et l'année.

Les lignes 5, 6 et 7 calculent le numéro du premier jour du mois.

La ligne 8 calcule le nombre de jours du mois (il n'est pas tenu compte ici des 29 février, pour simplifier).

La boucle A (12-15) 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 RETURN.

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 FÉVRIER, etc.).

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

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  COLOR 3,1:CLS:DEFLNG a-z
3  INPUT"POUR QUELLE ANNEE :";a
4  INPUT"POUR QUEL MOIS   :";m
5  d=INT(365.25*(a+(m<3)))+1
6  d=d+INT(30.6*(m+1-(m<3)*12))-INT((INT((a+(m<3))/100
   )-7)*.75)
7  d=d-7*INT(d/7)
8  m=31+(m=4 OR m=6 OR m=9 OR m=11)+(m=2)*3
9  PRINT:COLOR 2
10 PRINT" DIMANCHE LUNDI      MARDI      MERCREDI JEUDI
    VENDREDI SAMEDI  "
11 COLOR 0:PRINT
12 FOR a=1 TO m
13 PRINT TAB(9*(d+a-1)+2);a;
14 IF d+a-1=6 THEN PRINT:d=d-7
15 NEXT a

```

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 5 imprime les chiffres et les lettres en haut de l'écran.

Les lignes 6 et 7 définissent les positions initiales de la balle et de la cible.

Les lignes 8 et 9 effectuent leur impression.

Le symbole correspondant à la direction choisie est entré en ligne 10.

La boucle J (12-15) réalise le déplacement de la balle.

La ligne 14 imprime la nouvelle position.

Le changement de direction, lors de la rencontre d'un bord, est effectué par la ligne 14.

La ligne 16 effectue l'impression de fin de partie.

Le nombre de rebonds est compté en ligne 14.

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 CTRL et Break.

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  RANDOMIZE TIMER
3  WHILE r$<>"N" AND r$<>"n"
4  COLOR 1,0:CLS
5  PRINT" 0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
6  x=INT(RND*18+3)
7  y=INT(RND*18+3)
8  COLOR 2:LOCATE x,1:PRINT"#"
9  COLOR 3:LOCATE y,40:PRINT"+"
10 COLOR 1:LOCATE 19,1:INPUT"VOTRE COUP :";r$:r$=UCASE
    $(r$)
11 s=0:t=x:COLOR 2
12 FOR j=1 TO 39
13 t=t-x/(ASC(r$)-ASC("0")+7*(r$>="A"))
14 IF t>=20 OR t<=0 THEN x=-x:s=s+1:BEEP:ELSE LOCATE t
    +1,j+1:PRINT"#";
15 NEXT j
16 COLOR 1:IF ABS(t-y+1)<3 THEN PRINT"BRAVO :";s;" REB
    ONDS"
17 INPUT"AUTRE PARTIE (O/N):";r$
18 WEND
19 END

```

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 26 et 28.

L'affichage est réalisé par les lignes 14 et 15.

Les lignes 17 à 19 calculent le gain dans les diverses configurations possibles.

La totalisation est effectuée en ligne 21 pour les gains ou pour les pertes.

La variable r\$, testée en ligne 25, 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 RETURN. On obtient alors un deuxième affichage DONNEZ UN CHIFFRE DE 1 A 6 ; on entre un chiffre suivi de RETURN ; 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 (suivi de RETURN).

EXTENSIONS POSSIBLES :

On pourra facilement introduire des contrôles (mise minimum, 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, faire même participer l'Amiga.

```

1  REM DEVINE UN CHIFFRE
2  COLOR 1,2:RANDOMIZE TIMER:DEFINT a-z:g=0
3  debut:
4  CLS
5  INPUT"VOTRE MISE :";n
6  PRINT
7  INPUT"DONNEZ UN CHIFFRE DE 1 A 6 :";d
8  GOSUB tirage
9  a=x
10 GOSUB tirage
11 b=x
12 GOSUB tirage
13 c=x
14 PRINT:PRINT:PRINT"      A              B              C"
15 PRINT:PRINT"      ";a;"              ";b;"              ";c
16 t=0
17 IF d=a THEN t=t+n
18 IF d=b THEN t=t+n
19 IF d=c THEN t=t+n
20 PRINT:PRINT
21 IF t=0 THEN PRINT "PERDU":g=g-n ELSE PRINT "VOUS GA
    GNEZ ";t:g=g+t
22 PRINT:PRINT:PRINT"TOTAL ACTUEL :";g
23 PRINT
24 INPUT"ON CONTINUE (O/N) :";r$
25 IF r$="N" OR r$="n" THEN END ELSE GOTO debut
26 tirage:
27 x=INT(RND*6+1):SOUND 100*(x+5),5
28 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 4, 5 et 6 permettent de définir une position de départ, en dehors de la diagonale.

La ligne 8 et la boucle I (9-12) dessinent le terrain de jeu.

La ligne 15 imprime la position du point mobile.

Le test de fin est effectué en ligne 14.

Le changement de joueur est assuré par la ligne 19.

Les directions de déplacement disponibles, 1, 2 et 3, sont décodées par trois sous-programmes qui commencent aux lignes 22, 25 et 28.

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, suivi de RETURN.

C'est ensuite au tour du joueur 2.

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

EXTENSIONS POSSIBLES :

On peut tout d'abord améliorer les contrôles (la sortie du terrain n'est pas interdite, par exemple), 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 à l'Amiga de jouer, etc.

```

1  REM LE JEU DE WYTTTHOF
2  RANDOMIZE TIMER:DEFINT a-z
3  tirage:
4  x=INT(RND*5)
5  y=INT(RND*5)
6  IF y=x THEN GOTO tirage
7  CLS
8  PRINT"      0  1  2  3  4  5  6  7  8  9"
9  FOR i=0 TO 9
10 PRINT i;"  .  .  .  .  .  .  .  .  .  .";
11 IF i<9 THEN PRINT:PRINT
12 NEXT i
13 h=INT(RND*2+1)
14 WHILE x<>9 OR y<>9
15 COLOR h+1:LOCATE 2*y+2,3*x+6:PRINT"+";:BEEP
16 COLOR h+1:LOCATE 8,40:PRINT"JOUEUR ";h;:INPUT" - DI
   RECTION (1,2,3)";d$
17 d=VAL(d$)
18 ON d GOSUB droite,bas,diago
19 h=2-ABS(h-1)
20 WEND
21 PRINT "C'EST FINI....":END
22 droite:
23 x=x+1:IF x>9 THEN x=9
24 RETURN
25 diago:
26 GOSUB droite:GOSUB bas
27 RETURN
28 bas:
29 y=y+1:IF y>9 THEN y=9
30 RETURN

```

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. L'Amiga 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 31 et 32.

La ligne 8 tient le compte du nombre de parties, et les lignes 14, 26 et 28 comptabilisent les points marqués.

La parité du total est déterminée par les lignes 13 et 14 (la ligne 14 compare avec la réponse du joueur).

La ligne 20 réalise l'affichage de fin de partie : nombre de points, nombre de parties jouées.

La ligne 22 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 RETURN 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 suivi de RETURN 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  COLOR 1,2:DEFINT a-z
3  RANDOMIZE TIMER:t=0:r=0
4  debut:
5  GOSUB tirage
6  a=x
7  GOSUB tirage
8  r=r+1
9  s=a+x
10 CLS
11 INPUT"TOTAL PAIR (0) OU IMPAIR (1) :";n
12 PRINT:BEEP
13 p=s MOD 2
14 IF n=p OR n-.5=p THEN t=t+1:PRINT"EXACT" ELSE PRINT
   "FAUX"
15 PRINT
16 INPUT"TOTAL PRESUME :";n
17 IF n=s THEN BEEP:GOSUB bravo ELSE PRINT:PRINT "PERD
   U"
18 FOR i=1 TO 3000:NEXT i
19 CLS
20 PRINT"VOUS AVEZ UN TOTAL DE ";t;" SUR ";r
21 PRINT
22 INPUT"ON CONTINUE (O/N) :";r$
23 IF r$="O" OR r$="o" THEN GOTO debut ELSE END
24 bravo:
25 PRINT:BEEP
26 t=t+2
27 INPUT"BRAVO . DONNEZ UN DE :";n:PRINT
28 IF n=a OR n=s-a THEN t=t+3:PRINT:PRINT "TRES BIEN J
   QUE" ELSE PRINT "DOMMAGE"
29 BEEP:RETURN
30 tirage:
31 x=INT(RND*6+1)
32 RETURN

```

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 l'Amiga ; il suffira d'appuyer sur une touche après chaque lancer de dé, et de regarder ce qui se passe.

LE PROGRAMME :

La ligne 5 permet d'appeler le sous-programme 28 à 39 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 6 s'occupe du changement de joueur, et la ligne 7 effectue le lancer de dé.

La ligne 8 traite le cas où l'un des joueurs termine.

La ligne 12 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 29 et 30 font avancer les pions, et les lignes 32 à 37 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 aussi s'arrêter à tout moment en tapant CTRL et Break.

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  RANDOMIZE TIMER:COLOR 1,0:CLS:DEFINT a-z
3  x=0:y=0:j=1:i=x
4  debut:
5  GOSUB dessin
6  j=ABS(j-1)
7  i=INT(RND*6+1)
8  IF x=35 OR y=35 THEN LOCATE 20,1:PRINT"BRAVO":END
9  BEEP:LOCATE 13,1:PRINT"JOUEUR ";j+1;" - VOTRE COUP
   ";i
10 INPUT"APPUYER SUR - ENTREE -";r$
11 GOSUB dessin
12 i=INT(RND*4)
13 f=INT(RND*5)
14 ON f GOSUB plus,opp,double,hasar
15 GOTO debut
16 plus:
17 i=i+3
18 RETURN
19 opp:
20 i=-i
21 RETURN
22 double:
23 i=2*i
24 RETURN
25 hasar:
26 IF j=0 THEN x=INT(RND*30):RETURN
27 y=INT(RND*30):RETURN
28 dessin:
29 IF j=0 THEN x=x-i*(x+i<=35)
30 y=y-j*i*(y+i<=35)
31 LOCATE 5,1:PRINT"0123456789ABCDEFGHIJKLMNPOQRSTUVWXYZ"
   YZ"
32 FOR i=0 TO 35
33 LOCATE 6,i+1:COLOR 3,1:PRINT" ";
34 IF i=x THEN LOCATE 6,i+1:PRINT"1";
35 LOCATE 7,i+1:COLOR 2,1:PRINT" ";
36 IF i=y THEN LOCATE 7,i+1:PRINT"2";
37 NEXT i
38 FOR i=1 TO 800:NEXT i
39 COLOR 1,0:RETURN
```

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 4 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 :

La ligne 3 et la boucle I (4-7) dessinent le terrain de jeu.

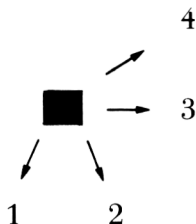
La ligne 9 réalise le test de fin de partie.

La ligne 14 permet d'exécuter l'un des sous-programmes de déplacement.

Les lignes 15 et 16 assurent 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 4 directions proposées correspondent aux déplacements suivants :



Le joueur 1 entre son déplacement, et tape RETURN.

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.

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  COLOR 0,1:CLS:DEFINT a-z
3  PRINT"      0  1  2  3  4  5  6  7  8  9"
4  FOR i=0 TO 9
5  PRINT i;" . . . . .";
6  IF i<9 THEN PRINT:PRINT
7  NEXT i
8  x=0:y=0:j=0
9  WHILE x+y<>18
10 COLOR j+2:LOCATE 2*y+2,3*x+6:PRINT"#";
11 COLOR 0:LOCATE 8,40:PRINT"JOUEUR ";:COLOR j+2:PRINT
   j+1
12 LOCATE 10,40:COLOR 0:INPUT"DIRECTION 1,2,3,4:";i$
13 i=VAL(i$):BEEP
14 id=0:a=x:b=y:ON i GOSUB bas,diagbas,droit,diago
15 IF id=1 THEN j=ABS(j-1):x=a:y=b
16 j=j+1+2*(j=1)
17 WEND
18 COLOR j+2:PRINT"LE JOUEUR ";j+1;" A GAGNE
   "
19 END
20 bas:
21 y=y+1:IF y>9 THEN id=1
22 RETURN
23 diagbas:
24 GOSUB bas
25 droit:
26 x=x+1:IF x>9 THEN id=1
27 RETURN
28 diago:
29 GOSUB droit
30 y=y-1:IF y<0 THEN id=1
31 RETURN

```

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 un carré noir, 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 :

La boucle K (4-10) dessine le terrain de jeu, auquel la ligne 11 rajoute les trois symboles P, 00 et I.

La boucle K (13-20) réalise le déplacement de la boule, l'arrêt étant contrôlé par la ligne 12.

La boucle J (ligne 17) réalise une légère temporisation.

La ligne 21 tire la position d'arrêt.

La ligne 24 gère les essais suivants.

POUR JOUER :

Il suffit de faire RUN, et de regarder.

Une fois la boule arrêtée, il suffira de frapper une touche quelconque (sauf "N") pour la relancer. Taper N 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  PALETTE 2,.1,.9,.8:COLOR 2,3
3  RANDOMIZE TIMER:DEFINT a-z:CLS
4  FOR k=1 TO 4
5  FOR i=1 TO 9
6  s=i+i
7  PRINT TAB(4*i-1)(s AND k=1)+(s+18 AND k=2)+(s-1 AND
   k=3)+(s+17 AND k=4);
8  NEXT i
9  PRINT:PRINT:PRINT:PRINT
10 NEXT k
11 LOCATE 3,1:PRINT"P":LOCATE 7,1:PRINT"00":LOCATE 11,
   1:PRINT"I"
12 WHILE x<>1
13 FOR k=1 TO 4
14 FOR i=1 TO 9-(k<4)
15 IF i=10 AND k<4 THEN f=4*k:g=1 ELSE f=4*k-2:g=4*i
16 x=x-1:COLOR 0:LOCATE f,g:PRINT "##"
17 FOR J=1 TO 200-400*(x>0)-(200*(10-x))*(x>0 AND x<8)
   :NEXT J:SOUND 800,1
18 LOCATE f,g:IF x<>1 THEN PRINT " " ELSE PRINT " *":G
   OTO fin
19 NEXT i
20 NEXT k
21 x=INT(RND*38)
22 WEND
23 fin:
24 LOCATE 18,1:INPUT"AUTRE ESSAI (O/N) :";r$
25 IF r$="N" OR r$="n" THEN END ELSE RUN

```

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 6 permet de définir la position du trou, et la ligne 8 dessine le terrain de jeu.

Le déplacement à réaliser est calculé en ligne 13.

La boucle I (15-20) réalise le déplacement de la balle.

La ligne 7 regarde si le coup est, ou non, gagnant ; s'il l'est, on recommence (ligne 26) jusqu'à ce que l'on ait réalisé 18 trous (boucle S (4-26) ; fin de boucle en 28, 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 RETURN, 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.

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  COLOR 3,1:CLS:PALETTE 2,.2,.8,.5
3  RANDOMIZE TIMER:DEFINT a-z:e=0
4  FOR s=1 TO 18
5  CLS
6  x=1:t=INT(52*RND+7)
7  WHILE x<>t
8  COLOR 2:LOCATE 6,1:FOR i=1 TO 62:PRINT CHR$(127);:N
   EXT i
9  COLOR 3:LOCATE 6,t:PRINT" ":LOCATE 5,x:PRINT"O"
10 WHILE f=0
11 LOCATE 8,15:PRINT"          "
12 LOCATE 8,1:PRINT"TROU ";s;:INPUT"FORCE ";r
13 f=INT(r*.16-(ABS(r)*(SQR(RND)+.1)^2)/1000)
14 WEND
15 FOR i=SGN(f) TO f STEP SGN(f)
16 x=x+SGN(f):SOUND ABS(i)*80,1
17 IF x<63 AND x>=1 THEN LOCATE 5,x-SGN(f):PRINT" ":LO
   CATE 5,x:PRINT"O"
18 IF x>=62 THEN x=62
19 IF x<=1 THEN x=1
20 NEXT i
21 e=e+1:f=0
22 WEND
23 e=e-1
24 LOCATE 5,x:PRINT" ":LOCATE 6,x:PRINT"O"
25 FOR p=1 TO 500:NEXT p
26 NEXT s
27 CLS
28 PRINT"18 TROUS EN ";e;" COUPS"

```

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 (6-8) permet le dessin de la machine, la ligne 9 affichant les emplacements (vides) des chiffres.

La mise sera entrée en ligne 10, et testée en ligne 11 : si on mise 0, la partie est terminée (départ vers la ligne 29).

Les lignes 12, 14 et 16 permettent le tirage des 3 chiffres, en appelant un sous-programme qui se trouve en ligne 26.

Le résultat est affiché par la ligne 17, puis testé en ligne 18, et en ligne 19, 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 RETURN, et on entre une première mise (différente de 0), suivie de RETURN.

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 RETURN 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.

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  RANDOMIZE TIMER:DEFINT a-z
3  INPUT"SOMME DISPONIBLE :";p
4  debut:
5  COLOR 1,0:CLS
6  FOR i=1 TO 3
7  LOCATE i+4,10:COLOR 3,1:PRINT "
  "
8  NEXT i
9  COLOR 3,2:LOCATE 6,14:PRINT"   ":LOCATE 6,19:PRINT"
  ":LOCATE 6,24:PRINT"   "
10 LOCATE 11,1:INPUT"VOTRE MISE :";m
11 IF m=0 OR p-m<0 THEN GOTO fin
12 GOSUB tirage
13 a=x
14 GOSUB tirage
15 b=x
16 GOSUB tirage
17 LOCATE 6,15:PRINT a:LOCATE 6,20:PRINT b:LOCATE 6,25
  :PRINT x
18 n=-m:IF a=b OR a=x OR b=x THEN n=-n
19 IF a=b AND a=x THEN n=m*50-450*(a=7):SOUND 500,4
20 COLOR 0:LOCATE 11,1:PRINT"MISE :";
21 COLOR 3:PRINT m;:COLOR 0:PRINT" - RESULTAT :";
22 COLOR 3:PRINT n;"   "
23 COLOR 0:PRINT"APPUYEZ SUR UNE TOUCHE":r$=INPUT$(1)
24 p=p+n
25 GOTO debut
26 tirage:
27 x=INT(RND*9+1):SOUND 100*(x+5),2
28 RETURN
29 fin:
30 PRINT:PRINT"IL VOUS RESTE ";p
31 PRINT"A BIENTOT":END

```

Le jeu du 21

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : jeu contre l'Amiga.

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 l'Amiga 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 :

La ligne 6 réalise le contrôle de celui qui doit jouer (joueur ou Amiga).

Le sous-programme (27-29) lance les dés.

Les lignes 7 et 8 calculent le total fait par le joueur et l'affichent.

Le total de l'Amiga est calculé en ligne 14 et affiché par la ligne 15.

La ligne 13 contrôle la fin du jeu pour l'Amiga.

La ligne 20 affiche les résultats, et la ligne 21 cherche quel est le gagnant.

Le gain (ou la perte) sera affiché par la ligne 23, ainsi que le total.

La ligne 24 contrôle la fin de partie.

POUR JOUER :

Taper RUN pour lancer le jeu.

A la question VOTRE MISE, entrer la mise initiale, suivie de RETURN.

Le premier tirage est affiché sous la forme TOTAL ACTUEL... et on devra faire 0 tant que l'on désire recommencer à tirer.

Dès que l'on a terminé, on tape N pour laisser jouer l'Amiga (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  RANDOMIZE TIMER:DEFINT a-z:g=0
3  WHILE r$<>"N"
4  COLOR 1,2:CLS:t=0
5  INPUT"VOTRE MISE :";m
6  WHILE r$<>"N"
7  GOSUB TIRAGE:t=t+x:BEEP
8  COLOR 0:PRINT"TIRAGE ";x;" - TOTAL ACTUEL ";t
9  COLOR 1:PRINT"AUTRE TIRAGE (O/N)":r$=UCASE$(INPUT$(
10 1))
10 WEND
11 l1=CSRLIN:s=0:k=1
12 LOCATE k,32:PRINT "LA BANQUE JOUE"
13 WHILE s<t AND s<17
14 GOSUB TIRAGE:s=s+x:k=k+2
15 COLOR 3:LOCATE k,32:PRINT"LA BANQUE TIRE ";x;" - TO
TAL ";s
16 FOR i=1 TO 500:NEXT i
17 WEND
18 l2=CSRLIN:l=-12*(l2>l1)-11*(l1>=l2)
19 LOCATE l,1:BEEP:BEEP:BEEP
20 COLOR 1:PRINT"VOUS AVEZ ";t;TAB(32);"LA BANQUE A ";
s
21 IF s<21 AND (s)=t OR t>21) THEN m=-m
22 g=g+m
23 PRINT "GAIN ";m;TAB(32) "VOTRE TOTAL ACTUEL ";g
24 PRINT"ON CONTINUE (O/N)":r$=UCASE$(INPUT$(1))
25 WEND
26 PRINT "AU REVOIR....":SOUND 700,10:END
27 TIRAGE:
28 x=INT(RND*6+1)
29 RETURN

```

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 :

La ligne 5 définit la position de l'objet.

La ligne 10 calcule la distance par rapport aux deux coordonnées que vous avez entrées aux lignes 7 et 8.

La ligne 11 regarde si vous avez gagné.

Les lignes 13-14 normalisent la distance (qui doit avoir un sens, quelles que soient les dimensions de la grille initiale).

La ligne 15 effectue le départ, s'il y a lieu, vers l'un des sous-programmes d'affichage.

POUR JOUER :

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

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

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

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 COMPLETÈMENT GELE à BRULANT). Lorsque l'on a trouvé, on obtient l'affichage du nombre d'essais effectués.

Pour sortir du programme avant la fin de la partie, il suffit de taper CTRL et Break.

EXTENSIONS POSSIBLES :

Si l'on dispose 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  SCREEN 1,320,200,4,1:WINDOW 1,"CHAUD - FROID",,2,1
3  COLOR 2,1:CLS:RANDOMIZE TIMER:DEFINT a-z
4  INPUT "DIMENSION DE LA GRILLE :";d
5  x=INT(RND*d+1):y=INT(RND*d+1)
6  debut:
7  COLOR 2:INPUT"DONNEZ X :";a
8  INPUT"DONNEZ Y :";b
9  s=s+1
10 t=ABS(x-a)+ABS(y-b)
11 IF t=0 THEN GOTO fin
12 COLOR 2:PRINT"VOUS ETES ";
13 n=INT(SQR(d)/t)+1
14 IF n>5 THEN n=5
15 ON n GOSUB gele,froid,tiede,chaud,treschaud,brulant
16 GOTO debut
17 gele:
18 COLOR 4:PRINT"COMPLETEMENT GELE":RETURN
19 froid:
20 COLOR 14:PRINT"A PEINE TIEDE":RETURN
21 tiede:
22 COLOR 13:PRINT"TIEDE":RETURN
23 chaud:
24 COLOR 11:PRINT"CHAUD":RETURN
25 treschaud:
26 COLOR 9:PRINT"TRES CHAUD":RETURN
27 brulant:
28 COLOR 3:PRINT"BRULANT":RETURN
29 fin:
30 COLOR 2:PRINT"VOUS AVEZ TROUVE EN ";s;" COUPS":BEEP
   :BEEP:BEEP

```


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.
L'électricien fou	Logique et difficile, pour un seul joueur.
Le jeu de la vie	Un grand classique, en deux versions : manuelle et automatique.
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.
Solfège	Apprenez le solfège, à l'aide de votre ordinateur.
Jour de la semaine	Profitez-en pour faire le point sur les anniversaires.
421	Une simulation simplifiée, plusieurs joueurs.
Musique	Réviser vos notions de musique.
Machinamot	Laissez l'ordinateur créer des vocables nouveaux.
Bridge	Mettez au point vos propres problèmes de bridge.
Espion	Dites-lui votre amour en code.
Pendu inversé	L'ordinateur essaie (honnêtement) de deviner votre mot.
Répétitions	Retrouver une séquence de chiffres de plus en plus longue.
Tour de cartes	Votre ordinateur a plus d'un tour dans son sac.
Course de chevaux	Un jeu de hasard, pour plusieurs joueurs.
Idem	Un jeu de calcul mental, avec plusieurs niveaux, pour deux joueurs.
Belote	Pour simuler une donne, avec un jeu de 32 cartes.
Un taquin de calcul	Un bon entraînement, pour deux joueurs.
Hanoï	Un bon conseil, commencez petit.

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 trente 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 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 en AmigaBasic que si vous 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 2 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 AmigaBasic il n'est pas obligatoire de dimensionner un tableau qui contient 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, calcul, 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\$, etc.

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".

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 inscrites les unes à la suite des autres, séparées par une virgule.

Ainsi, on pourra écrire :

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

Ces données peuvent être lues dans des boucles FOR... NEXT, au moyen de l'instruction READ. On peut recommencer la lecture au début des DATA, à l'aide de l'instruction RESTORE.

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

Pour faciliter les manipulations, il est possible d'utiliser les deux fonctions LEFT\$ et RIGHT\$.

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

Ainsi, LEFT\$ (A\$, 2) = "MO", avec l'exemple précédent (on pourrait d'ailleurs utiliser MID\$ (A\$, 1, 2) pour obtenir le même résultat).

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

Ainsi RIGHT\$ (A\$, 3) = "EUR", toujours avec le même exemple.

Il est plus difficile de se passer de la fonction RIGHT\$, mais il serait possible de faire MID\$ (A\$, LEN (A\$) - 3, 3) pour obtenir le même résultat.

L'AmigaBasic permet de plus le remplacement direct d'une sous chaîne par une affectation du type MID\$ (-.) = "-".

Ainsi, on pourra écrire MID\$ (A\$, I, 1) = "-", ce qui permet de simplifier considérablement certaines écritures.

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 bleus et de 3 moutons rouges, représentés par des symboles graphiques, qui occupent 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 (4-7) initialise une table A (I), qui contient les positions des moutons. La boucle I (9-21) dessine le terrain de jeu et les moutons.

Les lignes 13 à 17 s'occupent du texte et des entrées.

La ligne 16 contrôle la validité du coup. Le changement de joueur est réalisé en ligne 20.

Le sous-programme "dessine", en ligne 26, est appelé par la ligne 10, pour effectuer le dessin.

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 séparés par une virgule, suivi de RETURN.

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.

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  COLOR 0,1:CLS:DEFINT a-z
3  DIM a(8):a(1)=0:a(8)=0
4  FOR i=1 TO 3
5    a(i+1)=1
6    a(i+4)=-1
7  NEXT i
8  h=1
9  WHILE a(1)+a(2)+a(3)<>-3 AND a(6)+a(7)+a(8)<>3
10 GOSUB dessine
11 j=1-(h=-1):SOUND 300*j,4
12 entree:
13 LOCATE 13,1:PRINT SPACE$(30):LOCATE 15,1:PRINT SPAC
   E$(30)
14 LOCATE 13,1:PRINT"JOUER ";:COLOR j+1:PRINT j
15 COLOR 0:PRINT:INPUT"DEBUT , FIN :";d,f
16 IF d<>INT(d) OR f<>INT(f) OR d<1 OR d>8 OR f<1 OR f
   >8 THEN entree
17 IF a(d)<>h OR a(f)<>0 THEN entree
18 a(d)=0
19 a(f)=h
20 h=-h
21 WEND
22 GOSUB dessine
23 LOCATE 17,1:PRINT "LE JOUEUR ";j;" GAGNE"
24 FOR i=600 TO 300 STEP -20:SOUND i,4:NEXT i
25 END
26 dessine:
27 FOR i=1 TO 8
28 LOCATE 8,4*i:PRINT i
29 LOCATE 6,4*i+1:PRINT " "
30 IF a(i)=1 THEN LOCATE 6,4*i+1:COLOR 2:PRINT"#":COL
   OR 0
31 IF a(i)=-1 THEN LOCATE 6,4*i+1:COLOR 3:PRINT"%":COL
   OR 0
32 NEXT i
33 RETURN

```

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 ceci dans un seul tas à la fois. Celui qui prend la dernière étoile est considéré comme perdant.

LE PROGRAMME :

La boucle I (3 à 5) dessine 4 rangées de 20 étoiles.

La variable F, dimensionnée en ligne 6, servira à enregistrer le nombre d'étoiles prises dans chaque rangée.

La variable T contiendra le nombre total d'étoiles enlevées (ligne 6).

Les lignes 12 à 15 vont contrôler qu'il reste suffisamment d'étoiles sur la ligne indiquée.

La boucle J (17 à 20) supprime les étoiles ; la ligne 7 teste la fin de partie.

POUR JOUER :

Lancer le programme par RUN ; le terrain de jeu s'affiche, puis les inscriptions JOUEUR 1 et LIGNE, puis CASES. On répond en entrant un numéro de ligne (entre 1 et 4) suivi de RETURN, et le nombre d'étoiles à enlever suivi de RETURN. 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.

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  CLS:DEFINT a-z
3  FOR i=1 TO 4
4  COLOR 1:LOCATE 2*i+1,1:PRINT i;" *****
   ***"
5  NEXT i
6  DIM f(4):b=1:t=0
7  WHILE t<80
8  COLOR b+1:LOCATE 13,1:PRINT"JOUER " ;b
9  choix:
10 LOCATE 15,8:PRINT SPACE$(30):LOCATE 17,8:PRINT SPAC
   E$(30)
11 COLOR 1:LOCATE 15,1:INPUT"LIGNE :";l
12 IF l<>INT(l) OR l<1 OR l>4 THEN GOTO choix
13 LOCATE 17,1:INPUT"CASES :";c
14 IF c+f(1)>20 OR c<0 OR c<>INT(c) THEN GOTO choix
15 f(1)=f(1)+c
16 t=t+c
17 FOR j=f(1)-c TO f(1)
18 LOCATE 2*i+1,26-j:PRINT" ":SOUND 800,1
19 FOR i=1 TO 200:NEXT i
20 NEXT j
21 b=b+1
22 IF b=3 THEN b=1
23 WEND
24 COLOR 4*b-3:PRINT"LE JOUEUR " ;b;" GAGNE"

```

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 4, contiendra l'état (allumé ou éteint) des 10 fenêtres. La boucle I (5-7) et l'autre boucle I (10-15) dessinent l'immeuble dans son état initial.

La ligne 16 regarde si tout est éteint, auquel cas l'on ira en ligne 28 afficher le nombre d'essais.

La ligne 17 attend l'indication de l'interrupteur à manœuvrer.

Les lignes 18 à 26 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 RETURN.

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.

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  SCREEN 1,320,200,4,1:RANDOMIZE TIMER
3  WINDOW 1,"L'ELECTRICIEN FOU",,2,1
4  DIM a(10):m=0:CLS
5  FOR i=1 TO 7
6  COLOR 0,1:LOCATE i+1,1:PRINT"
   "
7  NEXT i
8  debut:
9  t=0
10 FOR i=1 TO 10
11 t=t+a(i)
12 LOCATE 4,2*i+6:COLOR 0,2:PRINT" "
13 IF a(i)=0 THEN COLOR 0,11:LOCATE 4,2*i+6:PRINT" "
14 COLOR 3,6:LOCATE 7,2*i+5:PRINT i-1
15 NEXT i
16 WHILE t<>10
17 COLOR 8,7:LOCATE 15,1:BEEP:INPUT"INTERUPTEUR NUMER
   0 :";n
18 n=n+1
19 m=m+1
20 a(n)=1-a(n)
21 calcul:
22 r=TAN(RND+n/RND-n)-SIN(RND/n)+336*SIN(8*n)
23 n=INT(10*(r-INT(r)))
24 IF n=0 THEN n=INT(RND*10+1)
25 IF a(n)=1 THEN a(n)=0:GOTO calcul
26 a(n)=1:GOTO debut
27 WEND
28 COLOR 4,9:PRINT:PRINT"TROUVE EN ";m;" ESSAIS"
29 FOR i=200 TO 800 STEP 20:SOUND i,2:NEXT i

```

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 l'Amiga (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 4, pour enregistrer l'état de la population. Les boucles F (19-24) et G (20-24) sont utilisées en début de jeu pour définir la population initiale (lignes 21 et 22). Le calcul de la nouvelle population est ensuite réalisé à l'aide d'une quadruple boucle : 7-16, 8-16, 10-13, 11-13. La ligne 12 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 (19,24) et G (20,24).

La ligne 25 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 CTRL et Break. 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 RANDOMIZE TIMER:CLS:DEFINT a-z
3 INPUT"DIMENSION :";a
4 DIM b(a,a):DIM c(a,a):COLOR 3,1:CLS
5 i=0:GOTO affiche
6 calcul:
7 FOR f=1 TO a
8 FOR g=1 TO a
9 s=0
10 FOR i=-1 TO 1
11 FOR j=-1 TO 1
12 IF f+j<>0 AND g+i<>0 AND f+j<>a+1 AND g+i<>a+1 THEN
    s=s+b(f+j,g+i)
13 NEXT j,i
14 s=2*s-b(f,g):c(f,g)=0
15 IF s>=5 AND s<=7 THEN c(f,g)=1
16 NEXT g,f
17 affiche:
18 CLS:k=0
19 FOR f=1 TO a
20 FOR g=1 TO a
21 IF RND<.5 AND i=0 THEN c(f,g)=1
22 LOCATE 2*f+1,2*g+1:COLOR 3:IF c(f,g)=1 THEN k=1:PRI
    NT "*":SOUND INT(RND*3000+400),1
23 b(f,g)=c(f,g)
24 NEXT g,f
25 IF k=1 THEN calcul ELSE PRINT "C'est fini....":SOUN
    D 700,10

```

```

1 REM LE JEU DE LA VIE (MANUEL)
2 RANDOMIZE TIMER:CLS:DEFINT a-z
3 INPUT"DIMENSION :";a
4 IF a<2 OR a>11 OR a<>INT(a) THEN RUN
5 DIM b(a,a):DIM c(a,a):COLOR 3,1:CLS
6 i=0:GOTO affiche
7 calcul:
8 FOR f=1 TO a
9 FOR g=1 TO a
10 s=0
11 FOR i=-1 TO 1
12 FOR j=-1 TO 1
13 IF f+j<>0 AND g+i<>0 AND f+j<>a+1 AND g+i<>a+1 THEN
    s=s+b(f+j,g+i)
14 NEXT j,i
15 s=2*s-b(f,g):c(f,g)=0
16 IF s>=5 AND s<=7 THEN c(f,g)=1
17 NEXT g,f
18 affiche:
19 CLS:k=0
20 FOR f=1 TO a
21 FOR g=1 TO a
22 IF i=0 THEN
23 COLOR 2:LOCATE 4,50:PRINT "Ligne :";f
24 LOCATE 6,50:PRINT "Colonne :";g
25 COLOR 0:LOCATE 2*f-1,3*g+1:PRINT "_";
26 COLOR 2:LOCATE 8,50:PRINT "Taper 0 ou 1";
27 r$=INPUT$(1):c(f,g)=VAL(r$)
28 END IF
29 LOCATE 2*f-1,3*g+1:COLOR 3
30 IF c(f,g)=1 THEN
31 k=1:PRINT "*":SOUND INT(RND*3000+400),1
32 END IF
33 b(f,g)=c(f,g)
34 NEXT g,f
35 IF k=1 THEN calcul ELSE PRINT "C'est fini....":SOUN
    D 700,10

```

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

La boucle I (7-10) permet d'entrer ce nombre.

La boucle I (15-18) imprime les tas.

La ligne 19 assure l'affichage du joueur.

La ligne 22, ainsi que les lignes 23 et 26, contrôlent si le coup est autorisé.

La ligne 12 contrôle la fin de la partie (en ligne 31).

POUR JOUER :

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

On entrera ensuite le nombre d'allumettes dans chaque tas (faire à chaque entrée N suivi de RETURN). 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 RETURN. 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.

EXTENSIONS POSSIBLES :

On peut désigner chaque joueur par son nom, faire un graphisme plus élaboré, écrire un programme permettant à l'Amiga de servir de partenaire, etc.

```

1  REM LE JEU DE MARIENBAD
2  COLOR 0,1:CLS
3  choix:
4  INPUT "NOMBRE DE TAS ( <=7 ) :";n
5  IF n<>INT(n) OR n<2 OR n>8 THEN choix
6  s=0:DIM a(n)
7  FOR i=1 TO n
8  PRINT:PRINT "TAS ";i;:INPUT "NOMBRE D'ALLUMETTES :";a(i)
9  s=s+a(i)
10 NEXT i
11 h=1
12 WHILE s<>0
13  entre1:
14  CLS
15  FOR i=1 TO n
16  LOCATE 5,5*i+3:COLOR 2:PRINT a(i)
17  LOCATE 7,1:COLOR 3:PRINT "TAS":LOCATE 7,5*i+3:COLOR 0:PRINT i
18  NEXT i
19  LOCATE 11,1:COLOR 0:PRINT "JOUEUR ";:COLOR h+1:PRINT T h
20  SOUND INT(RND*4000+500),2
21  COLOR 0:PRINT:INPUT "DANS QUEL TAS :";t
22  IF t>n OR t<>INT(t) OR t<1 THEN entre1
23  IF a(t)=0 THEN entre1
24  entre2:
25  LOCATE 15,1:INPUT "QUANTITE A ENLEVER :";o
26  IF o>a(t) OR o=0 THEN entre2
27  a(t)=a(t)-o
28  s=s-o
29  h=h+1:IF h=3 THEN h=1
30  WEND
31  PRINT:PRINT"LE JOUEUR ";:COLOR h+1:PRINT h;:COLOR 0 :PRINT "A GAGNE"
32  END

```

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 de l'Amiga supposé de qualité convenable (on pourra aisément utiliser tout autre générateur, si besoin est). La boucle I (3-11) s'occupe de l'entrée de chacune des probabilités, avec vérification à chaque entrée et sur le total (ligne 8) ; chaque probabilité est conservée dans une variable indicée E (I) : ligne 6 du programme.

Le lancer de dé truqué est effectué à l'aide d'une autre boucle I (16-18), la ligne permettant le trucage se trouvant en 17. Un nouveau tirage pourra être effectué sur pression d'une touche (ligne 21).

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 CTRL et Break.

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  COLOR 1,3:CLS:p=0:DIM e(6)
3  FOR i=1 TO 6
4  tirde:
5  LOCATE 3,18:PRINT SPACE$(30)
6  LOCATE 3,1:PRINT"PROBABILITE DU DE ";i;:INPUT e(i)
7  p=p+e(i):a$=SPACE$(5):e$=STR$(e(i)):RSET a$=e$
8  IF e(i)<0 OR e(i)>1 OR p>1.1 THEN GOSUB faux:GOTO t
   irde
9  COLOR 0:LOCATE 7,6*i-4:PRINT"DE ";i
10 COLOR 1:LOCATE 9,6*i-5:PRINT a$
11 NEXT i
12 IF p<.9 THEN BEEP:RUN
13 FOR i=1 TO 500:NEXT i
14 tirage:
15 n=INT(RND*6+1)
16 FOR i=1 TO 6
17 IF n=i AND RND>e(i) THEN GOTO tirage
18 NEXT i
19 LOCATE 15,1:PRINT"TIRAGE :";n;"           ":SOUND I
   NT(RND*800+200),5
20 PRINT:PRINT"FAIRE - ENTREE - POUR UN AUTRE TIRAGE"
21 r$=INPUT$(1)
22 GOTO tirage
23 faux:
24 LOCATE 4,1:PRINT"IMPOSSIBLE :0<= P <=1"
25 p=p-e(i):BEEP
26 FOR j=1 TO 3000:NEXT j
27 LOCATE 4,1:PRINT SPACE$(24)
28 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 (8-10) dessine une portée.

La note est choisie en ligne 11 et affichée en lignes 12-13 et 14.

La ligne 15 affiche le score.

La ligne 16 vous interroge.

La ligne 6 s'occupe de la fin de la partie.

Les lignes 17 et 18 effectuent le décodage de la note entrée.

La ligne 20 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 RETURN.

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 RETURN, 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, le son. On pourra aussi réaliser suivant le même principe un jeu vous permettant de reconnaître des accords, etc.


```

1  REM SOLFEGE
2  CLS:SCREEN 1,320,200,5,1:RANDOMIZE TIMER:DEFINT a-z
3  WINDOW 1,"SOLFEGE",,15,1:PALETTE 0,0,0,0:COLOR ,0:CLS
4  FOR i=1 TO 15:PALETTE i,RND,RND,RND:NEXT i
5  s=0:p=0:s$="993880784701659587523494440392349329293
   261247"
6  WHILE r$<>"0"
7  CLS:COLOR INT(RND*15+1)
8  FOR i=1 TO 5
9  LOCATE 2*i+1,1:FOR j=1 TO 37:PRINT "-";:NEXT j
10 NEXT i
11 x=INT(RND*12)+1:COLOR INT(RND*15+1)
12 LOCATE x,18:PRINT "|"
13 LOCATE x+1,18:PRINT "|"
14 LOCATE x+2,17:PRINT"0"
15 COLOR INT(RND*15+1):LOCATE 18,1:PRINT"SCORE ";s;" S
   UR ";p;" ESSAIS"
16 COLOR INT(RND*15+1):INPUT"QUELLE EST LA NOTE :";r$:
   r$=UCASE$(r$)
17 y=(1 AND r$="FA")+(2 AND r$="MI")+(3 AND r$="RE")+(
   4 AND r$="DO")
18 y=y+(5 AND r$="SI")+(6 AND r$="LA")
19 p=p+1
20 IF y=x OR y=x-7 THEN s=s+1:SOUND VAL(MID$(s$,3*x-2,
   3)),30
21 WEND

```

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 3 à 6 contrôlent les entrées. La ligne 4 s'occupe de la fin de partie. La ligne 8 permet une correction pour les dates antérieures à 1582 (mise en service du calendrier actuel).

Les lignes 9 et 10 permettront l'affichage en clair du mois dans la réponse. La formule utilisée apparaît aux lignes 16 et 17.

Les lignes 18 et 19 permettront l'affichage en clair du jour dans la réponse.

La ligne 20 imprime les résultats.

POUR JOUER :

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

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  COLOR 3,1:CLS
3  INPUT"JOUR.....":j
4  IF j=0 THEN END
5  PRINT:INPUT"MOIS.....":m
6  PRINT:INPUT"ANNEE.....":a
7  b=a:s=INT(a/100)-INT(a/400)
8  IF a<=1582 THEN s=2
9  m$="JANVIER  FEVRIER  MARS      AVRIL    MAI      JU
   IN          "
10 m$=m$+"JUILLET  AOUT      SEPTEMBRE OCTOBRE  NOVEMBRE
    DECEMBRE "
11 m$=MID$(m$,9*m-8,9):r$=""
12 FOR i=1 TO LEN(m$)
13 IF MID$(m$,i,1)<>" " THEN r$=r$+MID$(m$,i,1)
14 NEXT i
15 IF m<3 THEN m=m+12:a=a-1
16 s=j-1+INT(13*(m+1)/5)+INT(5*a/4)-s
17 s=s-7*INT(s/7)
18 j$="DIMANCHE LUNDI  MARDI  MERCREDI JEUDI  VENDRED
   ISAMEDI  "
19 j$=MID$(j$,8*s+1,8)
20 LOCATE 12,1:PRINT"LE ";:COLOR 0:PRINT j;": ";r$;b;:C
   OLOR 3:PRINT " ETAIT UN ";:COLOR 0:PRINT j$
21 FOR i=800 TO 200 STEP -20:SOUND i,2:NEXT i
22 COLOR 2:PRINT:PRINT:PRINT"APPUYEZ SUR UNE TOUCHE":r
   $=INPUT$(1)
23 RUN

```

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 3.

La ligne 13 affiche le numéro de joueur et le nombre d'essais déjà effectués.

Un sous-programme placé en ligne 39, utilisé trois fois dans la boucle J (21-25), réalise le lancer de dés.

Les retirages sont effectués par les lignes 33-35 et 37.

La variable R\$ gère les 2^e et 3^e tirages. L'impression du résultat est réalisée dans tous les cas par les lignes 14 et 15.

Les tests pour le nombre d'essais et pour le changement de joueur sont effectués aux lignes 26 et 28.

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 RETURN. 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 RETURN. 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 RETURN : par exemple, pour retirer A et C on tapera AC suivi de RETURN.

Les joueurs suivants procéderont de la même façon.

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 CLS:RANDOMIZE TIMER:DEFINT a-z
3 INPUT"NOMBRE DE JOUEURS :";n
4 p=1:COLOR 3,1
5 debut:
6 i=1
7 WHILE i<=n
8 CLS:k=0:r$="ABC"
9 GOTO affiche
10 essai:
11 k=k+1
12 LOCATE 1,17:COLOR 2:PRINT "-421-":COLOR 3
13 LOCATE 4,1:PRINT"Joueur NO. ";i;" - ESSAI ";k;" - P
   ARTIE ";p
14 LOCATE 7,7:PRINT a :LOCATE 7,17:PRINT B:LOCATE 7,27
   :PRINT C
15 LOCATE 9,8:PRINT"A":LOCATE 9,18:PRINT"B":LOCATE 9,2
   8:PRINT"C"
16 r$="":LOCATE 14,1:PRINT"          "
17 LOCATE 14,1:INPUT"ABC :";r$:r$=UCASE$(r$)
18 IF r$="" THEN GOTO suivant
19 r$=r$+"DDD"
20 affiche:
21 FOR j=1 TO 3
22 GOSUB tirde
23 x$=MID$(r$,j,1):x=ASC(x$)-64
24 ON x GOSUB premier,deuxieme,troisieme,tirde
25 NEXT j
26 IF k<3 THEN GOTO essai
27 suivant:
28 i=i+1
29 WEND
30 p=p+1
31 INPUT "ON CONTINUE ( O/N ) : ";r$
32 IF r$<>"O" AND r$<>"o" THEN END ELSE GOTO debut
33 premier:
34 a=D:RETURN
35 deuxieme:
36 B=D:RETURN
37 troisieme:
38 C=D:RETURN
39 tirde:
40 D=INT(RND*6+1):SOUND 200*D+400,2:RETURN

```

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 (5-7) dessine la portée. Une deuxième boucle I (8-11) affiche les numéros des lignes.

La ligne 12 définit le nom des notes, et la ligne 14 réalise le tirage aléatoire d'une note.

Le nom de la note est calculé à la ligne 15 et il est affiché par la ligne 16, suivi de la mention GRAVE ou AIGUE, selon le cas.

La réponse sera entrée en ligne 18.

La ligne 19 s'occupe du décodage de la réponse.

La ligne 20 affiche la note.

Les lignes 22 et 23 regardent si l'on a gagné ou perdu et affichent le message correspondant.

La ligne 24 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 RETURN.

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

On doit taper la touche 5 pour continuer, et s'arrêter en tapant CTRL et Break, ou une touche quelconque.

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  RANDOMIZE TIMER:DEFINT a-z
3  WHILE r$<>"S" AND r$<>"s"
4  COLOR 2,1:CLS
5  FOR i=1 TO 5
6  LOCATE 2*i+5,5:FOR j=1 TO 35:PRINT "-";:NEXT j
7  NEXT i
8  FOR i=1 TO 14
9  LOCATE i+3,1:COLOR 0
10 IF i<=9 THEN PRINT i ELSE PRINT " ";CHR$(i+55)
11 NEXT i
12 p$="SI LA SOLFA MI RE DO "
13 n$="993880784701659587523494440392349329293261247"
14 x=INT(RND*14+1)
15 t=x-1-7*INT((x-1)/7)
16 COLOR 3:LOCATE 19,1:PRINT"NOTE ";MID$(p$,3*t+1,3);
17 IF x<=7 THEN PRINT" AIGUE" ELSE PRINT " GRAVE"
18 LOCATE 19,30:INPUT"NUMERO DE LA LIGNE :";l$
19 l=- (ASC(l$)-48)*(l$<"")-(ASC(l$)-55)*(l$>"A")+32*(l$>"a")
20 LOCATE x+1,16:PRINT "!":LOCATE x+2,16:PRINT "!":LOCATE x+3,15:PRINT"0"
21 LOCATE 20,1
22 IF l=x THEN COLOR 2:PRINT"BRAVO";:SOUND VAL(MID$(n$,3*x-2,3)),30
23 IF l<>x THEN COLOR 3:PRINT"PERDU";
24 LOCATE 20,15:COLOR 0:PRINT"TAPER - S - POUR ARRETER";:r$=INPUT$(1)
25 WEND

```

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 à l'Amiga 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 5 : 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 (6-9) lit les lignes DATA (20-22) concernant les préfixes suivis des suffixes.

La ligne 11 tire au hasard préfixe et suffixe.

Le mot obtenu en ligne 12 est affiché par la boucle L (13-17) et K (14-16) lettre par lettre, puis on recommence (ligne 19).

POUR JOUER :

Lancer l'exécution du programme par RUN. Un premier mot sera immédiatement affiché, puis un nouveau mot sera affiché à la suite du précédent.

Taper CTRL et Break 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 à l'Amiga...


```

1 REM MACHINAMOT
2 SCREEN 1,300,200,4,1
3 WINDOW 1,"MACHINAMOT",(1,1)-(275,180),15,1
4 RANDOMIZE TIMER:DEFINT a-z:COLOR ,0:CLS
5 DIM a$(15),b$(15)
6 FOR i=1 TO 15
7 PALETTE i,RND,RND,RND
8 READ a$(i),b$(i)
9 NEXT i
10 debut:
11 i=INT(RND*15+1):j=INT(RND*15+1)
12 c#=a$(i)+b$(j)
13 FOR l=1 TO LEN(c$)
14 FOR k=1 TO 20-1
15 LOCATE 20,k:COLOR INT(RND*15+1):PRINT " "+MID$(c$,L
    EN(c$)-l+1,1);
16 NEXT k
17 SOUND INT(RND*2000)+400,2:NEXT 1
18 FOR s=1 TO 1200:NEXT s:PRINT
19 GOTO debut
20 DATA ANDRO,CRATE,COSMO,PHILE,MICRO,PHOBE,HELIO,TROP
    E,SEXO,GYNE,STEREO,MANE
21 DATA PALEO,NAUTE,MACRO,PHAGE,HYDRO,FUGE,STENO,PATHE
    ,PATHO,CARDE,OLEO,DERME
22 DATA MYTHO,SCOPE,HOMEO,VORE,OCTO,PHONE

```

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 :

Les lignes 6 et 7 définissent un jeu de 52 cartes.

La table B\$, dimensionnée en ligne 3, 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 (8-22) réalise le tirage de quatre jeux de 13 cartes.

La boucle I (9-19) réalise tirage et impression. La carte est choisie parmi les 52 disponibles, en ligne 11.

La ligne 12 calcule la couleur de la carte tirée.

La ligne 13 vérifie que cette carte n'a pas déjà été tirée.

La ligne 16 s'occupe 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 15 et 17.

La ligne 18 permet d'éliminer les cartes déjà tirées, et la ligne 20 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.

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 de l'Amiga, par exemple), ou introduire des possibilités de jeu de la carte, y compris pour l'Amiga.

```

1  REM BRIDGE
2  SCREEN 1,640,200,4,2:WINDOW 1,"BRIDGE",,2,1
3  RANDOMIZE TIMER:COLOR 1,2:CLS:DIM b$(9)
4  b$(1)="TREFLE":b$(2)="PIQUE":b$(3)="COEUR":b$(4)="C
   ARREAU"
5  b$(5)="0":b$(6)="A":b$(7)="A":b$(8)="0":b$(9)="S"
6  r$="234567891VDRA":c(1)=0:c(2)=14:c(3)=3:c(4)=9
7  r$=r$+r$+r$+r$
8  FOR j=1 TO 4
9  FOR i=1 TO 13
10 tirage:
11 x=INT(RND*LEN(r$)+1)
12 s=INT((x-1)/13)+1
13 IF MID$(r$,x,1)="0" THEN tirage
14 t=x-13*INT(x/13):t=t-13*(t=0)
15 LOCATE i+1,16*j-14:COLOR 1:PRINT MID$(r$,x,1);
16 IF t>=9 THEN PRINT b$(t-4)
17 COLOR c(s):LOCATE i+1,16*j-10:PRINT b$(s)
18 MID$(r$,x,1)="0":SOUND INT(RND*4000+500),1
19 NEXT i
20 LOCATE 19,1:COLOR 1:PRINT"APPUYEZ SUR UNE TOUCHE":t
   $=INPUT$(1)
21 LOCATE 19,1:PRINT SPACE$(30)
22 NEXT j

```

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.

Le générateur aléatoire de l'appareil est initialisé à partir du code introduit par le joueur ; 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 grâce à une boucle WHILE-WEND (lignes 5 à 7) ; le message est entré à la ligne 9.

La ligne 10 permet de choisir entre les options de codage et de décodage.

La boucle I (12-20) est utilisée soit pour effectuer le codage, soit pour effectuer le décodage.

La ligne 14 appelle le générateur aléatoire.

La ligne 15 réalise la transposition des lettres.

Les lignes 16 et 17 vérifient que l'on reste bien dans les limites autorisées.

Les résultats seront imprimés par la ligne 21.

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 RETURN.

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

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

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  CLS
3  PRINT"DONNEZ VOTRE CODE SECRET : ";
4  s$="":r$=""
5  WHILE r$<>CHR$(13)
6  s$=s$+r$:r$=INPUT$(1):IF r$<>CHR$(13) THEN PRINT "X
";
7  WEND
8  RANDOMIZE VAL(s$)
9  PRINT:LINE INPUT"VOTRE MESSAGE :";a$:a$=UCASE$(a$)
10 PRINT:PRINT"VOULEZ-VOUS (C)ODER OU (D)ECODER :";r$
    =INPUT$(1)
11 b$=""
12 FOR i=1 TO LEN(a$)
13 s=ASC(MID$(a$,i,1))
14 a=INT(RND*26+1)
15 t=s-a*((r$="C" OR r$="c")-(r$="D" OR r$="d"))
16 IF t>90 THEN t=t-26
17 IF t<65 THEN t=t+26
18 IF t>=65 AND t<=90 AND s>=65 THEN s$=CHR$(t) ELSE s
    $=" "
19 b$=b$+s$
20 NEXT i
21 PRINT:PRINT"LE RESULTAT EST : ";b$

```

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 l'Amiga 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 4 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 en ligne 6.

La boucle I (8-12) imprime des tirets aux emplacements des lettres à trouver.

La boucle en ligne 16 sert uniquement à simuler le temps de recherche d'une lettre.

La formule, placée en ligne 18, permet de modifier légèrement la probabilité de tirage, en faveur des voyelles et des consonnes usuelles.

La ligne 20 provoque un nouveau tirage, et la boucle I (24-27) réimprime les tirets et les lettres trouvées.

La ligne 28 incrémente le compteur d'essais.

La ligne 30 affiche le nombre d'essais réalisés, en fin de partie.

POUR JOUER :

Lancer le programme par RUN.

Entrer le mot à découvrir, suivi de RETURN.

Des tirets sont alors affichés (un par lettre) et l'Amiga 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 CTRL et Break.

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 l'Amiga ; on pourra aussi imaginer d'ajouter un affichage graphique, par exemple une potence ou une voiture qui avance vers un précipice, etc.

```

1  REM PENDU INVERSE
2  SCREEN 1,320,200,4,1:WINDOW 1,"PENDU INVERSE",,2,1
3  RANDOMIZE TIMER:COLOR 3,2:CLS
4  d$="YAEIOUBCDFGLMNPSTVWHKJQXZ"
5  c=1:t$="":u=1:s=18
6  INPUT"ENTREZ VOTRE MOT :";r$:r$=UCASE$(r$)
7  CLS
8  FOR i=1 TO LEN(r$)
9  IF MID$(r$,i,1)>="a" THEN MID$(r$,i,1)=CHR$(ASC(MID
$(r$,i,1))-32)
10 LOCATE 6,i+1:PRINT "-"
11 t$=t$+"-"
12 NEXT i
13 LOCATE 16,1:PRINT "LETTRES ESSAYEES SANS SUCCES"
14 WHILE t$<>r$
15 COLOR 1:LOCATE 10,1:PRINT"JE CHERCHE....; ESSAI ";c
16 FOR j=1 TO 1000:NEXT j
17 tirage:
18 x=-6*(RND<.7)-13*(RND<.5)-7*(RND<.4)
19 x=INT(x*RND+1)
20 IF MID$(d$,x,1)="-" THEN GOTO tirage
21 e$=MID$(d$,x,1)
22 MID$(d$,x,1)="-":k=0
23 COLOR 0:PRINT:PRINT:PRINT"JE PRENDS : ";:COLOR 3:PR
INT e$:SOUND INT(RND*500+300),1
24 FOR i=1 TO LEN(r$)
25 IF e$=MID$(r$,i,1) THEN k=1:MID$(t$,i,1)=e$:SOUND 2
00,3
26 COLOR 0:LOCATE 6,i+1:PRINT MID$(t$,i,1):COLOR 3
27 NEXT i
28 c=c+1:IF k=0 THEN LOCATE s,u:PRINT e$:u=u+2:IF u>=3
8 THEN u=1:s=s+1
29 WEND
30 COLOR 0:LOCATE 20,1:PRINT"TROUVE EN ";c-1;" COUPS":
END

```

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 6 appelle un sous-programme (lignes 26 à 32) qui donne la règle du jeu.

La ligne 8 fabrique les chiffres, qui sont ensuite stockés dans une variable alphanumérique A\$.

La ligne 9 permet l'impression du dernier chiffre au hasard sur la cinquième ligne de l'écran, et la durée de l'affichage est déterminée par la ligne 10.

La ligne 12 attend la réponse du joueur, qui est comparée à la réponse correcte en ligne 13.

La ligne 16 marque les points.

La ligne 17 contrôle le meilleur score.

Les affichages de fin de partie sont effectués de la ligne 19 à la ligne 24.

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 RETURN. 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 RETURN.

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 CTRL et Break avant l'affichage d'un chiffre.

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  SCREEN 2,640,200,4,2:WINDOW 1,"SIMONS",(0,0)-(420,1
   80),31,2
3  RANDOMIZE TIMER:DEFINT a-z:m=0:COLOR 1,2:CLS
4  FOR i=3 TO 15:PALETTE i,RND,RND,RND:NEXT i
5  debut:
6  a$="":h=0: GOSUB texte
7  tirage:
8  x=INT(RND*10):a$=a$+RIGHT$(STR$(x),1)
9  CLS:COLOR 1:LOCATE 5,RND*40+1:PRINT x
10 FOR i=1 TO 700+RND*900:NEXT i
11 CLS:COLOR 4
12 LOCATE 11,1:INPUT"VOTRE SEQUENCE ":"r$
13 IF r$<>a$ THEN fin
14 PRINT:PRINT"EXACT . ATTENTION"
15 FOR i=1 TO 1000:NEXT i
16 h=h+1
17 IF m<h THEN m=h
18 GOTO tirage
19 fin:
20 COLOR 0:CLS
21 PRINT"VOTRE SCORE EST DE ";h
22 PRINT:PRINT"MEILLEUR SCORE ACTUEL ";m
23 PRINT:PRINT"AUTRE PARTIE (O/N)":r$=UCASE$(INPUT$(1)
   )
24 IF r$="N" THEN END
25 GOTO debut
26 texte:
27 k=4:b$="VOUS DEVEZ REPETER TOUTE LA SERIE DES":GOSU
   B affiche
28 b$="NOMBRES APPARUS DEPUIS LE DEBUT DU JEU":GOSUB a
   ffiche
29 b$="IL Y EN A UN DE PLUS A CHAQUE FOIS":GOSUB affic
   he
30 b$="APPUYEZ SUR UNE TOUCHE POUR COMMENCER":GOSUB af
   fiche
31 r$=INPUT$(1)
32 RETURN
33 affiche:
34 FOR i=1 TO LEN(b$)
35 LOCATE k,i:COLOR INT(RND*13+3):PRINT MID$(b$,i,1)
36 SOUND INT(RND*4000+500),1
37 NEXT i
38 k=k+3
39 RETURN

```

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 l'Amiga 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 à l'Amiga 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 ; ceci est fait encore une fois et, au 4^e affichage, l'Amiga est en mesure de deviner la carte choisie.

LE PROGRAMME :

La boucle I (5-12) permet de choisir, au hasard, 21 cartes sur 52, et la boucle emboîtée J (8-10) permet de s'assurer que l'on ne réalise pas deux fois le même tirage. La ligne 15 contient les couleurs des cartes ainsi que les noms des cartes figures (VALET, DAME, ROI, AS), dans une variable alphanumérique B\$.

La boucle I (16-24) réalise l'affichage du jeu.

La ligne 17 décode la couleur de la carte.

La ligne 18 choisit la position d'affichage.

La ligne 19 décode la hauteur.

L'affichage proprement dit est effectué de la ligne 20 à la ligne 23.

La ligne 26 permet au joueur d'entrer le numéro de la colonne où se trouve sa carte. La boucle J (28-30) réalise le brouillage, et les boucles emboîtées J (33-36) et I (34-36) 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 RETURN.

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 CTRL et Break à la place du numéro de la colonne.

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  SCREEN 1,620,200,4,2:WINDOW 1,"TOUR DE CARTES",,2,1
3  COLOR 0,11:CLS:RANDOMIZE TIMER
4  a$="":c(0)=4:c(1)=3:c(2)=9:c(3)=14
5  FOR i=1 TO 21
6    tircarte:
7    b$=CHR$(INT(RND*52+38))
8    FOR j=1 TO LEN(a$)
9      IF b$=MID$(a$,j,1) THEN GOTO tircarte
10   NEXT j
11   a$=a$+b$
12   NEXT i
13   h=3:p=1
14   debut:
15   b$="TREFLE CARREAUCEUR PIQUE VALET DAME ROI
      AS      "
16   FOR i=1 TO LEN(a$)*p-11*(p=0)
17     s=INT((ASC(MID$(a$,i,1))-38)/13)
18     j=7*INT((i-1)/7):SOUND INT(RND*3000+400),1
19     t=ASC(MID$(a$,i,1))-37-13*s
20     IF p=0 THEN COLOR 0:LOCATE 1,1:PRINT"VOTRE CARTE ES
      T :      "
21     LOCATE (i-j)*p+1,3*j*p-20*(p=0)+1:COLOR 8
22     IF t<=9 THEN PRINT STR$(t+1); ELSE PRINT MID$(b$,AB
      S(7*t-41),7);
23     LOCATE (i-j)*p+1,(3*j+5)*p+2-25*(p=0):COLOR c(s):PR
      INT MID$(b$,7*s+1,7)
24   NEXT i
25   IF p=0 THEN END
26   LOCATE 17,1:COLOR 0:INPUT"QUELLE COLONNE :";n
27   CLS:h=h-1:r$="20120":b$=""
28   FOR j=0 TO 2
29     b$=b$+MID$(a$,7*VAL(MID$(r$,n+j,1))+1,7)
30   NEXT j
31   IF h=0 THEN a$=b$:p=0:GOTO debut
32   a$=""
33   FOR j=0 TO 2
34     FOR i=1 TO 7
35       a$=a$+MID$(b$,3*i-2+j,1)
36     NEXT i,j
37   GOTO debut
```

Course de chevaux

REFERENCES :

Difficulté de programmation : moyenne.

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 l'Amiga 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 :

Une première boucle (6-8) affiche le terrain de jeu.

Une deuxième boucle I (14-24) fera avancer les chevaux ; la position de chaque cheval sera repérée par un chiffre ou une lettre dans une variable alphanumérique A\$ (ligne 11).

La ligne 17 fera avancer les chevaux.

La ligne 18 gère les arrivées.

La ligne 20 réalisera les impressions.

Le sous-programme 28-32 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 RETURN.

La course se déroulera ensuite automatiquement.

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  SCREEN 1,320,200,5,1:WINDOW 1,"COURSE DE CHEVAUX",,
   15,1
3  RANDOMIZE TIMER:PALETTE 0,0,0,0:PALETTE 1,1,1,1:COL
   OR 1,0
4  CLS
5  INPUT"NOMBRE DE CHEVAUX :";n:IF n>9 THEN RUN
6  FOR i=1 TO n
7  LOCATE 2*i+1,1:COLOR ,n-i+7:PRINT SPACE$(40)
8  NEXT i
9  a$="":p=0:a=0:t=1:k=0:j=t
10 entree:
11 a$=a$+"0":i=j
12 GOTO place
13 debut:
14 FOR i=1 TO n
15 IF MID$(a$,i,1)="H" THEN fin
16 b$=CHR$(ASC(MID$(a$,i,1))+INT(RND*4+1))
17 a$=LEFT$(a$,i-1)+b$+RIGHT$(a$,LEN(a$)-i)
18 IF b$>="H" THEN GOSUB arrive
19 place:
20 LOCATE 2*i+1,ASC(MID$(a$,i,1))-47:COLOR i+1,n-i+7:P
   RINT " ";i
21 SOUND 250,1:FOR d=1 TO 500:NEXT d:SOUND 800,1
22 IF k=0 THEN j=j+1:IF j<=n THEN entree ELSE k=1:GOTO
   debut
23 fin:
24 NEXT i
25 p=p+t:t=0
26 IF a<n THEN debut
27 END
28 arrive:
29 MID$(a$,i,1)="H"
30 COLOR 0:LOCATE 2*i+1,1:PRINT p
31 t=1:a=a+t
32 RETURN

```

Idem

REFERENCES :

Difficulté de programmation : moyenne.

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

Difficulté du jeu : selon le niveau choisi.

LE JEU :

L'Amiga 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 l'Amiga 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 :

Une variable numérique N est dimensionnée à 2 en ligne 4, pour contenir les résultats successifs des deux joueurs.

La ligne 5 permettra d'entrer le niveau de difficulté choisi.

Le nombre initial est confectionné par la ligne 7. La quantité qui interviendra dans les décrémentations est obtenue en ligne 9.

La ligne 13 teste si le résultat n'est pas trop petit.

La ligne 14 affiche le numéro du joueur.

Le nombre est affiché en ligne 15 et le jeu du joueur est entré en ligne 18 (nombre de deux chiffres).

La ligne 20 incrémente la colonne choisie, et la ligne 21 décrémente le nombre de la quantité choisie au départ.

Le résultat sera affiché un court instant (ligne 24). La boucle I (28-30) permet de vérifier si tous les chiffres sont identiques.

L'impression et la fin de partie sont effectuées en ligne 31. La ligne 29 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 RETURN ; 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) sera alors affiché un court instant, et le trait passe au deuxième joueur, et ainsi de suite. On peut arrêter à tout moment en tapant CTRL et Break.

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  SCREEN 1,320,200,4,1:WINDOW 1,"IDEM",,2,1
3  RANDOMIZE TIMER:DEFINT a-z:COLOR 1,4:CLS
4  h=1:DIM n(2)
5  PRINT:INPUT "NIVEAU (2-7) :";n
6  u=10:p=0
7  n(1)=INT((u^n-u^(n-1))*RND+u^(n-1))
8  n(2)=n(1)
9  a=INT(2*u^(n-1)*RND+u^(n-1))
10 debut:
11 IF h=1 THEN p=p+1
12 CLS
13 IF n(h)<u^(n-1) THEN n(h)=u^n-ABS(n(h))
14 COLOR 4*h-1:SOUND 300*h,2:PRINT "JOUEUR ";h;"  ESS
AI ";p
15 COLOR 5:LOCATE 7,1:PRINT "NOMBRE :":LOCATE 7,16-n:CO
LOR 11:PRINT n(h)
16 COLOR 6:LOCATE 8,10:PRINT "7654321":COLOR 5:r$=""
17 WHILE LEN(r$)<>2 OR VAL(RIGHT$(r$,1))>n
18 LOCATE 11,1:INPUT "VOTRE JEU ( FORME NC ) :";r$
19 WEND
20 n(h)=n(h)+INT(VAL(LEFT$(r$,1))*u^(VAL(RIGHT$(r$,1))
-1))
21 n(h)=n(h)-a
22 IF n(h)>u^n THEN n(h)=n(h)-u^n
23 n(h)=INT(n(h))
24 LOCATE 14,1:PRINT "RESULTAT :";:COLOR 9:PRINT n(h)
25 FOR j=1 TO 2200:NEXT j
26 r$=STR$(n(h))
27 r$=RIGHT$(r$,LEN(r$)-1)
28 FOR i=1 TO LEN(r$)-1
29 IF MID$(r$,i,1)<>MID$(r$,i+1,1) THEN h=ABS(3-h):GOT
O debut
30 NEXT i
31 COLOR 4*h-1:PRINT:PRINT "BRAVO, JOUEUR ";h
32 FOR i=200 TO 900 STEP 20:SOUND i,2:NEXT i

```

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 5, on crée un tableau pouvant contenir les 20 cartes.

La boucle I (8-13) assure le tirage aléatoire de 20 nombres, pris parmi 32 ; la boucle J (11-13) vérifie que l'on ne réalise pas deux fois le même tirage.

La deuxième boucle I (15-24) assure ensuite l'affichage, les noms des cartes étant affichés en clair, et la couleur étant décodée et affichée en clair, grâce aux lignes 16 à 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.

EXTENSIONS POSSIBLES :

On pourra facilement mettre au point un affichage plus graphique en redéfinissant les caractères de l'Amiga ; 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 SCREEN 1,640,200,4,2:WINDOW 1,"BELOTE",,2,1
3 RANDOMIZE TIMER:COLOR 0,1:CLS
4 s$(5)="VALET":s$(6)="DAME":s$(7)="ROI":s$(8)="AS"
5 DIM a(20):s$(1)="TREFLE":s$(2)="COEUR":s$(3)="CARRE
  AU":s$(4)="PIQUE"
6 k=1:f(1)=14:f(2)=13:f(3)=3:f(4)=0:t$=""
7 LOCATE 12,20:PRINT " PATIENTEZ , JE TIRE "
8 FOR i=1 TO 20
9   tirage:
10  a(i)=1+INT(RND*32)
11  FOR j=1 TO i-1
12    IF a(i)=a(j) THEN tirage
13  NEXT j,i
14  CLS
15  FOR i=1 TO 20
16    t=-((a(i)<=8)+(a(i)<=16)+(a(i)<=24)+(a(i)<=32))
17    s=1+a(i)-8*INT(a(i)/8)
18    r$=STR$(s+6)+" ":r$=RIGHT$(r$,LEN(r$)-1)
19    r$=LEFT$(r$,2)
20    IF s>=5 THEN r$=s$(s)
21    l=13-LEN(r$)-LEN(s$(t)):COLOR 2
22    LOCATE 3*k,16*INT((i-1)/5)+1:PRINT r$;" ";:COLOR f(
      t):PRINT s$(t);LEFT$(t$,l)
23    k=k+1:k=k+5*(k=6):SOUND INT(4000*RND+50),1
24  NEXT i

```

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 l'Amiga) 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 4 permet d'obtenir l'entier de départ compris entre 5 et 20.

Une boucle I (9-19) permet d'entrer successivement les 5 opérations.

4 sous-programmes sont utilisés pour les calculs (aux lignes 28 à 39).

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 18).

La boucle I (21-25) 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 RETURN.

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).

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 l'Amiga participer davantage en lui confiant le rôle du codage, etc.

```

1  REM UN TAQUIN DE CALCUL
2  SCREEN 1,320,200,4,1:WINDOW 1,"TAQUIN DE CALCUL",,2
   ,1
3  RANDOMIZE TIMER:COLOR 8,15:CLS
4  x=INT(RND*16+5):SOUND INT(4000*RND+500),2
5  PRINT"NOMBRE INITIAL :";:COLOR 13:PRINT x
6  COLOR 9:PRINT:PRINT"ENTRER 5 OPERATIONS"
7  PRINT"SOUS LA FORME +5 , /2 , *4 ..."
8  f=x:DIM r$(5):COLOR 4
9  FOR i=1 TO 5
10 nouveau:
11 LOCATE 7,1:INPUT"VOTRE OPERATION :";r$(i)
12 LOCATE 7,17:PRINT SPACE$(10)
13 a=ASC(LEFT$(r$(i),1))-41
14 IF a<=0 OR a=3 OR a=5 OR a>6 THEN nouveau
15 IF a>3 THEN a=(a/2)+1
16 ON a GOSUB multi,addi,sous,divi
17 LOCATE 11,15:PRINT "          ":SOUND INT(RND*800
   +200),3
18 LOCATE 11,1:PRINT"NOMBRE OBTENU :";f
19 NEXT i
20 PRINT:PRINT"OPERATIONS :";
21 FOR i=1 TO 5
22 tirage:
23 x=INT(RND*5+1):IF t(x)=1 THEN tirage ELSE t(x)=1
24 COLOR(INT(RND*7)):PRINT r$(x);" ";
25 NEXT i
26 COLOR 0:PRINT:PRINT:PRINT"BON COURAGE...."
27 FOR i=100 TO 1000 STEP 20:SOUND i,1:NEXT i:END
28 multi:
29 f=f*VAL(RIGHT$(r$(i),LEN(r$(i))-1))
30 RETURN
31 addi:
32 f=f+VAL(RIGHT$(r$(i),LEN(r$(i))-1))
33 RETURN
34 sous:
35 f=f-VAL(RIGHT$(r$(i),LEN(r$(i))-1))
36 RETURN
37 divi:
38 f=f/VAL(RIGHT$(r$(i),LEN(r$(i))-1))
39 RETURN

```

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, par contre, les coups illicites seront refusés.

LE PROGRAMME :

La variable E\$, en ligne 4, servira à dessiner les disques, et D\$ dessine les socles.

Le nombre de disques est entré en ligne 6 et refusé s'il y a plus de 8 disques.

La boucle I (8-10) 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 13 et les trois piles par les boucles emboîtées I (15-19) et J (16-19).

Le contrôle du coup est effectué par la ligne 27 et l'état des piles est modifié par les lignes 28 et 29.

POUR JOUER :

Après un RUN qui lance le programme, on entre le nombre de disques demandé, suivi de RETURN. 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 RETURN 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.

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 HANOI
2  SCREEN 1,320,200,4,1:WINDOW 1,"LES TOURS DE HANOI",
   ,2,1:COLOR 2,15:CLS
3  DIM a$(3):a$(1)="9":a$(2)="9":a$(3)="9":s=1
4  d$=SPACE$(9):e$=STRING$(10,"#"):n=9
5  WHILE n>=9 OR n<>INT(n) OR n<1
6  INPUT"NOMBRE DE TOURS ( < 9 ):";n
7  WEND
8  FOR i=0 TO n-1
9  a$(1)=a$(1)+RIGHT$(STR$(n-i),1)
10 NEXT i:f$=a$(1)
11 debut:
12 CLS:COLOR 2:LOCATE 1,15:PRINT "ESSAI ";s
13 LOCATE 15,1:COLOR ,13:PRINT d$:LOCATE 15,12:PRINT d
   $:LOCATE 15,23:PRINT d$
14 LOCATE 15,5:COLOR 10,13:PRINT"1":LOCATE 15,16:PRINT
   "2":LOCATE 15,27:PRINT"3"
15 FOR i=0 TO n-1
16 FOR j=1 TO 3
17 b$=a$(j):l=VAL(MID$(b$,i+2,1))
18 IF LEN(b$)>i+1 THEN LOCATE 14-i,11*j-9:COLOR 1,15:P
   RINT LEFT$(e$,1)
19 NEXT j,i
20 entree:
21 SOUND 200,3:COLOR 4:LOCATE 17,1:INPUT"TOUR DE DEPAR
   T :";d
22 IF d<>INT(d) OR d>3 OR d<1 THEN entree
23 IF a$(d)="9" THEN entree
24 SOUND 600,2:PRINT:INPUT"TOUR D'ARRIVEE :";a
25 IF a<>INT(a) OR a=d OR a>3 OR a<1 THEN entree
26 b$=a$(d):c$=a$(a)
27 IF MID$(b$,LEN(b$),1)>MID$(c$,LEN(c$),1) THEN entre
   e
28 a$(a)=c$+MID$(b$,LEN(b$),1)
29 a$(d)=LEFT$(b$,LEN(b$)-1)
30 s=s+1
31 IF a$(3)<>f$ THEN debut
32 COLOR 2:LOCATE 19,1:PRINT "C'est fini.....
   "
33 FOR i=200 TO 1000 STEP 20:SOUND i,2:NEXT i

```


Niveau 3

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Compose	Faire de la musique, c'est d'abord l'écrire...
Jeu de loto	Gagnez sans problème, au prochain tirage.
Mastermind	Avec des chiffres, mais très classique. Un seul joueur.
Nicomaque	Pour se perfectionner en calcul mental, un ou plusieurs joueurs.
Cube	Jeu de dessin ; perspective sur un écran.
Von Koch	Une intéressante façon de déformer des figures.
A travers	Un jeu de stratégie pour deux joueurs.
Visions	Un jeu sur les mots, pour l'ordinateur.
Mathieu	Vous ne savez pas dessiner ? Laissez faire votre ordinateur.
Le loup et les agneaux	Un jeu de réflexion pour 2 joueurs.
Accrochez les wagons	Pour les plus jeunes ; très spectaculaire.
Le pendu	Un jeu de lettres très classique (2 joueurs).
Plusieurs en un	Un terrain de jeu et les pions, pour deux joueurs.
Chasse au sous-marin	Il vous faudra pas mal de réflexion, et beaucoup de chance.
Le lézard d'Alice	Beaucoup de hasard, mais très spectaculaire.
Dessin de dés	Améliorer vos jeux de dés en incorporant ce programme.
Chapeaux	Soyez très observateur et vous réussirez, pour deux joueurs.
Labyrinthix	Il vous faudra beaucoup de réflexion pour réussir.

Graphisme et jeux

LES INSTRUCTIONS GRAPHIQUES

L'Amiga fonctionne en permanence en mode graphique, texte et dessin pouvant être mélangés à volonté. L'instruction **SCREEN** permet de définir le mode de fonctionnement, en donnant la taille de l'écran et le nombre de couleurs.

Il existe quatre modes graphiques différents ; le nombre de couleurs pouvant varier de 2 à 32 suivant la quantité de mémoire réservée aux images.

Mode	320 × 200	non entrelacé basse résolution
	640 × 200	non entrelacé haute résolution
	320 × 400	entrelacé basse résolution
	640 × 400	entrelacé haute résolution.

L'instruction **PSET (x,y)**, **C** permet d'allumer un point en colonne **x**, ligne **y** sur l'écran. Le paramètre **C**, facultatif, indique la couleur. On peut effacer le point avec **PRESET (x, y)**.

L'instruction **LINE** va permettre soit de tirer un trait entre deux points désignés par leurs coordonnées, dans la couleur précisée, soit, en ajoutant un paramètre **B**, de tracer un rectangle, et même un rectangle plein, en précisant **BF**.

L'instruction **CIRCLE**, très puissante ici, permet de tracer des cercles de centre et de rayon désigné, mais on peut aussi lui faire tracer des arcs de cercle, ou des ovales...

L'instruction **PAINT** permet de colorier l'intérieur d'une figure fermée.

AREA (x,y) permet de mettre en place les points d'un contour polygonal fermé qui sera ensuite coloré avec **AREAFILL**.

PATTERN permet de modifier l'aspect d'une ligne, ou même d'une surface. en définissant un masque de ligne ou de surface de 16 bits.

LA SAISIE AU VOL D'UN CARACTERE FRAPPE AU CLAVIER : INKEY\$

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 l'AmigaBasic, cette propriété est offerte, par la **fonction INKEY\$** qui effectue une scrutation des touches du clavier et enregistre le nom de la dernière touche frappée.

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 (on n'a rien frappé) on attend, et l'on sort dès qu'une touche a été frappée.

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 ceci 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 (6-8) dessine la portée. La variable N\$, en ligne 9, contient les noms des notes et P\$ contient les notes sous forme codée.

La deuxième boucle I (10-18) permet de faire le choix d'une note de façon aléatoire, à la ligne 11 ; son nom est calculé en ligne 12 et la note est dessinée par les lignes 13 et 14.

La ligne 19 permet de recommencer en tapant une touche quelconque.

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 les touches CTRL et Break 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.

```

1  REM COMPOSE
2  SCREEN 1,640,200,4,2:WINDOW 1,"COMPOSE",,2,1
3  RANDOMIZE TIMER:DEFINT a-z
4  debut:
5  COLOR 3,1:CLS
6  FOR i=1 TO 5
7  LINE(0,18*i+34)-(600,18*i+34),0
8  NEXT i
9  n$="SI LA SOLFA MI RE DO ":p$="99388078470165958752
  3494440392349329293261247"
10 FOR i=1 TO 20
11  x=INT(RND*14+1)
12  d$=MID$(n$,3*(x+7*(x>7))-2,3):co=x+1-(x=6)
13  LINE(30*i-14,9*x+2)-(30*i-14,9*x+15),co
14  CIRCLE(30*i-21,9*x+15),7,co:PAINT(30*i-21,9*x+15),c
    o,co
15  COLOR 0:LOCATE 18,3*i-2:PRINT d$;
16  SOUND VAL(MID$(p$,3*x-2,3)),10,100
17  FOR j=1 TO 2100:NEXT j
18  NEXT i
19  LOCATE 20,1:PRINT"APPUYEZ SUR UNE TOUCHE";:r$=INPUT
    $(1)
20  GOTO debut

```

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 (6-11), une deuxième boucle J (9-11) étant utilisée pour contrôler qu'il n'y a pas deux tirages identiques.

Le tirage (ou le retraitage) est effectué en ligne 8.

La boucle I (15-18) affiche le résultat, et la ligne 19 affiche le numéro complémentaire.

Les lignes 20 et 21 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 N si l'on désire arrêter.

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  SCREEN 1,320,200,4,1:WINDOW 1,"LOTO",,2,1
3  RANDOMIZE TIMER:DEFINT a-z:DIM n(7)
4  WHILE r$<>"N"
5  COLOR 7,0:CLS
6  FOR i=1 TO 7
7  tirage:
8  n(i)=INT(RND*49+1)
9  FOR j=1 TO i-1
10 IF n(i)=n(j) THEN tirage
11 NEXT j,i
12 LINE (10,76)-(300,110),2,bf
13 LOCATE 4,8:PRINT D$;:COLOR 7:PRINT " TIRAGE DU LOTO
   ";;COLOR 12:PRINT D$
14 LINE (60,20)-(240,40),4,b
15 FOR i=1 TO 6
16 COLOR INT(RND*13+3),2:SOUND 12*n(i)+220,2
17 LOCATE 11,5*i-3:PRINT n(i)
18 NEXT i
19 LOCATE 17,1:COLOR 0,1:PRINT"NO. COMPLEMENTAIRE :";:
   COLOR 4:PRINT n(7)
20 LOCATE 19,1:COLOR 5,6:PRINT "UN AUTRE TIRAGE ( O/N
   )"
21 r$=UCASE$(INPUT$(1))
22 WEND

```

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, l'Amiga 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 4 et 5.

Le nombre est fabriqué à la ligne 7 (il pourra y avoir des chiffres répétés).

L'affichage est préparé par les lignes 8 à 11.

La ligne 12 s'occupe de la fin de partie.

Une première boucle I (16-22) teste les chiffres qui sont à la bonne place.

Une deuxième boucle J (18-20) teste les chiffres qui sont présents et mal placés.

Les compteurs sont aux lignes 17 et 19.

Les lignes 23 et 24 affichent le résultat, et la ligne 25 comptabilise les essais.

POUR JOUER :

On tape RUN et l'on choisit la longueur du nombre désiré (de 1 à 6), suivie de RETURN.

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

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 CTRL et Break.

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 SCREEN 1,320,200,4,1:WINDOW 1,"MASTERMIND",,2,1
3 RANDOMIZE TIMER:DEFINT a-z:CLS
4 INPUT"COMBIEN DE CHIFFRES :";n
5 IF n>6 THEN RUN ELSE CLS
6 e=1:f=1:f$=SPACE$(8)
7 x$=STR$(INT(RND*9^n+10^(n-1))):x$=RIGHT$(x$,LEN(x$)-1)
8 LINE (0,0)-(320,30),15,bf:LINE (0,30)-(320,30),9
9 LINE (45,30)-(45,200),9:LINE (130,30)-(130,200),9
10 LINE (220,30)-(220,200),9:COLOR 8,15
11 LOCATE 2,1:PRINT"NO.    NOMBRE    BON    PLACE":COL
   LOR 2,0
12 WHILE e$<>x$
13 COLOR 3:LOCATE 4+f,1:PRINT e;
14 COLOR 2,0:LOCATE 4+f,6:PRINT SPACE$(7);
15 x=0:y=x:LOCATE 4+f,6:INPUT "",e$
16 FOR i=1 TO n
17 IF MID$(x$,i,1)=MID$(e$,i,1) THEN x=x+1:GOTO suite
18 FOR j=1 TO n
19 IF j<>i AND MID$(x$,i,1)=MID$(e$,j,1) THEN y=y+1:GO
   TO suite
20 NEXT j
21 suite:
22 NEXT i
23 COLOR 11:LOCATE 4+f,17:PRINT x;
24 COLOR 14:LOCATE 4+f,27:PRINT y;:SOUND 250,3
25 e=e+1:f=f+1:IF f=16 THEN f=1
26 WEND
27 FOR i=200 TO 1000 STEP 10:SOUND i,1:NEXT i
28 PRINT:PRINT"TROUVE EN ";e;" COUPS":END

```

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 6 permet de confectionner l'entier.

La boucle I (12-17) calcule les trois restes successifs, et la boucle emboîtée J (15-17) permet de les afficher sous une forme assez visuelle, aux bons emplacements.

La ligne 13 indique de quelle division il s'agit (en encadrant le diviseur).

Le nombre proposé par le joueur est entré à la ligne 18 ; il est affiché et testé en ligne 19 (affichage de fin de partie : BRAVO ou PERDU).

Les lignes 20 et 21 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 RETURN.

On obtiendra, en réponse, un affichage PERDU ou BRAVO, puis la question LA MEME, UNE AUTRE ou STOP (M/A/S).

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

EXTENSIONS POSSIBLES :

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


```

1  REM NICOMAUQUE
2  SCREEN 1,320,200,4,1:WINDOW 1,"NICOMAUQUE",,2,1
3  RANDOMIZE TIMER
4  debut:
5  COLOR 8,15:CLS
6  x=INT(RND*100+1)
7  retour:
8  PRINT"RESTE DANS LA DIVISION PAR"
9  FOR i=3 TO 7 STEP 2
10 LOCATE 7,5*i-10:COLOR i+1:PRINT CHR$(48+i)
11 NEXT i
12 FOR i=1 TO 3
13 LINE (100*i-76,46)-(100*i-32,74),2*i+2,b
14 a=x-(2*i+1)*INT(x/(2*i+1))
15 FOR j=1 TO a
16 COLOR 2:LOCATE 5,j+10*i-8:IF a<>0 THEN PRINT "*"
17 NEXT j,i
18 COLOR 5:LOCATE 11,1:INPUT"VOTRE ESSAI :";n
19 COLOR 4:PRINT:IF n=x THEN PRINT"BRAVO" ELSE PRINT "
  PERDU"
20 PRINT:PRINT"LA MEME, UNE AUTRE OU STOP":r$=UCASE$(I
  NPUT$(1))
21 ON INSTR(1,"MAS",r$) GOTO retour,debut,fin
22 fin:
23 FOR i=200 TO 1200 STEP 10:SOUND i,1:NEXT i

```

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

La boucle I (ligne 3) lit les DATA (lignes 19 et 20).

La ligne 4 calcule la position initiale.

La boucle WHILE-WEND (lignes 5 à 17) permet d'opérer tant que l'angle de rotation n'est pas égal à 0.

Les lignes 6 et 7 dessinent les axes.

La boucle I (ligne 9) dessine le cube.

Les entrées sont faites aux lignes 11 à 14, et gérées par les lignes 15 et 16, 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  CLS:DIM x(20),y(20),z(20)
3  FOR i=0 TO 15:READ a,b,c:x(i)=a-100:y(i)=b-100:z(i)
   =c-50:NEXT i
4  a=30:GOSUB angle:GOSUB rotz:a=10:GOSUB angle:GOSUB
   roty
5  WHILE a<>0
6  CLS:LINE(10,170)-(40,120):LINE-(120,120):LINE (40,1
   20)-(40,80)
7  LOCATE 10,5:PRINT "z":LOCATE 15,5:PRINT "0":LOCATE
   15,14:PRINT "y":LOCATE 19,1:PRINT "x";
8  x=400+y(0):y=100-z(0)/2
9  FOR i=1 TO 15:u=400+y(i):v=100-z(i)/2:LINE(x,y)-(u,
   v):x=u:y=v:NEXT i
10 entree:
11 LOCATE 19,6:PRINT "ROTATION AUTOUR DES X, Y, ou Z :
   "':r$=INPUT$(1)
12 IF r$>="a" THEN r$=CHR$(ASC(r$)-32)
13 IF r$<>"X" AND r$<>"Y" AND r$<>"Z" THEN GOTO entree
   ELSE PRINT r$:f=ASC(r$)-87
14 LOCATE 20,6:PRINT "QUEL ANGLE :":INPUT a$
15 a=VAL(a$):GOSUB angle
16 ON f GOSUB rotx,roty,rotz
17 WEND
18 PRINT "C'est fini....":END
19 DATA 0,100,100,0,0,100,100,0,100,100,100,100,0,100,
   100,0,100,0,100,100,0,100
20 DATA 0 ,0,0,0,0,0,100,0,100,100,0,100,100,100,0
   ,100,100,0,0,0,0,0,0,100
21 angle:
22 a=a*3.1416/180:c=COS(a):s=SIN(a):RETURN
23 rotx:
24 FOR i=0 TO 15:b=y(i):d=z(i):y(i)=b*c-d*s:z(i)=b*s+d
   *c:NEXT i:RETURN
25 roty:
26 FOR i=0 TO 15:a=x(i):d=z(i):z(i)=d*c-a*s:x(i)=d*s+a
   *c:NEXT i:RETURN
27 rotz:
28 FOR i=0 TO 15:a=x(i):b=y(i):x(i)=a*c-b*s:y(i)=a*s+b
   *c:NEXT i:RETURN

```

Von Koch

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : dessin.

Difficulté du jeu : facile.

LE JEU :

Il s'inspire d'une définition de courbe fractale, proposée par le mathématicien Von Koch : un segment de droite est divisé en 3 parties égales ; sur la partie du milieu, on construit une pointe formée de 2 segments de même longueur que les précédents.

On peut ensuite recommencer l'opération sur chacun des 4 segments obtenus, et ceci indéfiniment. On obtient ainsi une courbe très déchiquetée, dite courbe fractale, dont la longueur peut être aussi grande que l'on veut.

LE PROGRAMME :

En ligne 3, trois tables X, Y et Z sont prévues pour enregistrer les données de chaque dessin, la table V devant contenir les emplacements des points de chaque segment après subdivision. La courbe à fractaliser est tirée au hasard.

Le niveau de fractalisation est entré en ligne 6. La ligne 7 gère la fin de partie.

Les données sont lues aux lignes 8 à 12.

La boucle I (13 à 24) gère les dessins des segments, la fractalisation étant effectuée par la boucle J (15-20) et la représentation graphique étant effectuée par une autre boucle J (21-23).

Les lignes 25 et 26 gèrent la fractalisation suivante.

POUR JOUER :

On lance le programme par RUN. Puis on choisit le niveau de fractalisation (de 0 à 5). La courbe est alors dessinée (ce peut être assez long avec un niveau de fractalisation élevé).

On tape F pour sortir du programme.

EXTENSIONS POSSIBLES :

On pourra définir d'autres courbes, d'autres méthodes de fractalisation.

On peut améliorer la précision du tracé.

```

1  REM VON KOCH
2  SCREEN 1,640,400,2,4:WINDOW 1,"VON KOCH",,2,1:RANDO
   MIZE TIMER:CLS
3  DIM v(2000),x(30),y(30),z(30):x=INT(RND*3+1)
4  debut:
5  LOCATE 1,1:PRINT "NIVEAU DE FRACTALISATION ( F pour
   Fin ) :      "
6  r$=UCASE$(INPUT$(1)):f=VAL(r$)
7  LOCATE 1,43:PRINT f:IF r$="F" THEN SOUND 500,3:END
8  RESTORE triangle:IF x=2 THEN RESTORE france ELSE IF
   x=3 THEN RESTORE maison
9  READ n,a,b,c:PSET (a,b):a=0
10 FOR i=1 TO n
11 READ x(i),y(i)
12 NEXT i
13 FOR i=1 TO n
14 z(i)=y(i)/(3^f):v(1)=x(i):v(2)=x(i)-60:v(3)=x(i)+60
   :v(4)=x(i)
15 FOR j=5 TO 4^f
16 IF j MOD 4=1 THEN v(j)=v(INT(j/4)+1)
17 IF j MOD 4=2 THEN v(j)=v(j-1)-60
18 IF j MOD 4=3 THEN v(j)=v(j-2)+60
19 IF j MOD 4=0 THEN v(j)=v(j-3)
20 NEXT j
21 FOR j=1 TO 4^f
22 a=6.2832*((v(j)+c) MOD 360)/360:b=z(i):LINE-STEP(-b
   *COS(a),b*SIN(a))
23 NEXT j
24 NEXT i
25 LOCATE 2,1:PRINT "APPUYEZ UNE TOUCHE":r$=UCASE$(INP
   UT$(1))
26 IF r$<>"F" THEN CLS:GOTO debut ELSE SOUND 1000,4:EN
   D
27 triangle:
28 DATA 3,400,170,0,0,150,120,150,240,150
29 france:
30 DATA 18,200,120,180,45,10,135,10,0,20,30,20,135,20,
   45,5,315,25,30,50
31 DATA 330,50,290,50,260,60,170,50,200,25,250,25,170,
   40,80,55,100,45,180,20
32 maison:
33 DATA 14,250,160,180,45,25,0,75,90,15,0,15,270,15,0,
   50,315,25,270,100
34 DATA 180,75,90,45,180,25,270,45,180,75,90,100

```

A travers

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion (2 joueurs).

Difficulté du jeu : moyenne.

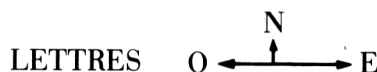
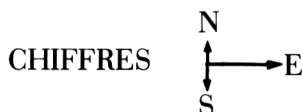
LE JEU :

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 (5, 5) ; les lignes 6 à 9 définissent les valeurs de départ.

Les lignes 13 à 22 dessinent le terrain de jeu et les lignes 25 et 26 affichent le joueur qui a le trait.

Le changement de joueur est assuré par la ligne 44.

La boucle I (27-29) contrôle les réponses, et les boucles I et J (31 à 34) contrôlent le coup.

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 suivi de RETURN.

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 obtient l'affichage correspondant au deuxième joueur.

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

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 l'Amiga.

```

1  REM A TRAVERS
2  SCREEN 1,320,210,4,1:WINDOW 1,"A TRAVERS",,2,1
3  RANDOMIZE TIMER:DEFINT a-z
4  DIM n(5,5):k=INT(RND*2):a$(0)="JEU DES "
5  a$(1)=a$(0)+"CHIFFRES":a$(0)=a$(0)+"LETTRES"
6  FOR i=1 TO 4
7    n(i,1)=ASC("0")+i
8    n(4,i)=ASC("?")+i
9  NEXT i
10 n(4,1)=0
11 dessin:
12 CLS
13 FOR u=1 TO 4
14   FOR v=1 TO 4
15    LINE (40*v-10,36*u-30)-(40*v+20,36*u-6),11,b
16    LOCATE 4*v-1,4*u+1
17    IF n(v,u)=0 THEN
18     COLOR 5:PRINT "."
19    ELSE
20     COLOR-3*(ASC(CHR$(n(v,u)))>62)+1:PRINT CHR$(n(v,u))
21    END IF
22   NEXT v,u
23  debut:
24  SOUND 200+500*k,3
25  COLOR 3*k+3:LOCATE 17,1:PRINT a$(k)
26  COLOR 11:LOCATE 19,1:INPUT"VOTRE JEU, AVEC N,S,E,O
27  :":r$:r$=UCASE$(r$)
28  FOR i=1 TO LEN(r$)
29   IF MID$(r$,i,1)>="a" THEN MID$(r$,i,1)=CHR$(ASC(MID
30   $(r$,i,1))-32)
31  NEXT i
32  IF (k=1 AND RIGHT$(r$,1)="O") OR (k=0 AND RIGHT$(r$
33  ,1)="S") THEN debut
34  FOR i=1 TO 4
35   FOR j=1 TO 4
36    IF n(i,j)=ASC(LEFT$(r$,1)) THEN ecrit
37  NEXT j,i
38  GOTO debut
39  ecrit:
40  a=i:b=j
41  i=i-(RIGHT$(r$,1)="S")+(RIGHT$(r$,1)="N")
42  j=j-(RIGHT$(r$,1)="E")+(RIGHT$(r$,1)="O")
43  IF (i=0 AND k=0) OR (j>4 AND k=1) THEN change
44  IF n(i,j)<>0 THEN debut
45  n(i,j)=ASC(LEFT$(r$,1))
46  change:
47  n(a,b)=0:k=ABS(k-1)
48  GOTO dessin

```

Visions

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin.

Difficulté du jeu : facile.

LE JEU :

L'Amiga 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 (6-15) se charge de réaliser les dessins.

La ligne 7 permet de transformer en nombre chaque lettre du mot entré au début (ligne 5). Ce nombre sera ensuite traduit en base 2 par une boucle J (lignes 8 à 14) ; selon le résultat de cette transformation, un cercle sera dessiné, ou non (ligne 12).

POUR JOUER :

On débute l'exécution du programme par RUN, ce qui amène l'affichage de la question QUEL DESSIN. On répond en entrant un mot quelconque qui sera transformé en dessin par l'Amiga dès que l'on aura frappé RETURN.

Pour recommencer, il suffira de frapper une touche quelconque, et CTRL et Break pour finir.

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 l'Amiga deviendra de plus en plus savant. On peut aussi réaliser un jeu à deux joueurs dans lequel l'Amiga vous demande de trouver le mot qui correspond à un certain dessin, etc.


```

1  REM VISIONS
2  SCREEN 1,320,200,5,1:WINDOW 1,"VISIONS",,2,1
3  debut:
4  RANDOMIZE TIMER:COLOR 1,2:CLS
5  INPUT"QUEL DESSIN :";r$:r$=UCASE$(r$)
6  FOR i=1 TO LEN(r$)
7    n=9*ASC(MID$(r$,i,1))-340
8    j=1
9    WHILE j<10
10     s=n/2-INT(n/2)
11     n=INT(n/2):SOUND INT(RND*1000+200),2
12     IF s<>0 THEN CIRCLE(12*j+90,10*i+30),6,0:PAINT(12*j
        +90,10*i+30),INT(RND*31+1),0
13     j=j+1
14   WEND
15 NEXT i
16 LOCATE 20,1:PRINT"APPUYER SUR UNE TOUCHE";
17 r$=INPUT$(1):GOTO debut

```

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 PSET 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 trois premiers paramètres Z, K et T, introduits en ligne 6, définissent le rayon maximum et l'occupation de l'écran.

Les variables aléatoires A et B, choisies entre 0 et 3, de la ligne 10, définissent l'allure de la courbe ; il pourra être intéressant de les introduire manuellement (par exemple : INPUT A, B).

La boucle H (11-16) réalise le tracé, point par point. Le rayon R est calculé en ligne 12, et le passage en coordonnées cartésiennes est effectué aux lignes 13 et 14.

L'affichage des points est réalisé en ligne 15.

La ligne 17 imprime un message de fin.

La ligne 20 permet de refaire un nouveau dessin, alors que la ligne 21 assure un nouveau tracé sans effacement du précédent.

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 0.

Si on veut un tracé sans effacement du dessin précédent, on tape une touche quelconque.

Pour arrêter, faire CTRL et Break.

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  SCREEN 1,600,380,4,2
3  WINDOW 1,"MATHIEU",,2,1
4  RANDOMIZE TIMER:f=2:COLOR ,f:CLS
5  WHILE a$<>" "
6  z=9:k=10:l=INT(RND*8+1)+12:t=3:c=f
7  WHILE c=f
8  c=INT(RND*16)
9  WEND
10 a=3*RND:b=3*RND
11 FOR h=0 TO 6.28 STEP 3.14/180
12 r=z*SIN(h*t):SOUND INT(RND*1800+200),.2
13 x=300+INT(l*r*COS(a*h))
14 y=100+INT(k*r*SIN(b*h))
15 PSET (x,y),c
16 NEXT h
17 LOCATE 20,1:PRINT"C'EST FINI...";
18 FOR i=700 TO 150 STEP-10:SOUND i,1:NEXT i
19 a$=INPUT$(1)
20 IF a$="Q" THEN f=INT(RND*16):COLOR ,f:CLS
21 WEND

```

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 3 contient les positions de départ.

La boucle I (6-10) dessine le damier, et les lignes 12 à 15 placent le loup et les agneaux.

La ligne 16 gère le changement de joueur.

La ligne 17 affiche JEU DU LOUP si c'est le joueur qui manipule le loup qui a le trait, et la ligne 18 JEU DES AGNEAUX dans le cas contraire.

Les lignes 19 à 22 ou 24 à 28 analysent la réponse, en fonction du joueur concerné.

POUR JOUER :

RUN déclenche le tirage au sort du joueur qui commence. Après quoi, le damier s'affiche et la phrase JEU DU LOUP (A, S, Z, 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 dépassement des limites du terrain, 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 à l'Amiga 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 l'Amiga).

```
1 REM LE LOUP ET LES AGNEAUX
2 SCREEN 1,320,200,4,1:WINDOW 1,"LE LOUP ET LES AGNEAUX",,2,1
3 COLOR 9,15:CLS:DEFINT a-z:a$="8183858714":b$=" "
4 RANDOMIZE TIMER:s=INT(RND*2+1)
5 PRINT"  A B C D E F G H"
6 FOR i=1 TO 8
7 LOCATE 2*i+1,1:PRINT CHR$(48+i)
8 FOR j=1 TO 8
9 LINE(30*i-10,18*j-6)-(30*i+20,18*j+12),3*((i+j) MOD 2)+3,bf
10 NEXT j,i
11 affiche:
12 l=2*VAL(MID$(a$,9,1))+1:k=3*VAL(MID$(a$,10,1))+1:COLOR 4,6:LOCATE 1,k:PRINT "L"
13 FOR i=1 TO 4
14 l=2*VAL(MID$(a$,2*i-1,1))+1:k=3*VAL(MID$(a$,2*i,1))+1:COLOR 8:LOCATE 1,k:PRINT CHR$(48+i)
15 NEXT i
16 s=ABS(s-1):COLOR 0,7:LOCATE 19,1:PRINT SPACE$(30)
17 LOCATE 19,1:IF s=0 THEN INPUT"JEU DU LOUP (Q,S,W,X):";r$
18 IF s=1 THEN INPUT"JEU DES AGNEAUX (ND OU NG):";r$:GOTO agneaux
19 IF r$>="a" THEN r$=CHR$(ASC(r$)-32)
20 c=10:COLOR ,6:LOCATE 2*VAL(MID$(a$,9,1))+1,3*VAL(MID$(a$,10,1))+1:PRINT b$
21 a=VAL(RIGHT$(a$,2))+(c+1 AND r$="X")+(c-1 AND r$="W")
22 a=a-(c+1 AND r$="Q")-(c-1 AND r$="S")
23 GOTO suite
24 agneaux:
25 IF RIGHT$(r$,1)>="a" THEN MID$(r$,2,1)=CHR$(ASC(RIGHT$(r$,1))-32)
26 c=2*VAL(MID$(r$,1,1)):l$=MID$(a$,c-1,1):k$=MID$(a$,c,1):a=VAL(l$+k$)
27 COLOR ,6:LOCATE 2*VAL(l$)+1,3*VAL(k$)+1:PRINT b$
28 a=a-(9 AND RIGHT$(r$,1)="D")-(11 AND RIGHT$(r$,1)="G")
29 suite:
30 a$=LEFT$(a$,c-2)+RIGHT$(STR$(a),2)+RIGHT$(a$,LEN(a$)-c)
31 GOTO affiche
```

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 4 à 6 préparent l'affichage de la locomotive et le wagon.

La position initiale de la locomotive est définie par la ligne 7, celle du wagon par la ligne 9.

La ligne 10 évite qu'ils soient initialement trop proches l'un de l'autre.

Les lignes 13 à 15 réalisent l'affichage du wagon et celui de la locomotive.

Le test de fin est en ligne 16.

Les mouvements à effectuer sont entrés dans une variable alphanumérique R\$, en ligne 17, et sont décodés dans une boucle I (19-22) pour être exécutés en ligne 23.

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 RETURN.

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.

EXTENSIONS POSSIBLES :

Il sera intéressant de prévoir la perte de la partie si, par accident, au cours du déplacement, la locomotive vient en collision avec 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  SCREEN 1,640,200,4,2:WINDOW 1,"ACCROCHEZ LES WAGONS
   ",,2,1
3  RANDOMIZE TIMER:DEFINT a-z
4  DIM l(20,2),w(20,2):u=30:v=120:w=56
5  FOR i=1 TO 20:READ a,b:l(i,1)=a:l(i,2)=b:NEXT i
6  FOR i=1 TO 19:READ a,b:w(i,1)=a:w(i,2)=b:NEXT i
7  n=0:ly=4*INT(RND*u)+u:lx=4*INT(RND*v)+u
8  tirwagon:
9  wy=4*INT(RND*u)+u:wx=4*INT(RND*v)+u
10 IF ABS(lx-wx)<u+15 THEN tirwagon
11 debut:
12 COLOR 2,15:CLS
13 FOR i=1 TO 19:LINE(l(i,1)+lx,ly-l(i,2))-(l(i+1,1)+l
   x,ly-l(i+1,2)),9:NEXT i
14 FOR i=1 TO 18:LINE(w(i,1)+wx,wy-w(i,2))-(w(i+1,1)+w
   x,wy-w(i+1,2)),4:NEXT i
15 PAINT (lx+3,ly-12),9,9:PAINT (wx+3,wy-12),4,4
16 IF lx=wx+w AND wy=ly THEN fin
17 LOCATE 20,1:INPUT"CHEMIN A SUIVRE (DGGHBB...):";r$
18 n=n+1:x=0:y=0:r$=UCASE$(r$)
19 FOR i=1 TO LEN(r$)
20 x=x-4*(MID$(r$,i,1)="D")+4*(MID$(r$,i,1)="G")
21 y=y-4*(MID$(r$,i,1)="B")+4*(MID$(r$,i,1)="H")
22 NEXT i
23 lx=lx+x:ly=ly+y
24 IF lx<0 OR lx>575 THEN lx=lx-x
25 IF ly<0 OR ly>185 THEN ly=ly-y
26 GOTO debut
27 fin:
28 LOCATE 20,1:PRINT"BRAVO ; TROUVE EN ";n;" COUPS";
29 FOR i=200 TO 1000 STEP 20:SOUND i,2:NEXT i:END
30 DATA 0,10,0,30,10,30,10,27,30,27,30,40,40,40,40,27,
   50,27
31 DATA 50,17,55,10,45,10,42,0,38,0,35,10,25,10,22,0,1
   8,0,15,10,0,10
32 DATA 0,10,0,30,5,30,5,15,45,15,45,30,50,30,50,12,55
   ,12
33 DATA 55,10,45,10,42,0,38,0,35,10,25,10,22,0,18,0,15
   ,10,0,10

```

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 affichera 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 4 à 8) ; des données, de la ligne 30 à la ligne 45, contiennent tous les éléments du dessin.

La boucle I (9) fabrique un mot P\$ qui contient des tirets à l'emplacement de chacune des lettres du mot initial.

La boucle I (15-19) s'occupe du dessin (plus ou moins complet, selon la valeur de P).

Les lignes 12 à 14 affichent le résultat et arrêtent le jeu.

La lettre essayée est entrée en ligne 20 et testée dans une boucle I (24-27) ; la ligne 26 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 RETURN. 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 suivie de RETURN.

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.

EXTENSIONS POSSIBLES :

Il sera intéressant de constituer une liste de mots dans lesquels l'Amiga 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 SCREEN 1,640,200,4,2:WINDOW 1,"PENDU",,2,1
3 r$="":t$="":p$="":k=1:COLOR 3,1:CLS
4 WHILE s$(<>CHR$(13))
5 r$=r$+UCASE$(s$):IF s$="0" THEN RUN
6 LOCATE 1,1:PRINT "VOTRE MOT :";t$;
7 s$=INPUT$(1):t$=t$+"#"
8 WEND
9 FOR i=1 TO LEN(r$):p$=p$+"-":NEXT i
10 p=12:CLS
11 debut:
12 COLOR 0:LOCATE 4,1:PRINT p$:RESTORE
13 IF p=1 THEN PRINT"PERDU ; LE MOT ETAIT :";r$:END
14 IF p=r$ THEN PRINT"BRAVO":END
15 FOR i=11 TO p STEP -1
16 READ n
17 FOR j=1 TO n
18 READ a,b,c,d:LINE(a+300,b-50)-(c+300,d-50),2
19 NEXT j,i
20 COLOR 2:LOCATE 14,1:INPUT"VOTRE LETTRE :";s$
21 t=0:s$=UCASE$(s$)
22 COLOR 0:LOCATE 16,1:PRINT "LETTE ESSAYEE"
23 COLOR 3:LOCATE 17,k:PRINT s$;" ";:k=k+1
24 FOR i=1 TO LEN(r$)
25 SOUND INT(RND*4000+500),1
26 IF MID$(r$,i,1)=s$ THEN t=1:MID$(p$,i,1)=s$
27 NEXT i
28 IF t=0 THEN p=p-1
29 GOTO debut
30 DATA 4,50,60,50,50,50,50,150,50,150,50,150,60,150,6
0,50,60
31 DATA 4,95,80,95,60,95,60,105,60,105,60,105,80,105,8
0,95,80
32 DATA 4,80,90,95,80,95,80,105,80,105,80,120,90,120,9
0,80,90
33 DATA 6,90,90,80,100,80,100,90,110,90,110,110,110,11
0,110,120,100,120,100
34 DATA 110,90,110,90,90,90
35 DATA 4,95,120,95,110,95,110,105,110,105,110,105,120
,105,120,95,120
36 DATA 10,50,135,60,120,60,120,140,120,140,120,150,13
5,150,135,140,135
37 DATA 140,135,130,125,130,125,130,135,130,135,70,135
,70,135,70,125,70,125
38 DATA 60,135,60,135,50,135
39 DATA 4,70,135,130,135,130,135,130,155,130,155,70,15
5,70,155,70,135
40 DATA 7,70,155,70,165,70,165,90,165,90,165,90,155,90
,155,110,155
41 DATA 110,155,110,165,110,165,130,165,130,165,130,15
5
42 DATA 6,70,165,70,185,70,185,90,185,90,185,90,165,11
0,165,110,185
43 DATA 110,185,130,185,130,185,130,165
44 DATA 8,50,185,50,200,50,200,90,200,90,200,90,185,90
,185,50,185
45 DATA 110,185,110,200,110,200,150,200,150,200,150,18
5,150,185,110,185
```

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 changement de joueur est effectué par la ligne 6.

Le joueur qui a le trait voit le symbole qu'il a choisi (x ou o) se placer en limite du cadre (ligne 9).

Les lignes 11 et 12 permettent aux deux joueurs de déplacer le symbole sur le terrain.

Les lignes 13 à 15 contrôlent si le déplacement est possible (case libre et non dépassement des limites).

La ligne 9 permet d'écrire sur une case libre.

POUR JOUER :

RUN provoque l'affichage des limites du terrain et du numéro du joueur, ainsi que des symboles correspondants que l'on devra déplacer sur l'écran à l'aide des quatre touches (2, 4, 6, 8) ou des touches (H, B, G, D).

Il n'est pas possible de venir sur une case déjà occupée et il faudra appuyer sur la barre d'espace pour inscrire son propre symbole dans une case disponible.

Dès que cela a été fait, le symbole du second joueur apparaît dans la colonne de gauche, à peu près au milieu, et le joueur procédera de la même manière que son adversaire.

Pour sortir du programme, il suffira de taper CTRL et Break.

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  SCREEN 1,640,200,4,2:WINDOW 1,"PLUSIEURS",,2,1
3  RANDOMIZE TIMER:COLOR ,11:CLS:j=INT(RND*2)
4  LINE (0,9)-(630,180),2,bf:DIM t(80,20)
5  debut:
6  j=ABS(j-1):LOCATE 1,26:COLOR j+3:PRINT "JOUEUR ";CHR$(88+9*(j=1))
7  x=12:y=1
8  affiche:
9  LOCATE x,y:COLOR j+3,2:PRINT CHR$(88+9*(j=1));:t(y,x)=1
10 a$=UCASE$(INKEY$):a=x:b=y
11 x=x-((a$="2" OR a$="B" ) AND x<20)+((a$="8" OR a$="H") AND x>2)
12 y=y-((a$="6" OR a$="D") AND y<62)+((a$="4" OR a$="G") AND y>2)
13 IF a$=" " AND y<>1 THEN pose ELSE c=t(y,x)
14 IF c<>0 AND a$<>" " THEN SOUND 2000,3
15 IF c<>0 THEN x=a:y=b ELSE LOCATE a,b:PRINT " ";:t(b,a)=0
16 GOTO affiche
17 pose:
18 LOCATE x,y:COLOR j+5:PRINT CHR$(88+9*(j=1));
19 IF j=0 THEN SOUND 700,3 ELSE SOUND 300,3
20 GOTO debut

```

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 1 (7) affiche le terrain, en appelant un sous-programme de dessin (27-30).

La ligne 9 définit la position initiale du sous-marin ; la ligne 11 permet son déplacement, contrôlé par la ligne 12 (on reste dans les limites du terrain).

La ligne 15 affiche votre position.

Les lignes 16 à 18 contrôlent votre déplacement.

La distance est calculée en ligne 19, affichée en ligne 20 et testée en ligne 21.

Le test de fin de partie est effectué à la ligne 8.

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 touches 2, 4, 6, 8 ou H, B, G, D.

Après chaque déplacement, la nouvelle distance du sous-marin s'affiche ; à chaque coup 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 utilisant les touches CTRL et Break.

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  SCREEN 1,320,200,4,1:WINDOW 1,"CHASSE",,15,1
3  RANDOMIZE TIMER:CLS:DEFINT a-z
4  LOCATE 1,1:INPUT "DIMENSION DE LA GRILLE : ",h
5  IF h<1 OR h<>INT(h) OR h>10 THEN RUN ELSE CLS
6  a=156-8*h:b=92-8*h:c=1
7  FOR i=0 TO h*h-1:j=i MOD h:k=INT(i/h):GOSUB grille:
   NEXT i:i=1
8  WHILE d<>INT(RND*3+2)
9  v=INT(RND*h):u=INT(RND*h)
10 tirage:
11 u=u+i:v=v+i:s=s+i
12 IF (u<0 OR u>h OR v<0 OR v>h) THEN i=-i
13 deplace:
14 IF x<>k OR y<>j THEN c=1:GOSUB grille:SOUND 300,4
15 k=x:j=y:c=3:GOSUB grille
16 r$=UCASE$(INKEY$):IF r$="" THEN deplace
17 x=x+((r$="8" OR r$="H") AND x>0)-((r$="2" OR r$="B"
   ) AND x<h-1)
18 y=y+((r$="4" OR r$="G") AND y>0)-((r$="6" OR r$="M"
   ) AND y<h-1)
19 d=INT(SQR((ABS(x-v))^2+(ABS(y-u)^2)))
20 LOCATE 1,1:PRINT "DIST. ";STR$(d)+" "
21 IF d<>0 THEN IF d<3 THEN tirage ELSE deplace
22 LOCATE 3,1:PRINT "PROF. ":d$=INPUT$(1)
23 d=VAL(d$):LOCATE 3,1:PRINT SPACE$(7)
24 WEND
25 LOCATE 5,1:PRINT "BRAVO"
26 FOR i=800 TO 200 STEP-20:SOUND i,2:NEXT i:END
27 grille:
28 LINE(a+16*j,b+16*k)-(a+16*j+16,b+16*k+16),,b
29 LINE(a+16*j+1,b+16*k+1)-(a+16*j+14,b+16*k+14),c,BF
30 RETURN

```

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 :

La boucle I (6-9) dessine les verres.

Les deux boucles emboîtées I (19-23) et J (20-23) les remplissent avec la quantité indiquée en ligne 18.

La variation de taille du lézard est définie par la ligne 28, avec contrôle par les lignes 29 et 30.

La boucle I (31-33) dessine le corps du lézard, les lignes 34 à 39 dessinent l'œil et les pattes.

Un message gagnant sera affiché par la ligne 41 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 RETURN.

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.

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  SCREEN 1,320,210,4,1:WINDOW 1,"LE LEZARD D'ALICE",,
   2,1
3  RANDOMIZE TIMER:DEFINT i,j
4  debut:
5  COLOR 2,1:CLS:n=18:C$=" ":COLOR ,9
6  FOR i=2 TO 10
7  LOCATE i+4,4:PRINT C$:LOCATE i+4,11:PRINT C$
8  LOCATE i+4,14:PRINT C$:LOCATE i+4,21:PRINT C$
9  NEXT i
10 FOR i=4 TO 11:LOCATE 15,i:PRINT C$:NEXT i
11 FOR i=14 TO 21:LOCATE 15,i:PRINT C$:NEXT i
12 COLOR 2,1:LOCATE 16,5:PRINT"GAUCHE":LOCATE 16,15:PR
   INT"DROIT"
13 LOCATE 18,1:PRINT "QUEL COTE :";
14 saisie:
15 a$=INKEY$:a$=UCASE$(a$):IF a$<>"D" AND a$<>"G" THEN
   saisie
16 u=2-20*(a$="D")
17 LOCATE 18,13:IF a$="D" THEN PRINT"DROIT" ELSE PRINT
   "GAUCHE"
18 PRINT:INPUT"QUELLE QUANTITE :";Q
19 FOR i=1 TO 4*SQR(Q)
20 SOUND 30*i+100,.5
21 FOR j=0 TO 59
22 PSET (5*u+j+30,126-i),6
23 NEXT j,i
24 a$="":SOUND 400,2
25 WHILE a$="":a$=INKEY$:WEND
26 COLOR 14,15:CLS
27 h=2*(RND<.5)-1
28 n=n+h*INT(SQR(Q))
29 IF n<12 THEN n=12
30 IF n>60 THEN n=60
31 FOR i=0 TO 4*(n+3)
32 PSET(i,100),1:PSET(i,99),1:PSET(i,98),1:PSET(i,97),
   1:SOUND 300+8*i,.5
33 NEXT i
34 t=4*(n-INT((SQR(n)/2)))
35 PSET(t,88),2:PSET(t-1,89),2:PSET(t-2,90),2:PSET(t+1
   ,89),2:PSET(t+2,90),2
36 t=4*SQR(n):PSET(t,102),2:PSET(t-1,103),2:PSET(t-2,1
   04),2
37 t=t+2:PSET(t,102),2:PSET(t+1,103),2:PSET(t+2,104),2
38 t=4*(n-SQR(n)):PSET(t,102),2:PSET(t-1,103),2:PSET(t
   -2,104),2
39 t=t+2:PSET(t,102),2:PSET(t+1,103),2:PSET(t+2,104),2
40 a$="":WHILE a$="":a$=INKEY$:WEND
41 IF n=28 THEN PRINT"BRAVO":SOUND 1200,4:END
42 GOTO debut

```

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 8 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 6 permet de contrôler s'il y a eu ou non action sur une touche de 1 à 8.

La boucle I (8-17) et les lignes 9 à 12 dessinent le cadre du dé.

La ligne 13 réalise le tirage, et la boucle J (14-17) dessine les points en relation avec le tirage réalisé (en appelant les variables A (J) et B (J), définies dans les sous-programmes 19 à 30).

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 8. Les dés demandés sont alors représentés, et la question NOMBRE DE DES est de nouveau posée. Il suffira de taper CTRL et Break pour sortir du programme.

EXTENSIONS POSSIBLES :

On pourra conserver la partie dessin utilisée dans ce programme pour réaliser toute sorte de jeux de dés, avec une représentation graphique de qualité convenable et facile à modifier si besoin était.


```

1 REM DESSIN DE DES
2 SCREEN 1,640,200,4,2:WINDOW 1,"DESSIN DE DES",,2,1
3 RANDOMIZE TIMER:DEFINT a-z
4 debut:
5 LOCATE 2,1:PRINT "NOMBRE DE DES ( 1 A 8 ): ";:r$=I
  NPUT$(1):n=VAL(r$)
6 IF n<=0 OR n<>INT(n) OR n>8 THEN RUN
7 CLS
8 FOR i=1 TO n
9 x1=70*i-66:y1=100:x2=70*i-6:y2=70:c=INT(RND*14+1):c
  =c+(c=9)
10 AREA(x1,y1):AREA(x2,y1):AREA(x2,y2):AREA(x1,y2):COL
  OR 10:AREAFILL
11 AREA(x1,y1):AREA(x1+8,y1+8):AREA(x2+8,y1+8):AREA(x2
  ,y1):COLOR 11:AREAFILL
12 AREA(x2+8,y1+8):AREA(x2+8,y2+8):AREA(x2,y2):AREA(x2
  ,y1):COLOR 11:AREAFILL
13 x=INT(RND*6+1):ON x GOSUB tir1,tir2,tir3,tir4,tir5,
  tir6
14 FOR j=1 TO x
15 x=a(j)+70*i-66:y=b(j)+54:SOUND INT(RND*1000+100),2
16 CIRCLE(x,y),5,4:PAINT(x,y),3,4
17 NEXT j,i
18 GOTO debut
19 tir1:
20 a(1)=30:b(1)=30:RETURN
21 tir2:
22 a(1)=10:a(2)=50:b(1)=20:b(2)=40:RETURN
23 tir3:
24 GOSUB tir2:a(3)=30:b(3)=30:RETURN
25 tir4:
26 GOSUB tir2:a(3)=10:b(3)=40:a(4)=50:b(4)=20:RETURN
27 tir5:
28 GOSUB tir4:a(5)=30:b(5)=30:RETURN
29 tir6:
30 GOSUB tir4:a(5)=10:b(5)=30:a(6)=50:b(6)=30: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 :

La ligne 5 contient le dessin des deux chapeaux (dans deux variables indicées C et D).

La ligne 4 va lire, en DATA (lignes 31 et 32), les symboles qui permettront de dessiner les personnages.

La ligne 7 permet, s'il y a lieu, l'échange des chapeaux.

La variable T de la ligne 7 sert à repérer s'il y a eu échange ou non.

Le dessin des deux personnages est réalisé par une boucle I (10-13), et leur déplacement est réalisé par la ligne 18.

Le test, permettant ou non le changement des couvre-chefs, est effectué en ligne 20.

Les lignes 23 à 27 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 1 et 2 et celui de droite, des touches 9 et 0.

S'il y a eu changement de chapeau, les touches à appuyer seront 1 (à gauche) et 0 (à droite).

S'il n'y a pas eu de changement, il faudra, pour gagner, être le premier à appuyer sur 2 (à gauche) et sur 9 (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  SCREEN 1,640,200,4,2:WINDOW 1,"CHAPEAUX",,2,1
3  RANDOMIZE TIMER:t=1:e=10:f=3:k=1:DIM u(26),v(26)
4  FOR i=1 TO 25 STEP 2:READ u(i),v(i),u(i+1),v(i+1):N
   EXT i
5  FOR i=1 TO 5:READ c(i),d(i):NEXT i
6  x=10:b=110:u(0)=0:v(0)=0:u(26)=0:v(26)=0:COLOR 8,11
7  IF RND<.5 THEN SWAP e,f:t=ABS(t-1)
8  dessin:
9  CLS:a=600-x
10 FOR i=1 TO 26
11 LINE (x+u(i-1),b-v(i-1))-(x+u(i),b-v(i))
12 LINE (a+u(i-1),b-v(i-1))-(a+u(i),b-v(i))
13 NEXT i
14 COLOR e:AREA(x+c(1),b-d(1)):AREA(x+c(2),b-d(2))
15 AREA(x+c(3),b-d(3)):AREA(x+c(4),b-d(4)):AREA(x+c(5)
   ,b-d(5)):AREAFILL
16 COLOR f:AREA(a+c(1),b-d(1)):AREA(a+c(2),b-d(2))
17 AREA(a+c(3),b-d(3)):AREA(a+c(4),b-d(4)):AREA(a+c(5)
   ,b-d(5)):AREAFILL
18 x=x+k*12:COLOR 8:SOUND 400,2:i=0:a$="":IF x>590 THE
   N k=-k
19 WHILE a$="" AND i<400:i=i+1:a$=INKEY$:WEND
20 IF ABS(x-a)<10 AND RND<.5 THEN t=ABS(t-1):SWAP e,f
21 IF a$<>"1" AND a$<>"2" AND a$<>"9" AND a$<>"0" AND
   a$<>" " GOTO dessin
22 IF x<320 GOTO perdu
23 IF (a$="1" AND t=1) OR (a$="2" AND t=0) THEN LOCATE
   15,1:PRINT "BRAVO GAUCHE":GOTO fin
24 IF (a$="0" AND t=1) OR (a$="9" AND t=0) THEN LOCATE
   15,1:PRINT "BRAVO DROIT":GOTO fin
25 perdu:
26 IF a$="1" OR a$="2" THEN LOCATE 15,1:PRINT "PERDU G
   AUICHE"
27 IF a$="9" OR a$="0" THEN LOCATE 15,1:PRINT "PERDU D
   ROIT"
28 fin:
29 LOCATE 17,1:PRINT "C'est fini...."
30 r$=INPUT$(1):END
31 DATA 0,0,0,7,7,7,13,55,10,65,3,40,0,40,10,70,13,70,
   12,77,15,85
32 DATA 18,77,17,70,20,70,30,40,27,40,20,65,17,55,23,7
   ,30,7
33 DATA 30,0,20,0,17,40,13,40,10,0,0,0
34 DATA 0,85,5,100,25,100,30,85,15,85

```

Labyrinthix

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 en ligne 6 (on pourra modifier les données pour en dessiner un autre, aux lignes 42 à 48).

Le dessin du labyrinthe est effectué par une boucle I (12-15).

La position du joueur est tirée par hasard et testée en ligne 19.

Le déplacement est entré en ligne 22.

La boucle I (23-29) contrôle si le déplacement est possible. Le résultat est affiché en ligne 30.

Les lignes 34 et 35 permettent d'entrer une solution, contrôlée dans les lignes 36 à 40.

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 CTRL et Break 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 LABYRINTHIX
2 SCREEN 1,640,200,4,2:WINDOW 1,"LABYRINTHIX",,2,1
3 WHILE ve<>1 AND ve<>2
4 LOCATE 2,1:PRINT "VERSION (1) OU (2) :";:ve=VAL(INP
  UT$(1))
5 WEND
6 RANDOMIZE TIMER:CLS:READ a:d=a:DIM l(a,a),des%(7):h
  =9:y=9:b$(1)="S"
7 des%(0)=&HFFFF:des%(1)=&H4444:des%(2)=&H4444:des%(3)
  )=&HFFFF
8 des%(4)=&H1111:des%(5)=&H1111:des%(6)=&HFFFF:des%(7)
  )=&H4444
9 PATTERN ,des%
10 FOR i=1 TO a-2:LOCATE 1,3*i+2:PRINT CHR$(64+i):NEXT
  i
11 WHILE a<>0
12 FOR i=1 TO a
13 READ b:x=30*b-30:l(1+((y-9)/h),b)=1
14 LINE (x,y)-(x+30,y+h),3,bf
15 NEXT i
16 READ a:y=y+h:c=64+(y-9)/h:IF c<75 THEN LOCATE c-62,
  62:PRINT CHR$(c)
17 WEND
18 tirage:
19 x=INT(RND*(c-65)+1):y=INT(RND*(d-1)+1):IF l(x,y)=1
  THEN tirage
20 a=x:b=y:t=0
21 debut:
22 s=0:LOCATE 15,1:COLOR 9:INPUT"DEPLACEMENT ( HHGHGGH
  B...):";r$:r$=UCASE$(r$)
23 FOR i=1 TO LEN(r$)
24 s$=MID$(r$,i,1)
25 x0=x:x=x+(s$="H")-(s$="B")
26 y0=y:y=y+(s$="G")-(s$="D")
27 IF l(x,y)<>1 THEN s=s+1 ELSE x=x0:y=y0:SOUND 100,3
28 SOUND 500+200*s,2
29 NEXT i
30 COLOR 14:LOCATE 16,1:PRINT"VOUS AVEZ ";s;" CASE";b$
  (->1));" LIBRE";b$(->1));" "
31 COLOR 11:PRINT"SOLUTION (O/N)":r$=UCASE$(INPUT$(1))
32 IF r$<>"O" THEN t=t+1:IF ve=2 THEN debut ELSE x=a:y
  =b:GOTO debut
33 IF ve=2 THEN a=x:b=y
34 COLOR 9:INPUT"LIGNE :";x$:x$=UCASE$(x$):u=ASC(x$)-6
  3
35 INPUT"COLONNE :";y$:y$=UCASE$(y$):v=ASC(y$)-63:COLO
  R 5
36 IF u=a AND v=b THEN
37 PRINT"BRAVO...";t;" ESSAIS"

```

```

38 ELSE
39 PRINT"PERDU . C'ETAIT LIGNE ";CHR$(63+a);", COLONNE
   ";CHR$(63+b)
40 END IF
41 FOR i=100 TO 1200 STEP 12:SOUND i,1:NEXT i:END
42 DATA 20,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,1
   8,19,20
43 DATA 4,1,5,13,20,15,1,3,5,7,8,9,10,11,12,13,15,16,1
   7,18,20
44 DATA 7,1,3,5,12,13,15,20,12,1,3,8,9,10,12,13,15,17,
   18,19,20
45 DATA 9,1,3,4,5,6,10,15,18,20,9,1,3,8,10,12,14,15,18
   ,20
46 DATA 9,1,3,5,7,8,10,12,18,20,9,1,5,7,12,14,16,17,18
   ,20
47 DATA 10,1,3,4,5,7,9,11,12,14,20,8,1,9,14,15,16,17,1
   8,20
48 DATA 20,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,1
   8,19,20,0

```

Niveau 4

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Smokey	Fumez sans risque, avec ce programme original pour un joueur.
Zombies	Un peu farfelu, mais très visuel (1 joueur).
Clavier	Apprenez à taper à la machine en vous amusant.
Calculatrice	Un programme de simulation très réaliste.
La bombe	Saurez-vous la découvrir à temps ? (1 joueur).
Adressse	Jeu de réflexe pour 1 joueur, très visuel.
Le jeu du missile	Une mission bien périlleuse, pour un seul joueur.
Horloge	Ne jetez pas votre vieille télévision : transformez-la en horloge.
Mots	Une idée très simple, mais ce n'est pas si facile de trouver.
Diago	Etes-vous vraiment sûr de vos réflexes ?
Décomaque	Pour vous éviter de sécher trop longtemps sur Nicomaque.
Affichage publicitaire	Si vous ne savez pas quoi mettre dans votre vitrine.
Tirage d'un dé truqué	Révissez vos connaissances en statistiques.
Le singe	Un jeu de lettres pour deux joueurs, où il faut de bons réflexes.
Lettres	Un jeu sur les mots, assez simple.
Mur de briques	Un grand classique dans une version très complète.
Œufs	Ne laissez pas faire la pondeuse diabolique, pour un joueur.
Course de voitures	Un jeu de réflexes pas trop difficile, pour un joueur.
Obstacles	Beaucoup d'obstacles à éviter dans ce jeu d'adresse, pour un joueur.
Echiquier	Une base graphique pour vos propres créations.
Cibles	Un peu d'adresse, beaucoup de chance : très visuel.

Structures de programmes Techniques d'animation Fenêtres et menus

STRUCTURES DE PROGRAMMES

La structure classique IF...THEN...ELSE... n'est pas utilisable lorsqu'il y a un ensemble complexe d'instructions à effectuer après le THEN ou le ELSE. AmigaBasic dispose d'une instruction beaucoup plus puissante ; la structure IF...END IF, d'utilisation très simple.

Exemple :

```
IF  condition THEN
    instruction
    instruction
    .
    .
ELSE
    instruction
    instruction
    .
    .
END IF
```

Il est possible également d'emboîter plusieurs structures de ce type, ce qui laisse une très grande souplesse au programmeur pour les tests.

- Les procédures

Outre les sous-programmes, appelés avec GOSUB et terminés avec RETURN, AmigaBasic propose une structure beaucoup plus autonome ; les procédures avec passage de paramètres. Contrairement au sous-programme, la procédure ne fait pas partie du programme principal et peut être placée n'importe où dans le programme. Le programme principal ne pourra pas y accéder en dehors des appels régulièrement prévus.

De plus, les variables utilisées dans la procédure sont autonomes et, sauf indication contraire, ne sont pas transmises au programme appelant. On

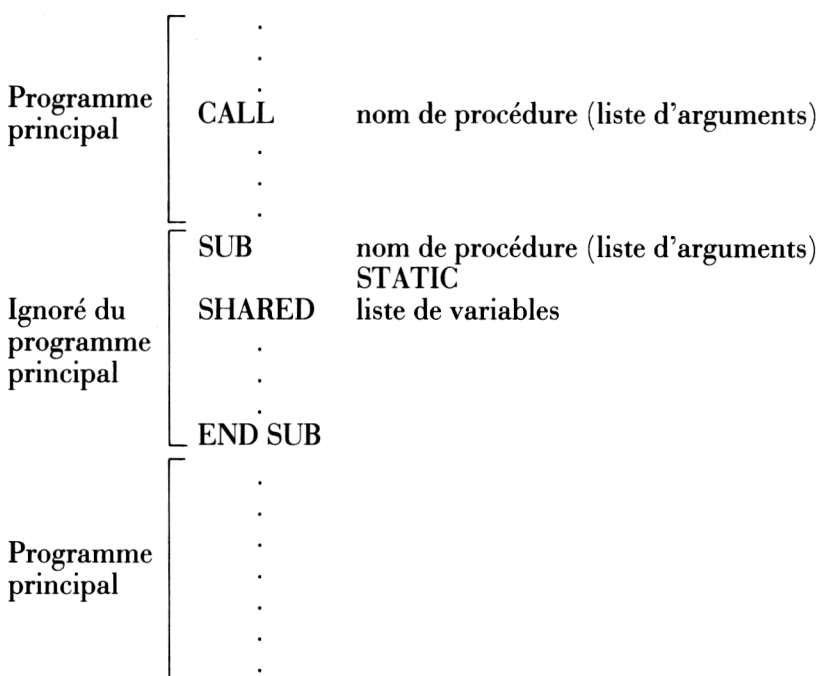
peut donc utiliser dans la procédure n'importe quel nom de variable sans risque de conflit.

Les variables qui doivent être communes au programme principal et à la procédure seront déclarées **SHARED** dans la procédure.

Lors de l'appel de la procédure on transmet la liste des valeurs nécessaires qui doivent être reprises (sous un nom différent mais avec le même format) dans la définition de la procédure.

Un tableau est transmis en précisant si nécessaire le nombre de dimensions (et non la taille) entre parenthèses.

Exemple :



TECHNIQUES D'ANIMATION

● INKEY\$

Au niveau 3 nous avons déjà utilisé **INKEY\$** pour saisir au vol une touche du clavier sans interruption du programme.

Ici, **INKEY\$** est utilisé pour provoquer le déplacement d'un objet sur l'écran. Le principe consiste à analyser la touche frappée et à modifier l'abscisse ou l'ordonnée de l'objet considéré en fonction de la valeur de la touche.

Dans la plupart des programmes les touches retenues pour le déplacement sont les touches 2, 4, 6 et 8 du pavé numérique.

- SCROLL

L'AmigaBasic dispose d'une très puissante instruction : SCROLL, qui permet de déplacer dans n'importe quelle direction l'ensemble des points présents dans le rectangle désigné. Elle permet d'obtenir des animations très spectaculaires.

Exemple :

SCROLL (x_3, y_1) - (x_2, y_2), dépl.x, dépl.y.

Si un objet se trouvant dans le rectangle désigné devait rester immobile il pourra être nécessaire de l'effacer dans sa nouvelle position et de le redessiner à son emplacement initial. On peut aussi effectuer un deuxième SCROLL sur cet objet.

- GET et PUT

L'instruction GET permet d'emmagasiner dans un tableau la portion d'écran se trouvant dans le rectangle désigné. PUT permettra ensuite de restituer l'image à n'importe quel emplacement de l'écran.

Syntaxe : GET (x_1, y_1) - (x_2, y_2), tableau
PUT (x_3, y_3), tableau.

Il sera nécessaire de dimensionner convenablement le tableau (de préférence en entier) en tenant compte du nombre de couleurs de l'emplacement photographié.

- POINT

Renvoie la couleur d'un point de l'écran.

Syntaxe : c = POINT (x,y).

- TIMER

Lors de l'allumage la variable TIMER de l'Amiga est initialisée à 0, son contenu évoluant en permanence au cours du temps.

On peut se servir de cette variable pour chronométrer des durées de façon précise en enregistrant la valeur de TIMER en début de chronométrage et en comparant avec la valeur finale.

Il est également possible de réaliser ceci automatiquement en utilisant une gestion d'interruption sous la forme :

ON TIMER (x) GOSUB étiquette.

Dans ce cas, la mise en route du chronométrage est déclenchée par TIMER ON et arrêtée par TIMER OFF.

LES FENETRES

Il est possible d'ouvrir simultanément plusieurs fenêtres sur l'écran et de les activer à tour de rôle.

Une fenêtre sera ouverte par l'instruction :

WINDOW numéro de fenêtre, "TITRE", taille, type, écran.

L'écran doit correspondre à celui sur lequel on veut faire apparaître la fenêtre, défini par **SCREEN**. Le type est un nombre qui indique de quelle sorte de fenêtre il s'agit (fixe, mobile, avec gadget de fermeture, de taille, etc.).

Une fenêtre est activée par **WINDOW OUTPUT** numéro de fenêtre. On peut la fermer avec **WINDOW CLOSE** numéro de fenêtre.

LES MENUS

AmigaBasic dispose d'excellentes facilités pour la gestion des menus déroulants, le bouton de droite de la souris étant d'ailleurs réservé à cet usage.

Les lignes de menus seront créées par des instructions de la forme :

MENU numéro de menu, numéro de ligne, état, "TITRE".

Le numéro de menu est un entier pouvant aller de 1 à 10. Le numéro de ligne varie de 0 à 20 et correspond aux diverses rubriques disponibles dans le menu choisi. L'état peut prendre les valeurs 0, 1 ou 2. 0 désactive la rubrique (elle apparaît en grisé), 1 l'active et 2 permet d'imposer un indicateur de sélection sur la rubrique.

MENU(0) renvoie le numéro de menu sélectionné.

MENU(1) renvoie le numéro de rubrique.

MENU ON active le processus de fonctionnement des menus désactivé par **MENU OFF** ou simplement suspendu par **MENU STOP**.

ON MENU GOSUB permet de dérouter vers un sous-programme spécifique.

Dans tous les cas, l'accès aux menus se fait à l'aide du bouton de droite de la souris.

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 une croix devient visible sur la droite. 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 8 fait apparaître la fumée au bas de l'écran ; la ligne 9 permet de changer (vers la droite ou vers la gauche) la direction de la colonne, et la ligne 11 fait monter la colonne.

Le joueur apparaît grâce à la ligne 13, et son déplacement est effectué par les lignes 14 à 19.

Les lignes 25 et 26 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'une croix colorée dans un coin de l'écran, va se déplacer sur elle à l'aide des touches 2, 4 et 6. 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.

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 SCREEN 1,320,200,4,1:WINDOW 1,"SMOKEY",,2,1
3 COLOR 3,1:RANDOMIZE TIMER
4 DIM p(26,26):e$=CHR$(164)
5 s=0:x=21:y=INT(RND*8+6)
6 u=0:v=12:CLS
7 debut:
8 LOCATE x+1,y+1:COLOR 2:PRINT e$;
9 y=y-(RND<.5 AND y<24) + (RND<.5 AND y>1)
10 t=1:p(x,y)=1
11 IF RND<.8 THEN x=x-t
12 WHILE u<20 AND x>0
13 LOCATE u+1,v+1:COLOR 8:PRINT CHR$(165);
14 IF t<>0 THEN FOR I=1 TO 300:a$=INKEY$
15 IF t<>0 AND a$="" THEN NEXT I
16 LOCATE u+1,v+1:PRINT " ";
17 u=u-((a$="2") AND t=1 AND x<24)
18 u=u-(t=0)
19 v=v-((a$="6") AND v<24)+((a$="4") AND v>1)
20 c=p(u+1,v):SOUND 200,1:SOUND 150,1:SOUND 200,2
21 IF c<>1 AND u<20 THEN debut
22 t=0:s=s+1:SOUND INT(RND*4000+500),1
23 WEND
24 COLOR 3:LOCATE 1,1
25 IF s>=8 AND x>0 THEN PRINT"BRAVO" ELSE PRINT "PERDU"
26 PRINT:PRINT"SCORE :";s

```

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 en ligne 11 non dimensionné (c'est inutile en dessous de 10).

Les lignes 4 à 13 (boucle I) dessinent le cadre, les ZOMBIES et les trous noirs.

La ligne 14 définit la position initiale du joueur.

Une deuxième boucle I (17-25) réalise le déplacement du joueur.

Les lignes 30 à 33 contrôlent le déplacement du joueur.

POUR JOUER :

Après avoir lancé l'exécution par RUN, le joueur devra se déplacer en utilisant les touches 2, 4, 6 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 coïncider.

Pour arrêter la partie avant la fin, il suffira de taper CTRL et Break.

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  SCREEN 1,320,220,4,1:WINDOW 1,"ZOMBIES",,2,1
3  RANDOMIZE TIMER:CLS:DIM t(25,25)
4  FOR i=0 TO 21
5  COLOR 4,6:LOCATE 1,i+2:PRINT CHR$(127);t(1,i+2)=127
6  LOCATE i+1,1:PRINT CHR$(127);t(i+1,1)=127
7  LOCATE i+1,24:PRINT CHR$(127);t(i+1,24)=127
8  IF RND<.5 AND i<>21 THEN
9  p=INT(RND*20+2):LOCATE p,i+1:COLOR 2,9:PRINT CHR$(3
5);t(p,i+1)=35
10 END IF
11 s=INT(RND*14+1):z(INT(i/3)+1)=5+s
12 LOCATE 22,i+2:COLOR 4,6:PRINT CHR$(127);t(22,i+2)=
127
13 NEXT i
14 k=INT(RND*18+1)
15 debut:
16 p=0
17 FOR i=1 TO 4
18 p=p+z(i)
19 IF z(i)<>0 THEN
20 LOCATE z(i)+1,z(i+4)+1:COLOR 3,0:PRINT " "
21 z(i)=z(i)-SGN(z(i)-s):z(i+4)=z(i+4)-SGN(z(i+4)-k)
22 IF t(z(i)+1,z(i+4)+1)=35 THEN z(i)=0 ELSE LOCATE z(
i)+1,z(i+4)+1:PRINT"Z";
23 IF z(i)=s AND z(i+4)=k THEN LOCATE 22,1:PRINT"VOUS
AVEZ PERDU...";GOTO fin
24 END IF
25 NEXT i
26 IF p=0 THEN LOCATE 22,1:PRINT"TOUS LES ZOMBIES SONT
DETRUITS";GOTO fin
27 LOCATE s+1,k+1:COLOR 11,0:PRINT"0"
28 FOR i=1 TO 700:a$=INKEY$:IF a$="" THEN NEXT i
29 LOCATE s+1,k+1:PRINT" "
30 k=k-2*((a$="6")-(a$="4"))
31 IF k<=1 OR k>=24 THEN LOCATE 22,1:PRINT"DOMMAGE..."
;GOTO fin
32 s=s-2*((a$="2")-(a$="8"))
33 IF s<=1 OR s>=22 THEN LOCATE 22,1:PRINT "DOMMAGE..."
;GOTO fin
34 SOUND INT(RND*400+100),3:GOTO debut
35 fin:
36 FOR i=100 TO 1200 STEP 15:SOUND i,1:NEXT i
37 END

```

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 ; un point apparaît en dessous d'un caractère, qu'il convient de frapper au clavier. Plus l'on frappe vite, et plus le point se déplace rapidement (le temps écoulé depuis le début s'affiche en fin de partie).

LE PROGRAMME :

Une variable A\$ dimensionnée en ligne 6, contiendra la liste des touches concernées. Les valeurs des touches sont entrées de la ligne 7 à la ligne 12 par appel d'un sous-programme situé dans les lignes 30 à 32.

Les boucles emboîtées J (13-16) et I (14-16) dessinent le clavier.

La ligne 18 définit la position du point imprimé par la ligne 19.

La touche frappée est testée aux lignes 20-23 et le point est effacé par la ligne 25.

Les affichages de fin sont réalisés aux lignes 27 et 28.

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 « espace » pour obtenir l'affichage du score, et du temps total.

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  SCREEN 1,320,200,4,1:WINDOW 1,"CLAVIER",,15,1
3  RANDOMIZE TIMER:CLS
4  PRINT:PRINT "(A)ZERTY ou (Q)WERTY ?":r$=UCASE$(INPUT$(1))
5  t=0:s=0:p=0
6  DIM a$(10,3)
7  j=1:IF r$="A" THEN s$="AZERTYUIOP" ELSE s$="QWERTYUIOP"
8  GOSUB ecris
9  j=2:IF r$="A" THEN s$="QSDFGHJKLM" ELSE s$="ASDFGHJKL;"
10 GOSUB ecris
11 j=3:IF r$="A" THEN s$="WXCVBN,;:=" ELSE s$="ZXCVBNM,./"
12 GOSUB ecris
13 FOR j=1 TO 3
14 FOR i=1 TO 10
15 LOCATE 5*j+1,3*i+j-3:COLOR 3,2:PRINT a$(i,j)
16 NEXT i,j
17 WHILE a$<>" "
18 x=INT(RND*10+1):y=INT(RND*3+1):SOUND INT(RND*4000+500),1
19 l=5*y+1:c=3*x+y-3:z$=a$(x,y):COLOR 1,2:LOCATE l,c:PRINT z$
20 FOR f=1 TO 250+RND*100:a$=INKEY$
21 t=t+1:IF a$="" THEN NEXT f
22 IF a$<>" " THEN a$=UCASE$(a$)
23 IF a$=a$(x,y) THEN s=s+1
24 p=p+1
25 LOCATE l,c:COLOR 3,2:PRINT z$
26 WEND
27 COLOR 1,0:LOCATE 18,1:PRINT p;" ESSAIS - ";s;" JUSTES"
28 PRINT"VOTRE TEMPS :";INT(t/100)
29 END
30 ecris:
31 FOR i=1 TO 10 : a$(i,j)=MID$(s$,i,1):NEXT i
32 RETURN

```

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 2 et 3, contiennent les informations codées qui permettront le dessin des chiffres en grand format.

Les lignes 6 à 15 s'occupent de la gestion des entrées.

La boucle I (19-23) 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 25 et 26.

Le nombre total des symboles entrés est contrôlé par la ligne 27.

Le résultat est inscrit grâce à un programme fonctionnant en automatique qui se trouve aux lignes 32 à 43 (l'automatisme est géré par la ligne 13, dans le programme principal).

Les sous-programmes placés aux lignes 45, 48, 51 et 54 effectuent l'opération indiquée, selon la valeur de Z (ligne 36).

Chaque résultat est mémorisé en A (F-1) (ligne 38).

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 taper CTRL et Break, puis RUN. Taper CTRL et Break 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 de l'Amiga), 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  c$="111100001101"
3  b$="14441333331312113131441331213112141133331414114
   131"
4  f=1:l=0:COLOR 3,0:CLS
5  entree:
6  n=0:t=0:k=0:r$="":a$=""
7  debut:
8  WHILE a$="":a$=INKEY$:WEND
9  IF k=1 THEN n=0:r$="":k=0
10 IF a$="" THEN calcul
11 x=(1 AND a$="+")+(2 AND a$="-")+(3 AND a$="*")+(4 AND a$="/")
12 r$=r$+a$
13 IF x<>0 THEN calcul
14 IF r$="-" THEN a$=r$:GOTO signe
15 IF a$< "." OR a$> "9" OR a$="/" THEN a$="":GOTO debut
16 affiche:
17 IF n=0 THEN COLOR 3,0:CLS
18 IF a$="." OR a$="-" THEN signe
19 FOR i=1 TO 5
20 s=3*VAL(MID$(b$,i+5*VAL(a$),1))-2
21 FOR j=0 TO 2
22 LOCATE 2*i+4,2*n+1+2*j:COLOR ,2*VAL(MID$(c$,s+j,1))
   :PRINT " ":LOCATE 2*i+5,2*n+1+2*j:PRINT " "
23 NEXT j,i
24 signe:
25 IF a$="-" THEN CLS:COLOR 3,2:LOCATE 10,2:PRINT "---"
26 IF a$="." THEN LOCATE 14,2*n+3:PRINT ")"
27 n=n-4*(n<28)
28 IF t<>0 THEN t=t+1:IF t<>0 AND t<=LEN(r$) THEN a$=M
   ID$(r$,t,1):GOTO affiche
29 IF k=1 AND l<>1 THEN entree
30 t=0:a$="":GOTO debut
31 calcul:
32 IF r$="-" AND a(f)=0 THEN a$=r$:GOTO signe
33 n=0:k=1
34 IF(a$="*" OR a$="/") AND VAL(r$)=0 THEN a(f)=1 ELSE
   a(f)=VAL(r$)
35 IF x<>0 THEN Z=x
36 ON Z GOSUB addit,soustrait,multi,divis
37 IF a$<> "=" THEN f=f+1 ELSE f=2
38 a(f-1)=VAL(r$)
39 IF VAL(r$)=0 THEN a(f-1)=a(f)
40 r$=STR$(VAL(r$))
41 IF LEFT$(r$,1)=" " THEN r$=RIGHT$(r$,LEN(r$)-1)
42 t=1
43 a$=MID$(r$,t,1)
44 GOTO affiche
45 addit:
46 r$=STR$(a(f-1)+a(f)):a(f)=VAL(r$):l=1
47 RETURN
48 soustrait:
49 IF f<>1 THEN r$=STR$(a(f-1)-a(f)):a(f)=VAL(r$):l=1
50 RETURN
51 multi:
52 IF f<>1 THEN r$=STR$(a(f-1)*a(f)):a(f)=VAL(r$):l=1
53 RETURN
54 divis:
55 IF f<>1 THEN r$=STR$(a(f-1)/a(f)):a(f)=VAL(r$):l=1
56 RETURN

```

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 ($100 \times 100 \times 100$).

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, non dimensionnée, contient les trois coordonnées définissant la position du joueur (ligne 23).

La position de la chambre contenant le colis piégé est définie par un tirage aléatoire de trois variables A, B, C, en ligne 5.

La boucle I (6-9) dessine le détecteur.

Le signal de proximité est calculé aux lignes 11 et 12 et la ligne 26 permet l'affichage des coordonnées de la chambre à découvrir, dans le cas où l'on a perdu.

Les lignes 13 et 14 permettent l'affichage, à la fois du signal et du temps restant à courir.

La ligne 15 réalise le décompte du temps.

La ligne 16 contrôle si l'on a trouvé ou non.

La boucle I (20-24) permet d'entrer les nouvelles coordonnées, en utilisant le sous-programme placé aux lignes 27 à 32.

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 RETURN, 45 puis RETURN, 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.

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  SCREEN 1,320,200,5,1:WINDOW 1,"LA BOMBE",,15,1
3  PALETTE 0,0,0,0:PALETTE 1,1,1,1:COLOR 0,1
4  l=100:t=200:RANDOMIZE TIMER:CLS
5  a=INT(RND*1):b=INT(RND*1):c=INT(RND*1):b$=CHR$(128)
6  FOR i=1 TO 17
7    FOR j=5 TO 7
8      LOCATE j,i+1:PRINT b$
9    NEXT j,i
10 WHILE t<>0
11   s=ABS((a/1+b+c*1)-(r(1)/1+r(2)+r(3)*1))
12   s$=LEFT$(STR$(s),7):s=VAL(s$)
13   COLOR INT(RND*14+2):LOCATE 6,3:PRINT USING "####.##"
14   ";s
15   COLOR INT(RND*14+2):LOCATE 6,15:PRINT USING "###";t
16   SOUND 800,3:t=t-INT(SQR(1))
17   IF s=0 THEN LOCATE 20,1:COLOR INT(RND*8+2):PRINT"BR
18   AVO":END
19   COLOR INT(RND*8+2):LOCATE 11,1:PRINT" - X  :"
20   LOCATE 13,1:PRINT" - Y  :"
21   LOCATE 15,1:PRINT" - Z  :"
22   FOR i=1 TO 3
23     r$=""
24     GOSUB entree
25     r(i)=VAL(r$)
26   NEXT i
27   WEND
28   COLOR 0:PRINT"PERDU . LA SOLUTION ETAIT :";a;"  ";b
29   "; " ";c:END
30   entree:
31   LOCATE 2*i+9,8:PRINT "?";r$;"  ":BEEP
32   WHILE a$="":a$=INKEY$:WEND
33   IF ASC(a$)=13 THEN a$="":RETURN
34   r$=r$+a$:a$=""
35   GOTO entree

```

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 21 et 22 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 l'Amiga est choisi en ligne 7, et le chiffre est tiré en ligne 8.

La ligne 9 réalise l'impression, et la ligne 10 détermine la durée d'affichage. La ligne 13 définit la durée du jeu, et la fin de partie est traitée par la ligne 15.

La boucle I (24-29) réalise le dessin grand format du chiffre frappé par le joueur.

Le contrôle de l'exactitude du résultat est effectué par la ligne 12.

Les lignes 19 à 30 contiennent la procédure de dessin avec une variable commune (ligne 20).

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 CTRL et Break.

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  SCREEN 1,320,200,5,1
3  WINDOW 1,"ADRESSE",,15,1:RANDOMIZE TIMER
4  FOR i=1 TO 15:PALETTE i,RND,RND,RND:NEXT i
5  t=100:s=0:PALETTE 0,0,0,0:COLOR ,0:CLS
6  tirage:
7  x=INT(RND*15+1):y=INT(RND*15+12)
8  n=INT(RND*10)
9  LOCATE x+1,y+1:COLOR INT(RND*15+1),0:PRINT n
10 FOR i=0 TO RND*600+200:a$=INKEY$:IF a$="" THEN NEXT
    i
11 u=VAL(a$):IF a$<>"" THEN CALL dessin
12 IF u=n THEN s=s+10
13 t=t-n
14 LOCATE x+1,y+1:COLOR ,0:PRINT "    "
15 IF t>=0 THEN tirage
16 CLS
17 COLOR INT(RND*15+1):PRINT:PRINT"VOTRE SCORE :";s;"
    SUR 100"
18 END
19 SUB dessin STATIC
20 SHARED u
21 c$="111100001101"
22 b$="14441333331312113131441331213112141133331414114
    131"
23 y=INT(RND*15+1)
24 FOR i=1 TO 5
25 a=3*VAL(MID$(b$,5*u+i,1))-2
26 FOR j=0 TO 2
27 LOCATE 2*i+1,3+j:COLOR ,VAL(MID$(c$,a+j,1))*y:PRINT
    " "
28 LOCATE 2*i+2,3+j:COLOR ,VAL(MID$(c$,a+j,1))*y:PRINT
    " "
29 NEXT j,i
30 END SUB

```

Le jeu du missile

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : contre l'Amiga.

Difficulté du jeu : moyenne.

LE JEU :

L'Amiga 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 l'Amiga 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 du pavé numérique ; réussirez-vous à remplir votre mission ?

LE PROGRAMME :

La ligne 6 dessine la base ennemie à survoler (sous forme d'un pavé agrandi).

La ligne 7 dessine votre base de départ.

La ligne 8 choisit l'instant de départ du missile.

La ligne 10 dessine votre avion.

Les lignes 13 et 14 définissent son mouvement suivant huit directions, en utilisant les sous-programmes en lignes 25, 27, 29 et 31.

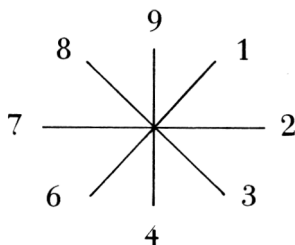
La ligne 16 assure le départ du missile qui est dessiné par la ligne 17 et dont le déplacement est géré par la ligne 18.

Les lignes 19 à 23 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, avec la convention suivante :



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 CTRL et Break 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  SCREEN 1,640,210,4,2:WINDOW 1,"MISSILE",,2,1
3  RANDOMIZE TIMER:DEFINT a-z:COLOR ,7
4  entree:
5  CLS:m=0:ON TIMER(3) GOSUB avance
6  LOCATE 6,38:COLOR ,3:PRINT "  ":LOCATE 7,38:COLOR ,
7  3:PRINT "  "
8  LOCATE 20,1:COLOR ,2:PRINT "  ";:COLOR ,7
9  u=19:x=20:s=INT(RND*u+8):y=2:v=69
10 debut:
11 LOCATE x,y:COLOR 2:PRINT CHR$(164);
12 saisie:
13 a$=INKEY$:TIMER ON:IF a$="" THEN saisie
14 f=-2*(VAL(a$)<5)-1:g=VAL(a$)+2*(f-1)
15 g=g+2 *(a$>"6" OR a$="3")-2*(a$="1" OR a$="6")
16 ON g GOSUB mo1,mo2,mo3,mo4
17 IF s<>0 THEN s=s-1:GOTO debut
18 LOCATE u,v:COLOR 0:PRINT CHR$(165);
19 u=u-SGN(u-x):v=v-SGN(v-y)
20 IF u=x AND v=y THEN PRINT"TOUCHE...";:GOTO fin
21 IF (u=6 OR u=7) AND (v=38 OR v=39) THEN entree
22 IF (x=6 OR x=7) AND (y=38 OR y=39) THEN entree
23 IF x<=6 AND y>=39 THEN m=1
24 IF m=1 AND x>=20 AND y<=4 THEN PRINT"MISSION ACCOMP
25 LIE";:GOTO fin
26 GOTO debut
27 mo1:
28 y=y+f
29 mo2:
30 x=x+f:RETURN
31 mo3:
32 GOSUB mo2
33 mo4:
34 y=y-f:RETURN
35 avance:
36 TIMER OFF:u=u-SGN(u-x):v=v-SGN(v-y)
37 LOCATE u,v:COLOR 13:PRINT CHR$(224):RETURN
38 fin:
39 FOR i=1300 TO 100 STEP -12:SOUND i,2:NEXT i
40 END
```

Horloge

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 37 vous permettra d'entrer l'heure sous forme d'un entier de quatre chiffres (par exemple : 1024 signifiera 10 heures 24 minutes). On peut garder l'heure de la machine en répondant "H" à la question "Cette heure ou une autre (H/A)".

Les lignes 19 à 24 réalisent le dessin, en décalant l'impression au fur et à mesure (à l'aide de la variable K, incrémentée en ligne 23).

Les lignes 29 à 35 assurent la progression de l'heure.

La ligne 7 prépare l'affichage suivant en utilisant la variable TIME\$ (utilisation de l'horloge interne).

POUR JOUER :

Dès le lancement par RUN, le programme se place en attente de l'heure ; on tape pour conserver l'heure de l'Amiga, ou A pour en entrer une autre. L'heure s'affiche alors assez rapidement et l'horloge est en route. Pour l'arrêter, il faudra appuyer sur les touches CTRL et Break.

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 RETURN.

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
2  SCREEN 1,320,200,4,1:WINDOW 1,"HORLOGE",,2,1
3  LOCATE 5,1:PRINT "CETTE HEURE OU UNE AUTRE (H/A)"
4  entree:
5  LOCATE 3,1:PRINT "HEURE ACTUELLE ";TIME$
6  r$=UCASE$(INKEY$):IF r$<>"A" AND r$<>"H" THEN entre
   e
7  b$=LEFT$(TIME$,2)+MID$(TIME$,4,2):a$=b$
8  IF r$="A" THEN GOSUB heure
9  WHILE a$=b$:a$=LEFT$(TIME$,2)+MID$(TIME$,4,2):LOCAT
   E 3,16:PRINT TIME$:WEND
10 debut:
11 COLOR 4,1:CLS:k=1:b$=" "
12 affiche:
13 a=VAL(MID$(a$,k,1)):COLOR ,3
14 IF a<>1 AND a<>4 THEN LOCATE 7,6*k:PRINT b$+b$+b$
15 FOR j=1 TO 5
16 IF a=0 OR (a>3 AND a<>7) THEN LOCATE j+6,6*k:PRINT
   b$
17 IF a<>5 AND a<>6 THEN LOCATE j+6,6*k+2:PRINT b$
18 IF a=0 OR a=2 OR a=6 OR a=8 THEN LOCATE j+10,6*k:PR
   INT b$
19 IF a<>2 THEN LOCATE j+10,6*k+2:PRINT b$
20 NEXT j
21 IF a>1 THEN LOCATE 11,6*k+1:PRINT b$+b$
22 IF a<>1 AND a<>4 AND a<>7 THEN LOCATE 15,6*k:PRINT
   b$+b$+b$
23 k=k+1
24 IF k<5 THEN affiche
25 IF r$="H" THEN a$=LEFT$(TIME$,2)+MID$(TIME$,4,2):GO
   TO battement
26 a=1+VAL(a$):IF a=2360 THEN a=0
27 a$="000"+RIGHT$(STR$(a),LEN(STR$(a))-1):a$=MID$(a$,
   LEN(a$)-3,4)
28 IF RIGHT$(a$,2)="60" THEN a$=RIGHT$(STR$(VAL(LEFT$(
   a$,2))+1),2)+"00"
29 battement:
30 s%=TIMER:IF s%>86340& THEN s%=86400&-s%
31 WHILE TIMER-s%<60
32 COLOR 14:LOCATE 11,16:IF (TIMER-s%) MOD 2=0 THEN PR
   INT " " ELSE PRINT ":"
33 WEND
34 SOUND 800,2
35 GOTO debut
36 heure:
37 LOCATE 8,1:INPUT "HEURE ( FORMAT HHMM ):";a$
38 IF LEN(a$)<>4 OR VAL(LEFT$(a$,2))>24 OR VAL(RIGHT$(
   a$,2))>59 GOTO heure
39 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 4 permet de dimensionner la variable A\$(I) qui contiendra les mots.

La boucle I (5) saisit la liste des mots (inscrits en DATA aux lignes 39 à 46).

La ligne 7 choisit un mot au hasard ; la boucle I (9-13) contrôle si une lettre n'a pas déjà été tirée, et reprend les lettres dans le désordre.

La ligne 14 affiche les lettres, et les lignes 18 à 22 attendent votre proposition, qui est testée dans les lignes 24 et 25.

Le nombre d'essais est incrémenté en ligne 25.

La fin de partie est contrôlée par la ligne 27, et le score est affiché en ligne 31.

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 RETURN, 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  SCREEN 1,320,200,4,1:WINDOW 1,"MOTS",,2,1
3  RANDOMIZE TIMER:DEFINT a-z
4  DIM a$(30):s=0:e=0:ON TIMER(20) GOSUB troptard
5  FOR i=1 TO 30:READ a$(i):NEXT i
6  debut:
7  WHILE a$(z)="" :z=INT(RND*30+1):WEND
8  r$=a$(z):a$="" :a$(z)="" :v$=r$:COLOR 2,1:CLS
9  FOR i=1 TO LEN(r$)
10 tirage:
11 z=INT(RND*LEN(r$)+1):IF MID$(r$,z,1)="-" THEN tirage
12 a$=a$+MID$(r$,z,1):MID$(r$,z,1)="-"
13 NEXT i
14 p$="LES LETTRES SONT :":c=4:x=1:y=3:d=0:CALL affiche
   (p$,c,x,y,d)
15 p=LEN(a$)*10+15:v=16-(LEN(a$)/2):LINE(10*v-28,50)-(
   10*v-5+p,65),9,b
16 x=v:y=7:c=4:d=1:CALL affiche (a$,c,x,y,d):COLOR 4,1
17 TIMER ON:LOCATE 11,1:PRINT"VOTRE SOLUTION :":s$="":
   v=17:COLOR 2
18 saisie:
19 r$="" :WHILE r$="" :r$=UCASE$(INKEY$):WEND
20 IF r$=CHR$(13) THEN TIMER OFF:GOTO controle
21 IF r$=CHR$(8) THEN r$="" :s$=LEFT$(s$,LEN(s$)+(LEN(s
   $)<>0))
22 s$=s$+r$:LOCATE 11,17:PRINT s$+" ":GOTO saisie
23 controle:
24 IF s$=v$ THEN s=s+1
25 e=e+1:IF s$<>v$ THEN PRINT:PRINT "LE MOT ETAIT ":C
   OLOR 5:PRINT v$
26 r$="" :COLOR 13
27 WHILE r$<>"0" AND r$<>"N"
28 LOCATE 15,1:PRINT"AUTRE ESSAI (O/N) ":r$=UCASE$(IN
   PUT$(1))
29 WEND
30 IF r$="0" THEN debut
31 COLOR 9:LOCATE 18,1:PRINT"VOTRE SCORE :":s;"SUR":e;
   "ESSAIS"
32 FOR i=100 TO 1200 STEP 12:SOUND i,1:NEXT i:END
33 SUB affiche (e$,a,b,c,d) STATIC
34 FOR i=1 TO LEN(e$)
35 IF d=0 THEN COLOR a ELSE COLOR INT(RND*7+8)
36 LOCATE c,i+b-1:PRINT MID$(e$,i,1):SOUND INT(RND*100
   0+100),2
37 NEXT i
38 END SUB
39 DATA JAMBON,ENERGUMENE,AFFILIE,ERUDIT,INSIGNIFIANT,
   REALISTE
40 DATA IMPRESSIONNANT,ALLECHANT,EPIQUE,INTREPIDE,COUA
   RD,ORDINATEUR
41 DATA TREVE,BANDEROLLE,SENAT,ACTUEL,CONSACRE,DIEDRE
42 DATA LIVRE,TERMINE,SIGNATURE,MALODORANT,AIMANT,ARBI
   TRAIRE
43 DATA CHALEUR,OPPORTUN,PELURE,ALTRUISTE,RAMONEUR,NOV
   ICE
44 troptard:
45 TIMER OFF:CLS
46 LOCATE 11,1:PRINT"DESOLE, C'EST UN PEU TARD"
47 GOTO controle

```

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 ronds ou 17 ronds...), 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 la touche 0 pour arrêter l'affichage des ronds dès que vous estimez que le nombre de ronds 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 ronds noirs sera choisi au hasard, en ligne 4.

Le temps est décompté par les lignes 4 à 10.

Les ronds sont imprimés (ligne 14) et la ligne diagonale est fabriquée par la ligne 16.

La pression de la touche 0 ou de la barre d'espace est détectée en ligne 18, et la ligne 19 contrôle la fin de partie (aucun réflexe).

Les affichages de fin sont exécutés par les lignes 21 à 26.

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 ATTENTION : IL RESTE ... SECONDES qui indique le temps restant avant l'apparition de la diagonale.

À 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 la touche 0 ou de la barre d'espace.

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  SCREEN 1,320,210,4,1:WINDOW 1,"DIAGO",,2,1
3  RANDOMIZE TIMER:CLS:a$(1)="S ":a$(0)=" "
4  x=INT(RND*18+3):ON TIMER(10) GOSUB efface
5  p=TIMER:TIMER ON
6  PRINT:PRINT"LE BON REFLEXE EST A ";x
7  tempo:
8  s=10-INT(TIMER-p)
9  LOCATE 11,1:PRINT"ATTENTION. RESTE :";s;"SECONDE";:
  PRINT a$(-(s>1)):SOUND INT(4000+500),1
10 GOTO tempo
11 efface:
12 CLS:u=0
13 debut:
14 LOCATE 22,u+1:COLOR 1:PRINT "0";
15 FOR i=1 TO 30 :a$=INKEY$:IF a$="" THEN NEXT i
16 SCROLL(0,0)-(320,210),0,-9
17 u=u+1
18 IF a$="0" OR a$=" " THEN LOCATE 1,1:GOTO test
19 IF u>22 THEN LOCATE 1,1:COLOR 5:PRINT"TROP TARD":GO
  TO fin
20 GOTO debut
21 test:
22 IF u=x THEN PRINT"BRAVO" ELSE PRINT "PERDU"
23 fin:
24 FOR i=100 TO 1200 STEP 12:SOUND i,1:NEXT i
25 COLOR 1:PRINT:PRINT"AUTRE ESSAI (O/N)"
26 a$=UCASE$(INPUT$(1)):IF a$="O" THEN RUN
```

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 l'Amiga calcule et affiche cet entier. Le procédé retenu peut d'ailleurs être généralisé (à vous de trouver comment).

LE PROGRAMME :

Les lignes 4 à 6 affichent quelques explications en appelant la procédure lecture (lignes 27 à 31).

La ligne 13 attend que vous frappiez un reste, qui sera refusé s'il est incompatible avec la division par le nombre affiché.

La ligne 14 emmagasine les restes dans une variable non dimensionnée A (S).

La ligne 14 réaffiche les restes entrés.

Les boucles I (17-20) et J (18-20) dessinent “#” selon les restes introduits.

Les lignes 23 à 25 gèrent la fin de partie.

POUR JOUER :

On lance le programme par RUN. Un texte de présentation est affiché, puis le message “RESTE DES DIVISIONS PAR 3, PAR 5, PAR 7” est affiché.

On doit répondre 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 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 rajouter du son et de la couleur pour rendre le programme plus attrayant.


```

1  REM DECOMAQUE
2  SCREEN 1,320,200,4,1:WINDOW 2,"SAISIE",(0,80)-(305,
   185),2,1
3  WINDOW 1,"DECOMAQUE",(0,0)-(305,65),2,1:COLOR 5,2:C
   LS
4  a$="VOUS DEVEZ ENTRER LES RESTES":l=2:CALL lecture(
   a$,l)
5  a$="DES DIVISIONS PAR 3 , 5 ET 7":l=4:CALL lecture(
   a$,l)
6  a$="D'UN ENTIER INFERIEUR A 100":l=6:CALL lecture(a
   $,l)
7  i=3:s=1:WINDOW OUTPUT 2
8  LOCATE 1,4:COLOR 8,11:PRINT"RESTE DES DIVISIONS PAR
   "
9  WHILE i<8
10 LOCATE 3,5*i-10:COLOR i+1:PRINT CHR$(48+i)
11 LINE (50*i-130,12)-(50*i-80,50),i+1,b
12 entree:
13 a$=INPUT$(1):IF a$<"0" OR a$>CHR$(47+i) THEN entree
   ELSE SOUND 400,2
14 a(s)=ASC(a$)-48:s=s+1:LOCATE 5,5*i-10:COLOR 1,0:PRI
   NT a$:i=i+2
15 WEND
16 COLOR 9,14:LOCATE 7,1:PRINT"  PAR 3      PAR 5      P
   AR 7"
17 FOR i=1 TO 3
18 FOR j=1 TO a(i)
19 IF a(i)<>0 THEN COLOR 2*i+2,0:LOCATE 8,13*i-2*j-7:P
   RINT "#"
20 NEXT j,i
21 t=a(1)*70+a(2)*21+a(3)*15
22 WHILE t>105:t=t-105:WEND
23 COLOR 2:PRINT:PRINT"LA SOLUTION EST ";:COLOR 3:PRIN
   T t
24 COLOR 2:PRINT "UN AUTRE ESSAI (O/N) ?";
25 a$=UCASE$(INPUT$(1)):IF a$="O" THEN RUN
26 FOR i=100 TO 1000 STEP 15:SOUND i,1:NEXT i
27 SUB lecture(r$,k) STATIC
28 FOR i=1 TO LEN(r$)
29 COLOR INT(RND*13+3):LOCATE k,i:PRINT MID$(r$,i,1):S
   OUND INT(RND*2000+200),1
30 NEXT i
31 END SUB

```

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, à la suite de quoi le texte va apparaître (une seule ligne à la fois) et commencera à défiler dans deux fenêtres qui peuvent être manipulées à volonté avec la souris.

LE PROGRAMME :

Le texte sera introduit lettre par lettre aux lignes 6 à 10.

L'affichage sera fait à l'intérieur de deux fenêtres (lignes 17 et 18).

Le défilement vers la gauche est préparé à la ligne 19, et le défilement vers la droite à la ligne 20.

La boucle WHILE...WEND (lignes 15 à 22) permet de continuer le défilement quelles que soient les manipulations sur les fenêtres. On sort du programme en tapant une touche quelconque (lignes 23 à 25).

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 RETURN. On peut corriger avec la touche BACKSPACE.

Le texte commencera à défiler, dans deux fenêtres, dès que l'on aura frappé RETURN.

On peut modifier à volonté la taille et la position des fenêtres en utilisant les gadgets correspondants.

On quitte le programme en tapant sur une touche quelconque.

EXTENSIONS POSSIBLES :

On pourra introduire de la couleur, réaliser un affichage ondulant, faire remonter les lignes sur l'écran, remodeler le texte en fonction des fenêtres, etc.

```

1  REM AFFICHAGE PUBLICITAIRE
2  SCREEN 1,640,200,4,2:WINDOW 4,"PUBLI",,2,1
3  t$="":COLOR 4,1:CLS
4  LOCATE 2,1:PRINT "ENTREZ VOTRE TEXTE"
5  COLOR 3:LOCATE 4,1:PRINT "_"
6  saisie:
7  r$="":WHILE r$="":r$=UCASE$(INKEY$):WEND
8  IF r$=CHR$(13) THEN controle
9  IF r$=CHR$(8) THEN r$="":t$=LEFT$(t$,LEN(t$)+(LEN(t$)<>0))
10 t$=t$+r$:LOCATE 4,1:PRINT t$+"_":GOTO saisie
11 controle:
12 a$=" "+t$+" ":b$=a$:r$="":SCREEN 2,320,200,4,1:WINDOW 4,"PUBLI",,2,2
13 CLS:WINDOW 1,"", (30,30)-(100,70),15,2:COLOR 3,11:CLS
14 WINDOW 2,"", (150,30)-(250,70),15,2:COLOR 4,11:CLS
15 WHILE r$=""
16 IF WINDOW(0)<>0 THEN WINDOW OUTPUT 1:COLOR ,11:CLS:WINDOW OUTPUT 2:CLS
17 WINDOW OUTPUT 2:LOCATE 2,1:COLOR 3,11:PRINT a$+" ";
18 WINDOW OUTPUT 1:LOCATE 2,1:COLOR 4,11:PRINT b$+" ";
19 a$=RIGHT$(a$,LEN(a$)-1)+LEFT$(a$,1)
20 b$=RIGHT$(b$,1)+LEFT$(b$,LEN(b$)-1)
21 r$=INKEY$:SOUND INT(RND*1000+100),2
22 WEND
23 WINDOW OUTPUT 4
24 LOCATE 1,1:PRINT "C'EST FINI...."
25 FOR i=100 TO 1200 STEP 12:SOUND i,1:NEXT i

```

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 (5-13) avec toutes les vérifications de validité nécessaires. Le nombre de tirages demandés est introduit en ligne 18.

La boucle J (20-28) comptabilise les tirages.

La boucle I (23-25) assure la conformité de chaque tirage avec les probabilités introduites en début de jeu.

Les résultats sont comptabilisés en ligne 26, et affichés grâce à la ligne 27.

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 RETURN. 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 apparaissant sous le 1 signifie que le tirage de 1, auquel on avait affecté une probabilité de x , a été réalisé 32 fois.

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  SCREEN 1,640,200,4,2:WINDOW 1,"TIRAGE D'UN DE TRUQU
   E",,2,1
3  COLOR 1,3:CLS:a$="###.##"
4  p=0:DIM e(6)
5  FOR i=1 TO 6
6    tirde:
7    LOCATE 1,18:PRINT SPACE$(30)
8    LOCATE 1,1:PRINT"PROBABILITE DU DE ";i;:INPUT e(i)
9    p=p+e(i):b$=SPACE$(5):e$=STR$(e(i)):RSET b$=e$
10   IF e(i)<0 OR e(i)>1 OR p>1.1 THEN GOSUB faux:GOTO t
       irde
11   COLOR 0:LOCATE 4,6*i-4:PRINT"DE ";i
12   COLOR 1:LOCATE 6,6*i-5:PRINT b$
13   NEXT i
14   IF p<.9 THEN BEEP:RUN
15   FOR i=1 TO 1000:NEXT i
16   WINDOW 2,"          TIRAGE", (0,100)-(630,185),2,1
17   WINDOW OUTPUT 2:COLOR 8,11:CLS
18   LOCATE 1,1:INPUT "NOMBRE DE TIRAGES :";p
19   FOR i=1 TO 6:LOCATE 3,10*i-6:PRINT "DE ";i:NEXT i
20   FOR j=1 TO p
21     tirage:
22     n=INT(RND*6+1)
23     FOR i=1 TO 6
24       IF n=i AND RND>e(i)-INT(e(i)) THEN GOTO tirage
25     NEXT i
26     e(n)=e(n)+1:SOUND INT(RND*800+200),4
27     LOCATE 5,10*n-7:COLOR n:PRINT USING a$;e(n)
28   NEXT j
29   LOCATE 8,1:COLOR 13:PRINT "C'EST FINI .... "
30   FOR i=200 TO 1000 STEP 20:SOUND i,1:NEXT i
31   END
32   faux:
33   LOCATE 2,1:PRINT"IMPOSSIBLE :0<= P <=1"
34   p=p-e(i):BEEP
35   FOR j=1 TO 3000:NEXT j
36   LOCATE 2,1:PRINT SPACE$(30)
37   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 l'Amiga 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 0).

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 4 contient une liste VO\$ des voyelles courantes, et une liste CO\$ 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 VO\$ ou deux R dans CO\$, etc.).

La ligne 6 définit la longueur du mot (en principe, de 3 à 6 lettres, mais on peut également modifier cette longueur).

La boucle I (10-15) permet la confection d'un "mot" en prenant une lettre, soit dans l'alphabet complet, soit dans la liste CO\$ (ligne 12), soit dans la liste VO\$ (ligne 13).

La ligne 21 temporise l'apparition du résultat.

Les lignes 7 et 8 gèrent l'affichage du score, et la ligne 23 s'occupe de la fin de la partie.

La ligne 25 regarde si l'un des joueurs a réagi.

Enfin, les lignes 39 et 40 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 0.

Si rien n'a été frappé, une autre série de lettres sera affichée après peu de temps.

Si l'un des joueurs a frappé 0 ou 1, le mot proposé est entré par le joueur, dans sa fenêtre, puis la question (J)USTE ou (F)AUX est affichée dans la fenêtre de l'autre joueur. On répond J si le mot est correct, F si le mot est incorrect, suivi de RETURN.

Les points sont alors calculés par l'Amiga, et la série de lettres suivante est alors affichée.

Pour sortir du programme, il faudra taper P ou la barre d'espace, au lieu de 0 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  SCREEN 1,320,210,4,1:RANDOMIZE TIMER
3  GOSUB fen2:GOSUB fen3:GOSUB fen1
4  a=0:b=0:vo$="AEIOU":co$="BCDGLMNPSTV":g$=SPACE$(40)
5  debut:
6  x=INT(RND*5+3):m$="":ON TIMER(2+(x/3)) GOSUB arret
7  COLOR 3,0:WINDOW OUTPUT 2:CLS:LOCATE 2,1:PRINT"SCOR
  E DE A :";a
8  COLOR 14,0:WINDOW OUTPUT 3:CLS:LOCATE 2,1:PRINT"SCO
  RE DE B :";b
9  WINDOW OUTPUT 1:COLOR 2,1:CLS:s$=""
10 FOR i=1 TO x
11  u$=CHR$(INT(RND*26+65))
12  IF RND<.4 THEN u$=MID$(co$,INT(LEN(co$)*RND+1),1)
13  IF RND<.4 THEN u$=MID$(vo$,INT(LEN(vo$)*RND+1),1)
14  m$=m$+u$
15 NEXT i
16 p=2*LEN(m$):LINE (144-5*p,2)-(164+5*p,25),3,b:LOCAT
  E 2,17-LEN(m$)
17 FOR i=1 TO LEN(m$)
18  c=INT(RND*13+2):COLOR c-(c=7):PRINT MID$(m$,i,1)+"
  ";
19 SOUND INT(RND*4000+500),1
20 NEXT i
21 WHILE s$="":s$=UCASE$(INKEY$):TIMER ON:WEND
22 IF s$="P" OR s$=" " THEN
23 FOR i=100 TO 1200 STEP 12:SOUND i,1:NEXT i:END
24 END IF
25 IF s$<>"0" AND s$<>"1" AND s$<>"&" AND s$<>"a" THEN
  debut
26 IF s$="0" OR s$="a" THEN WINDOW OUTPUT 3 ELSE WINDO
  W OUTPUT 2
27 LOCATE 4,1:COLOR 15,0:PRINT "VOTRE MOT ?"
28 TIMER OFF:ON TIMER (10) GOSUB arret
29 TIMER ON:t$=""
30 saisie:
31 r$="":WHILE r$="":r$=UCASE$(INKEY$):WEND
32 IF r$=CHR$(13) THEN TIMER OFF:GOTO controle
33 IF r$=CHR$(8) THEN r$="":t$=LEFT$(t$,LEN(t$)+(LEN(t
  $)<>0))
34 t$=t$+r$:LOCATE 5,1:PRINT t$+" ":GOTO saisie
35 controle:
36 IF s$="0" OR s$="a" THEN WINDOW OUTPUT 2 ELSE WINDO
  W OUTPUT 3
37 COLOR 9,0:LOCATE 4,1:PRINT"(J)USTE OU"
38 LOCATE 5,1:PRINT"(F)AUX":r$=UCASE$(INPUT$(1))
39 b=b-(r$="J" AND (s$="0" OR s$="a")) + (r$="F" AND (
  s$="0" OR s$="a"))
40 a=a-(r$="J" AND (s$="1" OR s$="&")) + (r$="F" AND (
  s$="1" OR s$="&"))
41 arret:
42 TIMER OFF:GOTO debut
43 fen1:
44 WINDOW 1,"LE SINGE",(1,1)-(310,65),2,1:COLOR 3,1:RE
  TURN
45 fen2:
46 WINDOW 2,"JOUER 1",(1,80)-(150,195),16,1:RETURN
47 fen3:
48 WINDOW 3,"JOUER 2",(150,80)-(310,195),16,1:RETURN

```

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). L'Amiga 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 (8-13) choisit dans l'alphabet entre 7 et 12 lettres, en fonction de la variable aléatoire L, tirée en ligne 6. Pour éviter d'avoir trop de consonnes, on utilise en ligne 10 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).

Les lignes 6 à 19 demandent à chacun des joueurs la longueur de son mot en appelant un sous-programme qui va de la ligne 29 à la ligne 37.

Les lignes 21 et 22 affichent le score de chaque joueur et la ligne 24 affiche le score global.

POUR JOUER :

On lance l'exécution du jeu avec RUN.

Les lettres sont alors affichées, puis la question JOUEUR 1 VOTRE MOT à laquelle le premier joueur répond en donnant le mot qu'il a obtenu. Le programme utilise des fenêtres pour les divers affichages, et chaque joueur devra taper à son tour le mot qu'il a pu former (et non sa longueur).

L'ordinateur vérifie que les lettres correspondent bien aux lettres affichées, et calcule lui-même la longueur.

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 faire CTRL et Break pour finir, ou répondre "N" à la question ON CONTINUE (O/N).

EXTENSIONS POSSIBLES :

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

On peut aussi étendre le jeu à plus de deux joueurs. Et pourquoi ne pas essayer de définir un jeu contre l'Amiga, 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  RANDOMIZE TIMER:DEFINT a-z
3  j1=0:j2=0:a$="AEIOUEAUAEIO"
4  WHILE r$<>"N" AND r$<>"n"
5  GOSUB fen1
6  l=INT(RND*6+7):f$="":CLS
7  LOCATE 2,1:PRINT"LETTRES :"
8  FOR i=1 TO l
9  r$=CHR$(INT(RND*26+65))
10 IF RND<.5 THEN r$=MID$(a$,INT(RND*LEN(a$)+1),1)
11 LOCATE 2,9+2*i:PRINT r$:SOUND INT(RND*400+200),1
12 f$=f$+r$
13 NEXT i
14 LOCATE 4,25:PRINT "ATTENTION.....":l=0:p=0
15 FOR i&=1 TO 30000+INT(RND*20000):NEXT i&
16 CLS:i=INT(RND*2+1):IF i=1 THEN GOSUB fen2 ELSE GOSU
   B fen3
17 GOSUB controle
18 i=ABS(i-3):IF i=1 THEN GOSUB fen2 ELSE GOSUB fen3
19 GOSUB controle
20 j1=j1+(p AND p>1):j2=j2+(1 AND 1>p)
21 LOCATE 6,1:PRINT "SCORE ACTUEL ";:IF i=1 THEN PRINT
   p ELSE PRINT 1
22 i=ABS(i-3):WINDOW OUTPUT i+1:LOCATE 6,1:PRINT "SCOR
   E ACTUEL ";:IF i=1 THEN PRINT p ELSE PRINT 1
23 GOSUB fen4:CLS
24 LOCATE 4,1:PRINT"SCORES - JOUEUR 1 :";j1;" - JOUEUR
   2 :";j2
25 PRINT:INPUT"ON CONTINUE ( O/N ) :";r$
26 WEND
27 PRINT "LE JOUEUR ";2+(j1>j2);" GAGNE"
28 END
29 controle:
30 LOCATE 4,1:INPUT"VOTRE MOT :";r$
31 r$=UCASE$(r$)
32 FOR j=1 TO LEN(r$)
33 m$=MID$(r$,j,1):s=INSTR(1,f$,m$):IF s=0 THEN r$="":
   GOTO suite
34 NEXT j
35 suite:
36 IF i=1 THEN p=LEN(r$) ELSE l=LEN(r$)
37 RETURN
38 fen1:
39 WINDOW 1,"LETTRES",(1,1)-(612,185):COLOR 3,1:RETURN
40 fen2:
41 WINDOW 2,"JOUEUR 1",(1,80)-(310,185):COLOR 3,1:RE
   TN
42 fen3:
43 WINDOW 3,"JOUEUR 2",(330,80)-(612,185):COLOR 3,1:RE
   TURN
44 fen4:
45 WINDOW 1,"LETTRES",(1,1)-(612,60):COLOR 3,1:RETURN

```

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 un point) 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 :

La boucle I (4-7) dessine le terrain de jeu et les briques sont mises en place par les boucles emboîtées I (8-11) et J (9-11).

La ligne 14 dessine la raquette dont la position est contrôlée par la ligne 15.

Les lignes 20 à 23 regardent si l'on a touché le mur, les briques ou la raquette.

Les rebonds sont gérés en conséquence avec des éventuels aléas, par les lignes 23 à 27.

La ligne 17 affiche le score en permanence.

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 4 et 6 du pavé numérique.

Si, à un moment quelconque, sa balle ne touche pas la raquette, le score sera affiché, la partie reprend en tapant RUN.

On peut aussi terminer à tout moment par CTRL et Break.

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  SCREEN 1,320,200,4,1:WINDOW 1,"MUR DE BRIQUES",,2,1
3  RANDOMIZE TIMER:COLOR 2,1:b$=" ":CLS
4  FOR i=1 TO 18
5  COLOR ,2:LOCATE 1,10+i:PRINT b$
6  LOCATE i+1,11:PRINT b$:LOCATE i+1,28:PRINT b$
7  NEXT i
8  FOR i=1 TO 7
9  FOR j=11 TO 26
10 LOCATE i+1,j+1:COLOR ,i+7:PRINT b$
11 NEXT j,i
12 h=-1:k=-.5:j=18:i=17:y=j
13 debut:
14 LOCATE 20,y-1:PRINT " ";:COLOR ,4:PRINT " ";:COLOR
   ,1:PRINT " ";
15 d$=INKEY$:y=y-((d$="6") AND y<28)+((d$="4") AND y>8
   )
16 IF t=1 THEN LOCATE i+1,j+1:PRINT " ";
17 LOCATE 10,1:PRINT"SCORE";s
18 IF i>20 THEN SOUND 700,3:END
19 i=i+h:j=j+k
20 c=POINT(10*j+4,9*i+4)
21 IF c=1 THEN LOCATE i+1,j+1:PRINT "o";:t=1:GOTO debu
   t
22 IF c>7 THEN f=12-i:LOCATE i+1,j+1:PRINT"o";:t=1:GOT
   O touche
23 IF c<>2 THEN h=-h:k=RND-RND
24 k=-k:t=0:SOUND 800,1:GOTO debut
25 touche:
26 s=s+9+f:k=-{k-(2*(RND<.5)+1)*RND/3}
27 h=-h:SOUND 200,2:GOTO debut

```

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 8 et 10 ; son déplacement aléatoire est géré par le sous-programme 22-23 (en particulier les lignes 29 à 32).

La ligne 12 permet de reboucler le programme sur la ligne 7.

Le déplacement du joueur est régi par les lignes 15 et 16, et le test permettant de détecter la présence d'un œuf est effectué en ligne 17.

La ligne 20 permet l'affichage du score.

La fin de jeu est détectée en ligne 26.

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 les 4 touches de déplacement. Il vous faut dès lors essayer de récupérer un maximum d'œufs avant la fin de la partie.

On pourra arrêter de jouer à tout moment en tapant CTRL et Break.

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 OEUFs
2  SCREEN 1,640,210,4,2:WINDOW 1,"OEUFs",,2,1
3  RANDOMIZE TIMER:DEFINT a-z:COLOR 1,2
4  a$(1)="\":a$(2)="#":a$(3)="/":a$(4)="_"
5  CLS:p=400:r=36:s=10:t=0
6  x=INT(RND*67):y=INT(RND*20)
7  debut:
8  LOCATE y+1,x+1:COLOR 6:PRINT a$(1)+a$(2)+a$(3);
9  GOSUB ponds
10 LOCATE y+1,x+1:COLOR 6:PRINT a$(4)+a$(2)+a$(4);
11 GOSUB ponds
12 GOTO debut
13 bouge:
14 LOCATE s+1,r+1:COLOR 0:PRINT " ";
15 r=r-((a$="6") AND r<66) + ((a$="4") AND r>0)
16 s=s-((a$="2") AND s<21) + ((a$="8") AND s>0)
17 g=POINT(10*r+4,9*s+4)
18 IF g=11 THEN FOR i=1 TO 6:SOUND 500+200*i,.2:NEXT
19 t=t-(g=11)
20 LOCATE 21,1:COLOR 9:PRINT"SCORE :";t;
21 RETURN
22 ponds:
23 FOR i=1 TO 50:a$=INKEY$
24 IF a$="" THEN NEXT i ELSE GOSUB bouge
25 a=x:b=y:p=p-1
26 IF p=0 THEN PRINT"C'EST FINI...":SOUND 100,4:END
27 LOCATE s+1,r+1:COLOR 3:PRINT CHR$(1);
28 CIRCLE(10*a+4,9*b+4),6,11:PAINT(10*a+4,9*b+4),11,11
29 x=ABS(x+INT(1+2*(RND<.5)))
30 y=ABS(y+INT(1+2*(RND<.5)))
31 IF x>66 THEN x=1
32 IF y>20 THEN y=1
33 RETURN

```

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 :

Les lignes 4 à 7 redéfinissent les caractères voiture et bordure.

L'impression de la route est assurée par les lignes 12 et 13.

La ligne 14 fait avancer la route, et la ligne 10 imprime la voiture.

Les lignes 18 et 19 contrôlent son mouvement, et la ligne 20 détermine si la route fait ou non des virages.

La ligne 17 regarde si l'on a touché le bord de la route.

La ligne 24 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 4 et 6 du pavé numérique. Il vous faudra éviter le plus longtemps possible de toucher une bordure, ce qui provoquera l'arrêt du programme et l'affichage du temps de parcours.

On pourra arrêter le déroulement de la partie à tout moment en pressant les touches CTRL et Break.

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 VOITURES
2  SCREEN 1,320,210,4,1:WINDOW 1,"COURSE",,2,1
3  DEFINT a-z:DIM d%(15),dess%(70)
4  RESTORE voiture:GOSUB lecture
5  LINE(0,0)-(15,15),i+2,bf
6  GET (0,0)-(15,15),dess%
7  RESTORE efface:GOSUB lecture
8  CLS:RANDOMIZE TIMER:t=TIMER
9  a=30:deb=40:fin=56
10 PUT (deb,61),dess%(0)
11 debut:
12 LINE(a,190)-(a+9,210),11,bf
13 LINE(a+54,190)-(a+63,210),11,bf
14 SCROLL(0,50)-(130,210),0,-1
15 SCROLL(deb,60)-(fin,76),0,1
16 b=POINT(deb+2,78):c=POINT(deb+13,78)
17 IF b=11 OR c=11 THEN fin
18 r$=INKEY$:x=(r$="4")-(r$="6")
19 IF x<>0 THEN SCROLL(deb-2,60)-(fin+2,76),4*x,0:deb=
    deb+4*x:fin=fin+4*x
20 y=4-INT(RND*9):a=a+y:a=a+a*(a<0)+(a>67)*(a>67)
21 GOTO debut
22 fin:
23 FOR i=10 TO 2000 STEP 20:SOUND i,1:NEXT i
24 PRINT:PRINT"TEMPS :";INT(TIMER-t):END
25 lecture:
26 FOR j=0 TO 15:READ d%(j):NEXT j
27 PATTERN ,d%
28 RETURN
29 efface:
30 DATA &HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFF
    FF,&HFFFF
31 DATA &HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFFFF,&HFF
    FF,&HFFFF
32 voiture:
33 DATA &H240,&H37EC,&H37EC,&H3FFC,&H37EC,&H37EC,&H7E0
    ,&H5A0
34 DATA &H5A0,&H660,&H3C0,&HBDO,&HFF0,&HBDO,&H3C0,&H18
    0

```

Obstacles

REFERENCES :

Difficulté de programmation : assez difficile.

Catégorie de jeu : jeu de réflexe (pour un joueur).

Difficulté du jeu : moyenne.

LE JEU :

Des obstacles 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 obstacle.

LE PROGRAMME :

Les lignes 4 à 13 définissent les caractères voitures et piétons.

La ligne 19 fait avancer les obstacles vers la voiture du joueur qui est imprimée grâce à la ligne 20.

La ligne 18 fabrique la route.

Les lignes 26 et 27 impriment les piétons.

La ligne 32 affiche le score en cas d'accident.

Les lignes 23 et 24 permettent de déplacer le véhicule.

Les lignes 21 et 22 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 23.

POUR JOUER :

On lance le programme en tapant RUN.

La route et les obstacles apparaissent en bas de l'écran et montent vers la voiture. La voiture peut se déplacer à l'aide des flèches.

La rencontre avec un obstacle ou la bordure interrompt le déroulement du programme et provoque l'affichage du score.

On peut, cependant, sortir du programme à tout moment en utilisant les touches CTRL et Break.

EXTENSIONS POSSIBLES :

Il sera assez facile d'améliorer le dessin de la voiture, de la route, des obstacles. 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 SCREEN 1,320,210,4,1:WINDOW 1,"OBSTACLES",,2,1
3 DEFINT a-z:DIM d%(15),dess%(400)
4 FOR i=1 TO 3
5 IF i=1 THEN
6 RESTORE voiture
7 ELSE
8 IF i=2 THEN RESTORE pieton ELSE RESTORE dame
9 END IF
10 GOSUB lecture
11 LINE(96*i-48,80)-(96*i-32,95),i+2,bf
12 GET (96*i-48,80)-(96*i-32,95),dess%(130*(i-1))
13 NEXT i
14 CLS:RANDOMIZE TIMER:t=TIMER
15 X=22:Y=11:a=0:b=10:s=0:deb=40:fin=56
16 PUT (40,61),dess%(0)
17 debut:
18 LINE(15,190)-(24,210),11,bf:LINE(119,190)-(128,210),11,bf
19 SCROLL(15,50)-(128,210),0,-1
20 SCROLL(deb,60)-(fin,76),0,1:IF deb<25 OR fin>118 THEN fin
21 a=POINT(deb+2,78):b=POINT(deb+8,81):c=POINT(deb+13,78)
22 IF a=4 OR b=4 OR c=4 OR a=5 OR b=5 OR c=5 THEN fin
23 r$=INKEY$:a=(r$="4")-(r$="6"):s=s+(d MOD 16)
24 IF a<>0 THEN SCROLL(deb-2,60)-(fin+2,76),4*a,0:deb=deb+4*a:fin=fin+4*a
25 d=d+1:IF (d MOD 32)=0 THEN
26 d=0:IF RND<.9 THEN PUT(25+16*INT(RND*5),180),dess%(260)
27 IF RND<.8 THEN PUT(25+16*INT(RND*5),180),dess%(130)
28 END IF
29 GOTO debut
30 fin:
31 FOR i=10 TO 2000 STEP 20:SOUND i,1:NEXT i
32 PRINT"ACCIDENT :SCORE :";s
33 PRINT:PRINT"TEMPS :";INT(TIMER-t):END
34 lecture:
35 FOR j=0 TO 15:READ d%(j):NEXT j
36 PATTERN ,d%
37 RETURN
38 voiture:
39 DATA &H240,&H37EC,&H37EC,&H3FFC,&H37EC,&H37EC,&H7E0,&H5A0
40 DATA &H5A0,&H660,&H3C0,&HBD0,&HFF0,&HBD0,&H3C0,&H180
41 pieton:
42 DATA &H7F0,&H7F0,&H1FFC,&H7D0,&H7F8,&H3E0,&H1C0,&H6B0
43 DATA &H7F8,&H1FFC,&H3B6E,&H33E6,&H3E0,&H770,&HE38,&HF1C
44 dame:
45 DATA &H0,&H3C0,&HF40,&H1BE0,&H1BC0,&H1980,&H7E0,&HF0
46 DATA &H1BD8,&H33CC,&H7E0,&H7E0,&HFF0,&H240,&H240,&H360

```

Echiquier

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : graphisme.

Difficulté du jeu : variable, selon utilisation.

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 :

Les lignes 3 à 12 définissent les divers menus.

La ligne 13 lit les sept couleurs autorisées ici.

La boucle 14-17 attend que le joueur clique avec le bouton de droite sur le menu choisi.

Le sous-programme 18-19 choisit la taille.

Les sous-programmes 20 et 22 choisissent la couleur.

Le sous-programme 24-30 permet d'exécuter le dessin ou de sortir.

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 de cliquer les couleurs, et enfin de cliquer sur "dessin" dans le menu "ACTION".

On peut quitter à tout moment en cliquant sur "fin" dans le menu "ACTION".

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 ECHIQUEUR
2  SCREEN 1,320,200,4,1:WINDOW 1,"ECHIQUEUR",,2,1
3  MENU 1,0,1,"TAILLE":MENU 1,1,1,"4":MENU 1,2,1,"5"
4  MENU 1,3,1,"6":MENU 1,4,1,"7":MENU 1,5,1,"8"
5  MENU 1,6,1,"9":MENU 1,7,1,"10":MENU 2,0,1,"COUL 1"
6  MENU 2,1,1,"blanc":MENU 2,2,1,"bleu":MENU 2,3,1,"ve
   rt"
7  MENU 2,4,1,"jaune":MENU 2,5,1,"rouge":MENU 2,6,1,"o
   range"
8  MENU 2,7,1,"noir":MENU 3,0,1,"COUL 2":MENU 3,1,1,"b
   lanc"
9  MENU 3,2,1,"bleu":MENU 3,3,1,"vert":MENU 3,4,1,"jau
   ne"
10 MENU 3,5,1,"rouge":MENU 3,6,1,"orange":MENU 3,7,1,"
   noir"
11 MENU 4,0,1,"ACTION":MENU 4,1,1,"dessin":MENU 4,2,1,
   "fin"
12 MENU 5,0,1," ":DIM co(16):CLS
13 FOR i=1 TO 7:READ co(i):NEXT
14 debut:
15 choix=MENU(0)
16 ON choix GOSUB ch1,ch2,ch3,ch4
17 GOTO debut
18 ch1:
19 h=3+MENU(1):RETURN
20 ch2:
21 car=co(MENU(1)):RETURN
22 ch3:
23 fon=co(MENU(1)):RETURN
24 ch4:
25 IF MENU(1)=2 THEN END
26 CLS:a=156-8*h:b=92-8*h
27 FOR i=0 TO h*h-1:j=i MOD h:k=INT(i/h)
28 c=car*((j+k) MOD 2)+fon*((j+k+1)MOD 2)
29 GOSUB grille:NEXT i:i=1
30 RETURN
31 grille:
32 LINE(a+16*j,b+16*k)-(a+16*j+16,b+16*k+16),,b
33 LINE(a+16*j+1,b+16*k+1)-(a+16*j+14,b+16*k+14),c,BF
34 RETURN
35 DATA 1,4,14,11,3,9,2

```

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 3 à 6) 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 grâce au sous-programme 38-39. La boucle I (11-13) effectue l'impression des cibles.

La ligne 17 affichera le score en permanence.

La balle est affichée par la ligne 19.

La boucle I (21-26) réalise l'impression des chiffres contenus dans les cibles, et permet le déplacement de la boule (ligne 25).

La ligne 28 regarde si l'on a tiré, et si la boule est encore sur la ligne de départ.

La ligne 29 fait progresser la boule d'une case vers la droite, et la ligne 30 vérifie si l'on a ou non atteint les cibles ; puis les lignes 31 et 32 regardent ce que l'on a touché.

La ligne 32 incrémente le score.

Les lignes 34 à 37 s'occupent de la fin de partie.

POUR JOUER :

Lancer le programme par RUN. Le programme utilise, pour le choix des options, une barre de menu.

Le niveau est choisi avec la souris en utilisant les menus déroulants (on clique sur le bouton droit). On clique le nombre de cibles choisi, puis on choisit le niveau (facile, moyen ou difficile) en cliquant sur la case correspondante dans une deuxième fenêtre. Au niveau facile, il est possible de continuer à contrôler le mouvement de la boule après l'avoir lancée. Ce n'est plus possible au niveau difficile. On fait ensuite monter ou descendre la balle avec les touches 2 et 8 du clavier numérique ou les touches B et H, puis on lance en appuyant sur la touche "Ø" ou la barre d'espace.

Le score sera modifié en fonction de votre résultat, et une nouvelle balle apparaît. Il est possible d'arrêter à tout moment en tapant CTRL et Break, sinon le jeu s'arrête au bout d'un certain nombre de tirs.

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  MENU 1,0,1,"NOMBRE":MENU 1,1,1," 3 ":MENU 1,2,1," 4
   ":MENU 1,3,1," 5 "
3  MENU 1,4,1," 6 ":MENU 2,0,1,"NIVEAU":MENU 2,1,1,"FA
   IBLE":MENU 2,2,1,"MOYEN"
4  MENU 2,3,1," FORT ":MENU 3,0,1,"OPTIONS":MENU 3,1,1
   ,"DEBUT":MENU 3,2,1," FIN "
5  MENU 4,0,1," ":RANDOMIZE TIMER:n=3:SCREEN 1,640,200
   ,4,2 :WINDOW 1,"CIBLES",,2,1:CLS
6  choix:
7  ON MENU(0) GOSUB ch1,ch2,ch3
8  GOTO choix
9  debut:
10 CLS:IF dif=1 THEN WINDOW 1,"CIBLES",(0,0)-(320,185
   ),2,1:v=2 ELSE WINDOW 1,"CIBLES",(0,0)-(610,185),2,1:v=1
11 FOR i=1 TO n
12 LINE(560/v,18*i-5)-(600/v,18*i-5):LINE-(600/v,18*i+
   13):LINE-(560/v,18*i+13)
13 NEXT i
14 p=9:j=58/v:s=0:b=1
15 WHILE p<>0
16 k=0:a=19
17 LOCATE 1,1:COLOR 4:PRINT"SCORE :";s;" - BALLE :";b;
18 deplace:
19 LOCATE a,k+1:COLOR 2:PRINT "o";
20 d=a
21 FOR i=1 TO n
22 u=INT(RND*p)
23 LOCATE 2*i+1,j:COLOR 1:PRINT CHR$(u+48):c'(i)=u
24 a$=UCASE$(INKEY$)
25 IF k=0 OR dif<>3 THEN a=a-(a$="2" OR a$="B")+(a$="8
   " OR a$="H")
26 NEXT i
27 LOCATE d,k+1:PRINT " "
28 IF k=0 AND (a$="0" OR a$=" ") THEN b=b+1:k=1:GOTO d
   eplace
29 IF k=0 THEN deplace ELSE k=k+1
30 IF k<j THEN deplace
31 IF ((a-1)MOD 2)<>0 OR a<3 OR a>2*n+2 THEN fin
32 s=s+c((a-1)/2):SOUND INT(RND*4000+500),2:p=p-1
33 WEND
34 fin:
35 PRINT "C'EST FINI....."
```

```
36 FOR i=100 TO 1200 STEP 15:SOUND i,1:NEXT i
37 END
38 ch1:
39 n=MENU(1)+2:RETURN
40 ch2:
41 dif=MENU(1):RETURN
42 ch3:
43 IF MENU(1)=1 THEN debut
44 FOR i=100 TO 1000 STEP 10:SOUND i,1:NEXT i:END
45 RETURN
```

Niveau 5

<i>Nom du programme</i>	<i>Thèmes et particularités</i>
Plus grand, plus petit	Un jeu de déduction intéressant, pour un joueur.
Anagrammes	Un bon jeu de réflexes, pour deux joueurs.
Permutations	Un jeu de réflexion, pas toujours évident.
Le jeu du 15	Un jeu de réflexion utilisant des chiffres.
Dessin de cercles	Tirage d'un dé, avec graphisme.
Reines sur l'échiquier	Un vieux problème, pour vous aider à réfléchir.
Gouttes	Un véritable gag visuel, à exploiter.
Ardoise magique	Mettez au point vos créations graphiques, sans papier ni crayon.
Tic Tac Toc	Un classique, mais pas sans intérêt, pour deux joueurs.
Quatre à quatre	Un jeu de réflexion un peu difficile, pour deux joueurs.
Taquin de lettres	Un proverbe est caché dedans ; à vous de le découvrir.
Parcours du cavalier	Très spectaculaire et instructif à plus d'un titre.
Anomalie	Exercez votre perspicacité dans la reconnaissance des symboles.
L'indésirable	Vous êtes propriétaire. Arrivez-vous à mettre la main sur ce locataire indésirable ?
Ordre	Un peu d'astuce pour remettre en ordre, le plus vite possible.
Dames	Pour deux joueurs : classique.
Othello	Pour deux joueurs ; il retourne les pions à votre place.
Echecs	Un programme assez simplifié, pour deux joueurs.
Poker	Une base de départ, pour mettre au point votre programme.
Fais-moi un dessin	Composez vous-même votre paysage avec des images déjà définies.
Animation d'images	Créez vous-même votre dessin animé.
Captures dans l'espace	Votre mission serait facile, sans ces damnées météorites.
Danger spatial	Serez-vous assez adroit pour éviter toutes les météorites.
Combat aérien	Pas si facile de descendre un appareil qui bouge tout le temps.
Sous-marin	Un classique des jeux vidéo, pour un joueur.

Souris, Sprites et Bobs

LA SOURIS

Au niveau 4 nous avons eu l'occasion d'utiliser le bouton de droite de la souris qui offre l'accès aux menus. Les programmes proposés en niveau 5 font un usage extensif de la souris, dans son emploi le plus répandu, puisqu'il s'agit ici de désigner des points sur l'écran avec ce périphérique particulier.

La fonction `MOUSE(0)` permet de savoir s'il y a eu, ou pas, action sur le bouton de gauche de la souris. `MOUSE(0)` prend la valeur 0 tant que le bouton n'est pas actionné.

Les fonctions `MOUSE(1)` et `MOUSE(2)` renvoient les coordonnées horizontale et verticale du curseur de la souris lors du dernier appel de `MOUSE(0)`.

`MOUSE(3)` et `MOUSE(4)` renvoient les valeurs des coordonnées juste avant l'appui sur le bouton.

`MOUSE(5)` et `MOUSE(6)` renvoient les valeurs des coordonnées juste avant que l'on relâche le bouton.

`MOUSE ON` active le processus de capture de ces informations qui peut être désactivé par `MOUSE OFF` et suspendu par `MOUSE STOP`. `ON MOUSE GOSUB` permet de dérouter vers un sous-programme spécifique.

LES SPRITES ET BOBS

L'AmigaBasic dispose de plusieurs instructions permettant de gérer des objets graphiques autonomes de type `SPRITES` ou `BOBS`. Les objets graphiques peuvent se déplacer sur l'écran d'une façon totalement indépendante du programme, sans ralentir le fonctionnement de ce dernier.

Les `SPRITES` sont plus rapides que les `BOBS`, mais plus limités en taille et en couleur. Les instructions de gestion s'appliquent indifféremment aux uns et aux autres.

Pour définir les `SPRITES` ou les `BOBS` on devra normalement utiliser le programme spécifique fourni avec la disquette AmigaBasic. Les astuces ont été utilisées dans les divers programmes se servant de ces objets graphiques pour éviter ce problème.

`OBJET.SHADE` numéro,définition : permet de définir des sprites ou des bobs.

OBJET.ON numéro : rend l'objet graphique visible.

OBJET.OFF numéro : le rend visible.

OBJET.PLANES numéro,plan de bit : définit le plan de couleur et l'objet graphique.

OBJET.X n,n_1 } positionne l'objet n.
OBJET.Y n,n_2 }

OBJET.VX n,n_1 } définit sa vitesse.
OBJET.VY n,n_2 }

OBJET.AX n,n_1 } définit son accélération.
OBJET.AY n,n_2 }

OBJET.START numéro } met en mouvement ou arrête
OBJET.STOP numéro } l'objet désigne.

COLLISION ON : active le processus de détection de collision désactivé par **COLLISION OFF** et suspendu par **COLLISION STOP**. **ON COLLISION GOSUB** permet de dérouter vers un sous-programme spécifique.

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 2 à 6 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 :

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 19 (qui teste si l'on a trouvé juste), et dans la boucle I (23-26) où l'on fait la comparaison "lettre à lettre".

L'affichage du résultat est réalisé grâce aux lignes 27 et 28.

Le compteur du nombre d'essais est en ligne 29.

POUR JOUER :

Le lancement du programme par RUN provoque l'affichage d'une fenêtre, et l'on devra cliquer le chiffre correspondant au niveau choisi.

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

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 l'Amiga affichera TROUVE... EN... ESSAIS. Puis AUTRE PARTIE (O/N) ; on fera O pour commencer une nouvelle partie.

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 l'Amiga qui tiendra alors le rôle de décodeur.

```

1  REM PLUS GRAND,PLUS PETIT
2  SCREEN 1,320,200,4,1:WINDOW 1,"PLUS GRAND,PLUS PETI
   T",,2,1
3  RANDOMIZE TIMER:CLS
4  WINDOW 4," CLIQUEZ LE NIVEAU",(70,60)-(220,90),16,1
5  WINDOW OUTPUT 4
6  FOR i=2 TO 6
7  LINE(30*i-60,0)-(30*i-30,30),i,bf:LOCATE 2,3*i-4:PR
   INT CHR$(48+i);
8  NEXT i
9  WHILE ABS(MOUSE(0))<2 OR b<>1 OR ( a MOD 3 )<>1
10 a=INT(MOUSE(3)/10):b=INT(MOUSE(4)/9)
11 WEND
12 WINDOW CLOSE 4:WINDOW OUTPUT 1
13 n=INT((a-1)/3)+2:e=1:f=1:f$=SPACE$(8)
14 x$=STR$(INT(RND*9^n+10^(n-1))):x$=RIGHT$(x$,LEN(x$)
   -1)
15 LINE (0,0)-(320,30),15,bf:LINE (0,30)-(320,30),9
16 LINE (45,30)-(45,200),9:LINE (130,30)-(130,200),9
17 LINE (220,30)-(220,200),9:COLOR 13,10:LOCATE 1,12:P
   RINT "NIVEAU ";n
18 COLOR 8,15:LOCATE 3,1:PRINT"NO.   NOMBRE   >
   <":COLOR 2,0
19 WHILE e$<>x$
20 COLOR 3:LOCATE 4+f,1:PRINT e;
21 COLOR 2,0:LOCATE 4+f,6:PRINT SPACE$(7);
22 a=0:b=0:LOCATE 4+f,6:INPUT "",e$
23 FOR i=1 TO n
24 IF MID$(e$,i,1)>MID$(x$,i,1) THEN a=a+1
25 IF MID$(e$,i,1)<MID$(x$,i,1) THEN b=b+1
26 NEXT i
27 COLOR 11:LOCATE 4+f,17:PRINT a;
28 COLOR 14:LOCATE 4+f,27:PRINT b;:SOUND 250,3
29 e=e+1:f=f+1:IF f=16 THEN f=1
30 WEND
31 FOR i=200 TO 1000 STEP 10:SOUND i,1:NEXT i
32 PRINT:PRINT"TROUVE EN ";e-1;" ESSAIS"
33 PRINT"AUTRE PARTIE ( O/N )?";:r$=UCASE$(INPUT$(1))
34 IF r$="O" THEN RUN ELSE END

```

Anagrammes

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu de réflexe (2 joueurs).

Difficulté du jeu : selon le niveau.

LE JEU :

Des lettres sont choisies au hasard par l'Amiga, 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 (1 ou Ø) 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 en cliquant sur le chiffre convenable avec la souris (boucle 9-11). La boucle I (13) réalise le choix des lettres.

Une deuxième boucle I (16-23) effectue le brouillage, et la ligne 20 permet de détecter la pression d'une touche.

La ligne 26 affiche un nouveau mot, et la ligne 27 fait monter d'une ligne sur l'écran les mots déjà obtenus.

La ligne 34 cherche quel est celui des deux joueurs qui a pressé une touche.

La boucle I (35-37) explore les mots présents sur l'écran ligne par ligne.

On obtient ainsi un mot qui est comparé au mot initial en ligne 36 et le résultat BRAVO apparaît si c'est exact.

Sinon, la partie s'arrête sur l'affichage PERDU (ligne 38).

POUR JOUER :

On lance le programme par RUN.

On devra ensuite entrer le niveau de difficulté (en principe de 1 à 4). On fait appel à une fenêtre d'alerte pour choisir le niveau : on clique avec le bouton de gauche sur le niveau choisi (1 à 4) pour lancer le programme. 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 (1 ou Ø), et son résultat sera affiché après un court instant.

La partie sera alors terminée.

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  SCREEN 1,320,200,4,1:WINDOW 1,"ANAGRAMMES",,2,1
3  RANDOMIZE TIMER:CLS:s$="":DIM t$(20)
4  WINDOW 4,"CHOIX DU NIVEAU",(80,60)-(200,90),16,1
5  WINDOW OUTPUT 4
6  FOR i=1 TO 4
7  LINE(30*i-30,0)-(30*i,30),i,bf:LOCATE 2,3*i-1:PRINT
   CHR$(48+i);
8  NEXT i
9  WHILE ABS(MOUSE(0))<2 OR b<>1 OR (a MOD 3)<>1
10 a=INT(MOUSE(3)/10):b=INT(MOUSE(4)/9)
11 WEND
12 WINDOW CLOSE 4:n=INT((a-1)/3)
13 FOR i=1 TO n+3:s$=s$+CHR$(INT(RND*26+65)):NEXT i
14 debut:
15 a$="":b$=s$
16 FOR i=1 TO LEN(s$)
17 tirage:
18 n=INT(RND*LEN(s$)+1)
19 d$=INKEY$
20 IF d$="1" OR d$="&" OR d$="a" OR d$="0" THEN suite
21 IF MID$(b$,n,1)="-" THEN tirage
22 a$=a$+MID$(b$,n,1):MID$(b$,n,1)="-"
23 NEXT i
24 COLOR INT(RND*15+1)
25 FOR i=1 TO LEN(a$)
26 LOCATE 20,i:d$=MID$(a$,i,1):PRINT d$;:t$(20)=a$
27 SOUND INT(4000*RND+500),1
28 NEXT i
29 SCROLL (0,0)-(120,200),0,-9
30 FOR i=1 TO 80*LEN(b$):NEXT i
31 FOR i=0 TO 19:t$(i)=t$(i+1):NEXT i
32 s$=a$:GOTO debut
33 suite:
34 IF d$="1" OR d$="&" THEN H$="JOUEUR DE GAUCHE" ELSE
   H$="JOUEUR DE DROITE"
35 FOR i=0 TO 18
36 IF s$=t$(i) THEN LOCATE i,LEN(s$)+2:PRINT"BRAVO , "
   ;H$;:END
37 NEXT i
38 LOCATE 20,LEN(s$)+5:PRINT"PERDU ";H$;

```

Permutations

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : réflexion.

Difficulté du jeu : moyenne.

LE JEU :

L'Amiga 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 :

Les lignes 5 à 18 permettent de choisir le nombre de lettres et contrôlent que l'on n'en demande pas plus de 12.

La ligne 22 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 23), supprimées dans A\$ et remplacées en C\$, dans le bon ordre (lignes 23 à 25).

La ligne 24 contrôle le nombre de lettres de B\$.

Les affichages sont effectués aux lignes 26 à 30.

Le retournement est fait, lettre par lettre, dans la boucle I (ligne 41) ; le résultat est placé dans B\$ en ligne 42, affiché par la ligne 43 et comparé au résultat final en ligne 44 .

Le nombre d'essais sera affiché en fin de partie, grâce à la ligne 46.

POUR JOUER :

Un nombre de lettres (de 1 à 12) est choisi en cliquant sur + ou - avec la souris, puis sur F pour valider. L'affichage suivant présente l'ordre initial (DEBUT) et l'ordre à venir (FIN). Les différentes permutations possibles sont affichées dans des carrés. On clique sur la permutation à réaliser à l'aide du bouton de gauche. On obtient alors le résultat, et l'on recommence jusqu'à obtenir l'ordre correct. On répond « O » à la question « AUTRE PARTIE » si l'on veut rejouer.

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 voisines de la lettre la plus à gauche, etc.).

```

1  REM PERMUTATIONS
2  SCREEN 1,320,200,4,1:WINDOW 1,"PERMUTATIONS",,2,1
3  RANDOMIZE TIMER:CLS:n=6
4  WINDOW 4,"NIVEAU", (30,60)-(285,125),16,1:WINDOW OUT
   PUT 4
5  LOCATE 2,1:PRINT "NOMBRE DE LETTRES : "
6  LOCATE 4,1:PRINT "CLIQUEZ SUR +, - ou F"
7  LOCATE 6,8:COLOR 3,11:PRINT "-":LOCATE 6,12:PRINT "
   +"
8  LOCATE 6,16:PRINT "F"
9  test:
10 LOCATE 2,20:COLOR 7,0:PRINT n
11 WHILE MOUSE(0)=0:WEND
12 WHILE MOUSE(0)<>0 OR y<>6 OR (x<>8 AND x<>12 AND x<
   >16)
13 x=INT(MOUSE(5)/10)+1:y=INT(MOUSE(6)/9)+1
14 WEND
15 IF x<>16 THEN
16 IF x=8 THEN n=n+(n>2):GOTO test
17 IF x=12 THEN n=n-(n<12):GOTO test
18 END IF
19 a$="ABCDEFGHijkl":c$=SPACE$(12):b$=""
20 WINDOW CLOSE 4:WINDOW OUTPUT 1:COLOR 2,1:CLS
21 tirage:
22 x=INT(RND*n+1):IF MID$(a$,x,1)=" " THEN tirage
23 b$=MID$(a$,x,1)+b$:MID$(c$,x,1)=MID$(a$,x,1)
24 MID$(a$,x,1)=" ":s=1:IF LEN(b$)<>n THEN tirage
25 c$=LEFT$(c$,LEN(b$))
26 LOCATE 5,1:PRINT"DEBUT  ":;:COLOR 3:PRINT b$:COLOR
   2
27 LOCATE 7,1:PRINT"FIN  ":;:COLOR 4:PRINT c$:COLOR
   2
28 debut:
29 LOCATE 11,1:PRINT"PERMUTATION NUMERO ";s
30 LOCATE 13,1:PRINT "CLIQUEZ LE NOMBRE A PERMUTER"
31 FOR i=1 TO n
32 LINE(20*i-4,120)-(20*i+14,156+18*(n<10)),9,bf:r$=ST
   R$(i)
33 COLOR 4,9:LOCATE 15,2*i+1:PRINT MID$(STR$(i),2,1)
34 LOCATE 16,2*i+1:IF LEN(r$)>2 THEN PRINT MID$(r$,3,1
   )
35 NEXT i
36 WHILE MOUSE(0)=0:WEND
37 WHILE MOUSE(0)<>0 OR 1<1 OR 1>n OR (y<>1 AND y<>2)
38 l=INT((MOUSE(5)+2)/20):y=INT((MOUSE(6)-103)/18)
39 WEND
40 s=s+1:a$=""
41 FOR i=1 TO l:a$=a$+MID$(b$,l-i+1,1):NEXT i
42 b$=a$+RIGHT$(b$,LEN(b$)-1):SOUND 400,2:COLOR 13,1
43 LOCATE 19,1:PRINT"RESULTAT ":;:COLOR 8:PRINT b$
44 IF b$<>c$ THEN debut
45 FOR i=100 TO 1200 STEP 15:SOUND i,2:NEXT i
46 PRINT"TROUVE EN ";s-1;" ESSAIS";
47 PRINT:PRINT"AUTRE PARTIE ( O/N )":r$=UCASE$(INPUT$(
   1))
48 IF r$="O" THEN RUN

```

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 (7-10) dessine le terrain de jeu, et les lignes 12 et 13 s'occupent de la gestion du tour des joueurs (numéro 1 ou 2 pour chaque joueur, et jeton O ou X correspondant).

La boucle 16-18 attend le coup choisi en cliquant la case jouée avec la souris.

Les lignes 20 et 21 vérifient la validité du coup.

La ligne 11 teste la fin de partie, en cas de match nul, et la ligne 19 refuse les coups non conformes (avec retour à la situation initiale, ligne 15).

La ligne 30 s'occupe de l'affichage du coup, et la ligne 32 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 33 à la ligne 40 : on utilise trois boucles emboîtées pour vérifier si l'un des joueurs a obtenu, avec trois pions, un total de 15. Le changement de joueur s'effectue en ligne 42.

POUR JOUER :

Lancer le jeu en tapant RUN.

Le terrain de jeu s'affiche, puis la question JOUEUR 1 : X, et le joueur qui a choisi les X devra cliquer avec le bouton de gauche le numéro de la case où il désire jouer. Si elle est libre, son coup y sera enregistré.

On obtient, ensuite, l'affichage JOUEUR 2 : 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 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 15
2  SCREEN 1,320,200,4,1:WINDOW 1,"QUINZE",,2,1
3  RANDOMIZE TIMER:CLS:s$(0)="0":s$(1)="X"
4  f=INT(RND*2):r$="123456789":o$="":x$=""
5  LINE (10,7)-(300,40),2,bf
6  LINE (10,41)-(300,60),13,bf
7  FOR i=0 TO 24 STEP 3
8  LINE (10*i+25,12)-(10*i+45,35),9,b
9  COLOR 1,13:LOCATE 6,i+4:PRINT CHR$(49+INT(i/3))
10 NEXT i
11 WHILE VAL(r$)<>0
12 COLOR 9,0:LOCATE 11,1:PRINT "TOUR DU JOUEUR ";f+1;"
   : ";
13 COLOR f+3:PRINT s$(f)
14 COLOR 7:LOCATE 13,1:PRINT "CLIQUEZ SUR LA CASE CHOI
   SIE"
15 tirage:
16 WHILE ABS(MOUSE(0))<2
17 x=MOUSE(5):y=MOUSE(6):x=INT((x+4)/5):a=x MOD 6:b=IN
   T(x/6)
18 WEND
19 IF y<13 OR y>36 OR a>3 OR x<6 THEN tirage
20 a$=RIGHT$(STR$(b),1):i=INSTR(r$,a$)
21 IF i<>0 THEN SOUND 800,4::LOCATE 6,3*(b-1)+4:COLOR
   f+3,13:PRINT a$:GOSUB inscrit
22 WEND
23 LOCATE 17,1:PRINT "MATCH NUL.....":GOTO fin
24 gagne:
25 LOCATE 17,1:PRINT "BRAVO JOUEUR ";s$(f)
26 fin:
27 FOR i=100 TO 1000 STEP 10:SOUND i,1:NEXT i
28 END
29 inscrit:
30 COLOR f+3,2:LOCATE 3,3*(b-1)+4:PRINT s$(f)
31 IF f=1 THEN o$=o$+a$ ELSE x$=x$+a$
32 MID$(r$,i,1)="0":IF f=1 THEN t$=o$ ELSE t$=x$
33 IF LEN(t$)<3 THEN change
34 FOR i=1 TO 3
35 IF i=LEN(t$) THEN change
36 FOR j=i+1 TO LEN(t$)
37 IF j=LEN(t$) THEN change
38 FOR k=j+1 TO LEN(t$)
39 IF VAL(MID$(t$,i,1))+VAL(MID$(t$,j,1))+VAL(MID$(t$,
   k,1))=15 THEN gagne
40 NEXT k,j,i
41 change:
42 f=ABS(f-1):SOUND 200,2:RETURN

```

Dessin de cercles

REFERENCES :

Difficulté de programmation : facile.

Catégorie de jeu : dessin.

Difficulté du jeu : aucune.

LE JEU :

On déplace le curseur de la souris sur l'écran pour tracer des cercles pleins avec la taille et la couleur choisies.

LE PROGRAMME :

Les lignes 6 et 7 affichent la taille et la couleur en cours en haut de l'écran.

La ligne 9 appelle la procédure taille.

La ligne 10 appelle la procédure couleur.

La ligne 13 contrôle la position du curseur et effectue le dessin lorsque le bouton de gauche est cliqué.

Les procédures sont dans les lignes 22 et 23.

POUR JOUER :

Lancer le programme par RUN.

On choisit la taille en appuyant sur T, et la couleur en appuyant sur C.

Les cercles sont tracés en appuyant sur le bouton de gauche de la souris.

On quitte le programme en tapant Ctrl et C.

EXTENSIONS POSSIBLES :

On peut envisager de tracer des cercles (et non des disques), prévoir d'autres figures (rectangles, etc.).

```

1  REM DESSIN DE CERCLES
2  SCREEN 1,640,210,4,1:WINDOW 1,"CERCLES",,2,1
3  RANDOMIZE TIMER:t=1:co=2:COLOR 2,1:CLS
4  boucle:
5  WHILE MOUSE(0)=0
6  COLOR 2,1:LOCATE 1,1:PRINT "TAILLE :";t
7  LOCATE 1,15:PRINT "COULEUR :";co
8  r%=UCASE$(INKEY%)
9  IF r%="T" THEN CALL taille (t)
10 IF r%="C" THEN CALL couleur (co)
11 WEND
12 WHILE MOUSE(0)<>0
13 a=MOUSE(5):b=MOUSE(6):CIRCLE(a,b),t,co
14 PAINT (a,b),co,co
15 COLOR 2,1:LOCATE 1,1:PRINT "TAILLE :";t
16 LOCATE 1,15:PRINT "COULEUR :";co
17 r%=UCASE$(INKEY%)
18 IF r%="T" THEN CALL taille (t)
19 IF r%="C" THEN CALL couleur (co)
20 WEND
21 GOTO boucle
22 SUB taille (t) STATIC
23 t=t+1+t*(t>63)
24 END SUB
25 SUB couleur (co) STATIC
26 co=co+1+co*(co>14)
27 END SUB

```

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 :

Les lignes 3 à 10 vous permettent de choisir la taille de l'échiquier en cliquant avec la souris sur l'un des chiffres affichés.

Une variable D sera dimensionnée en conséquence, en ligne 12.

Une boucle d'exploration principale I (13-20) contient une sous-boucle J (17-19) 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 34 à 39 et on recommence.

La boucle F (22-25) assure l'impression de l'échiquier, et la boucle F (26-31), 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 33 permet un arrêt après l'affichage de chacune des solutions.

POUR JOUER :

On lance le programme avec RUN, puis on clique avec le bouton de gauche la dimension de l'échiquier désirée.

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 CTRL et Break.

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  SCREEN 1,320,200,4,1:WINDOW 1,"REINES SUR L'ECHIQUE",,2,1:CLS
3  WINDOW 4," CLIQUEZ LA TAILLE", (55,60)-(235,90),16,1
4  WINDOW OUTPUT 4
5  FOR i=4 TO 9
6  LINE(30*i-120,0)-(30*i-90,30),i,bf:LOCATE 2,3*i-10:
  PRINT CHR$(48+i);
7  NEXT i
8  WHILE ABS(MOUSE(0))<2 OR b<>1 OR ( a MOD 3 )<>1
9  a=INT(MOUSE(5)/10):b=INT(MOUSE(6)/9)
10 WEND
11 WINDOW CLOSE 4:WINDOW OUTPUT 1
12 s=INT((a-1)/3)+4:DIM d(s)
13 i=1:k=1
14 debut:
15 d(i)=1
16 suite:
17 FOR j=1 TO i-1
18 IF d(i)=d(j) OR (ABS(d(i)-d(j))=i-j) THEN progres
19 NEXT j
20 i=i+1:IF i<=s THEN debut
21 COLOR ,0:CLS:LINE (0,0)-(32*s+26,18*s+20),8,bf
22 FOR f=1 TO s
23 FOR j=1 TO s
24 LINE (32*f-20,18*j-9)-(32*f+12,18*j+9),2*((f+j)MOD
  2)+1,bf
25 NEXT j,f
26 FOR f=1 TO s
27 COLOR 1,8:LOCATE 1,3*f+1:PRINT CHR$(48+f)
28 LOCATE 2*f+1,1:PRINT CHR$(64+f)
29 CIRCLE (32*d(f)-4,18*f),8,4:PAINT (32*d(f)-4,18*f),
  11,4
30 LOCATE 20,f:COLOR 15:PRINT RIGHT$(STR$(d(f)),1);
31 NEXT f
32 COLOR 5:LOCATE 20,10:PRINT"APPUYEZ SUR UNE TOUCHE";
33 R$=INPUT$(1)
34 i=i-1
35 progres:
36 d(i)=d(i)+1
37 IF d(i)<=s THEN suite
38 i=i-1
39 IF i<>0 THEN progres
40 PRINT:PRINT "C'est fini..."

```

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 :

Le sous-programme (42-50) est appelé trois fois par les lignes 5 à 10 pour choisir les couleurs avec la souris.

Les deux boucles emboîtées I (11-18) et J (13-18) dessinent le récipient et les obstacles.

La ligne 24 dessine la goutte. Les tests des lignes 28 et 30 déterminent s'il y a ou non rencontre avec un obstacle.

Si c'est non, on va en 33, et la goutte continue de tomber.

Si c'est oui, la goutte part à droite ou à gauche (ligne 36) selon la valeur définie pour K en ligne 22.

On regarde de nouveau s'il y a ou non obstacle, ou si la goutte peut continuer à tomber (lignes 21 à 30).

Selon le résultat du test, on utilisera 33, 35 ou 37.

POUR JOUER :

Il est possible de choisir, à l'aide de la souris, les couleurs du récipient, des obstacles, et de la goutte qui tombe. On devra pour cela cliquer avec la touche sur la case contenant la couleur choisie. Le programme débute ensuite automatiquement.

La goutte apparaît, se met à tomber et finit par se trouver une place. Une autre goutte apparaît et l'on recommence.

Le programme s'arrête de lui-même lorsque le récipient déborde ; on peut l'arrêter à tout moment en tapant CTRL et Break.

EXTENSIONS POSSIBLES :

On peut, bien entendu, ajouter 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  SCREEN 1,320,200,4,1:WINDOW 1,"GOUTTES",,2,1
3  RANDOMIZE TIMER:CLS:a=0:DIM t(20,20)
4  FOR i=1 TO 15: LINE (21*i-22,20)-(21*i-1,40),i,bf:N
   EXT i
5  LOCATE 15,1:PRINT "COULEUR DU RECIPIENT"
6  GOSUB saisie:cr=x:a=0:GOSUB pause
7  LOCATE 15,1:PRINT "COULEUR DES OBSTACLES"
8  GOSUB saisie:co=x:a=0:GOSUB pause
9  LOCATE 15,1:PRINT "COULEUR DE LA BILLE  "
10 GOSUB saisie:cb=x:GOSUB pause:CLS
11 FOR i=11 TO 19
12   t(i,0)=1:t(i,17)=1
13   FOR j=1 TO 16
14    LOCATE i+1,1:COLOR ,cr:PRINT " ";LOCATE i+1,18:PRI
      NT " ";
15    LOCATE 20,j+1:PRINT " ";t(19,j)=1
16    IF i<18 THEN LOCATE i+1,j+1:COLOR ,0:PRINT " ";t(i
      ,j)=0
17    IF i>11 AND i<19 AND RND<.4 THEN COLOR ,co:f=INT(RN
      D*16+2):LOCATE i+1,f:PRINT " ";t(i,f-1)=1
18   NEXT j,i
19   debut:
20   y=INT(RND*7+6):x=6
21   tombe:
22   k=-2*(RND<.5)-1:i=0
23   affiche:
24   LOCATE x+1,y+1:COLOR cb,0:PRINT CHR$(2)
25   FOR j=0 TO 100:IF y>18 THEN fin
26   NEXT j:SOUND INT(RND*800+100),1
27   LOCATE x+1,y+1:PRINT " "
28   u=x+1:v=y:j=t(u,v):IF j=0 THEN avance
29   suite:
30   u=x:v=y+k:j=t(u,v):IF j=0 THEN cherche
31   IF i=0 THEN change
32   LOCATE x+1,y+1:COLOR 3:PRINT CHR$(3):t(x,y)=1:GOTO
      debut
33   avance:
34   x=x+1:GOTO tombe
35   cherche:
36   y=y+k:i=1:GOTO affiche
37   change:
38   k=-k:i=1:GOTO suite
39   fin:
40   LOCATE 12,25:PRINT "C'EST FINI..."
41   FOR i=100 TO 1200 STEP 10:SOUND i,1:NEXT i:END
42   saisie:
43   WHILE a<2 OR MOUSE(6)<20 OR MOUSE(6)>40
44    x=MOUSE(5):x=INT(x/21)+1:y=MOUSE(6):a=ABS(MOUSE(0))
45   WEND
46   LINE (21*x-22,20)-(21*x-1,40),0,bf
47   FOR i=1 TO 600:NEXT i
48   LINE (21*x-22,40)-(21*x-1,60),x,bf
49   RETURN
50   pause:
51   a=MOUSE(0):FOR i=1 TO 2000:NEXT i:a=MOUSE(0):SOUND
      400,2:RETURN

```

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 la souris sur l'écran, avec la possibilité de marquer des points, tirer des traits, faire des cercles pleins, des rectangles ou des rectangles pleins, selon le mode choisi (choix M).

On peut choisir la couleur du point en appuyant sur E, la couleur du fond sur F, et la taille des figures sur T.

LE PROGRAMME :

Les lignes 4 et 5 permettent de redéfinir les couleurs grâce aux DATAS des lignes 33 à 35.

La ligne 9 affiche une ligne de renseignements.

Les lignes 10 à 15 contrôlent les entrées.

La ligne 18 contrôle la position de la souris.

Les lignes 24 à 28 tracent la figure choisie.

La fin du jeu est en ligne 30.

POUR JOUER :

On lance le programme par RUN.

Une ligne d'indication apparaît en haut de l'écran. Une pression sur E modifie la couleur. F change le fond. T change la taille. M change le mode, avec les règles suivantes :

MODE 1 = trace un point

2 = lignes courbes

3 = cercle (selon la taille)

4 = rectangle plein

5 = trait

6 = rectangle vide.

Pour les modes 4, 5 et 6, on clique sur deux points diagonalement opposés pour faire le tracé.

On quitte le programme en tapant ESPACE.

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  SCREEN 1,320,200,5,1:WINDOW 1,"ARDOISE MAGIQUE",,2,
   1:CLS
3  fo=1:co=1:mo=1:t=1:x=160:y=100:x0=160:y0=100
4  DIM pa(3,16):COLOR 3,4:CLS
5  FOR i=1 TO 16:READ a,b,c:pa(0,i)=a:pa(1,i)=b:pa(2,i)
   )=c:NEXT i
6  debut:
7  WHILE MOUSE(0)=0
8  PALETTE 3,pa(0,co),pa(1,co),pa(2,co):PALETTE 4,pa(0
   ,fo),pa(1,fo),pa(2,fo)
9  LOCATE 1,1:COLOR 2,1:PRINT"ENCRE";co;"FOND";fo;"MOD
   E";mo;"TAILLE";t:COLOR 3,4
10 r$=UCASE$(INKEY$)
11 IF r$="E" THEN co=co+1+co*(co>14)
12 IF r$="F" THEN fo=fo+1+fo*(fo>14)
13 IF r$="M" THEN mo=mo+1+mo*(mo>5)
14 IF r$="T" THEN t=t+1+t*(t>63)
15 IF r$=" " THEN fin
16 WEND
17 WHILE MOUSE(0)<>0
18 x=MOUSE(1):y=MOUSE(2):a=MOUSE(3):b=MOUSE(4):PSET(x,
   y),co
19 IF mo=1 THEN PSET(MOUSE(5),MOUSE(6)),co
20 FOR i=1 TO 100:NEXT i
21 IF mo>1 THEN PRESET(x,y)
22 IF mo>3 THEN PSET(a,b),co
23 WEND
24 IF mo=2 THEN PSET(x,y),co
25 IF mo=3 THEN CIRCLE(x,y),t,co:PAINT(x,y),co,co
26 IF mo=4 THEN LINE(x,y)-(a,b),co,bf
27 IF mo=5 THEN LINE(x,y)-(a,b),co
28 IF mo=6 THEN LINE(x,y)-(a,b),co,b
29 GOTO debut
30 fin:
31 FOR i=100 TO 1000 STEP 15:SOUND i,1:NEXT i
32 END
33 DATA 1,1,.13,1,.6,.67,.93,.2,0,.73,1,0,.8,.6,.53,.1
   ,.82,.24
34 DATA 0,.93,.87,.73,.73,.73,0,0,0,1,1,1,.23,.23,.53
35 DATA 0,0.4,0.2,1,0.27,0.45,.2,.56,.2,0,0,.26,.96,.2
   7,.96

```

Tic tac toc

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 (ici un cœur ou un carreau) 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 des boucles I et J emboîtées (4-9).

La ligne 12 s'occupe d'afficher le symbole X ou O correspondant au joueur qui a le trait.

Le joueur devra cliquer avec le bouton de gauche sur la case où il veut jouer (lignes 16 à 19).

Les lignes 16 et 18 vérifient si la case où l'on veut jouer est libre, et si l'on désire jouer.

Les lignes 22 et 24 réalisent le dessin correspondant.

La ligne 27 réalise le changement de joueur.

POUR JOUER :

Le terrain de jeu sera affiché dès la frappe de RUN.

Chaque joueur devra à son tour cliquer avec le bouton de gauche de la souris sur la case où il désire poser son pion (ce n'est possible que si elle est libre). Le trait passe alors automatiquement au joueur suivant, qui procède de même. Le jeu se termine si toutes les cases sont occupées, ou si l'on tape CTRL et C.

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  SCREEN 1,320,200,4,1:WINDOW 1,"TIC-TAC-TOE",,2,1
3  RANDOMIZE TIMER:CLS:b$(0)="X":b$(1)="O"
4  FOR i=1 TO 3
5  FOR j=1 TO 3
6  LINE (60*i-40,50*j-40)-(60*i+18,50*j+8),3,b
7  LINE (60*i-26,50*j-29)-(60*i+4,50*j-4),3,b
8  PAINT (60*i-35,50*j-35),3,3:PAINT (60*i,50*j-15),11
  ,3
9  NEXT j,i
10 k=INT(RND*2):s=0
11 debut:
12 co=0:LOCATE 20,1:COLOR 7:PRINT"JOUEUR ";;COLOR 5*k+
  8:PRINT b$(k);
13 WHILE MOUSE(0)=0:r$=UCASE$(INKEY$):IF r$="F" THEN E
  ND
14 IF r$=" " THEN k=1-k:GOTO debut
15 WEND
16 WHILE MOUSE(0)<>0 OR co<>11
17 x=MOUSE(5):y=MOUSE(6):a=INT((x-6)/30):b=INT((y+2)/2
  5)
18 co=POINT(30*a+20,25*b+8)
19 WEND
20 IF (a MOD 2)<>1 OR (b MOD 2)<>1 OR a>5 OR b>5 THEN
  debut
21 IF k=1 THEN
22 CIRCLE(30*a+20,25*b+8),8,5*k+8:PAINT (30*a+20,25*b+
  8),5*k+8,5*k+8
23 ELSE
24 AREA(30*a+10,25*b+8):AREA(30*a+20,25*b):AREA(30*a+3
  0,25*b+8):AREA(30*a+20,25*b+16)
25 COLOR 5*k+8:AREAFILL 0
26 END IF
27 s=s+1:k=ABS(k-1):SOUND 200+500*k,3
28 IF s<>9 THEN debut
29 FOR i=100 TO 1000 STEP 10:SOUND i,2:NEXT i

```

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 :

La variable alphanumérique A\$, en ligne 3, contient le nom du joueur.

La boucle I (5-9) dessine le terrain de jeu, et les positions de départ.

La ligne 11 permet d'afficher O ou X, selon le joueur qui a le trait.

La boucle (14-17) permet le choix du pion à déplacer, qui apparaît en gris.

La boucle (22-25) permet le choix de la case d'arrivée avec la souris.

La ligne 26 regarde si l'on a choisi l'une des cases autorisées.

La ligne 27 réimprime un point à l'emplacement de départ.

Le résultat est imprimé par la ligne 30, le changement de joueur est réalisé par la ligne 32.

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. Celui-ci doit alors cliquer avec le bouton de gauche de la souris sur la case de départ (contenant un de ses pions), puis, avec le même bouton, il clique sur la case d'arrivée (vide).

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 tapant CTRL et Break.

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 PC de participer au jeu avec de bonnes chances de l'emporter.

```

1  REM QUATRE A QUATRE
2  SCREEN 1,320,210,5,1:WINDOW 1,"QUATRE A QUATRE",,2,
   1:CLS
3  RANDOMIZE TIMER:h=INT(RND*2):a$(0)="JOUER 0":a$(1)
   ="JOUER X"
4  LINE(0,0)-(186,120),11,bf
5  FOR i=0 TO 5
6  FOR j=0 TO 5
7  t(i,j)=-33*(i<2 AND j<2)-42*(i>3 AND j>3)+46
8  LOCATE 2*i+2,3*j+2:COLOR INT(t(i,j)/10),5:PRINT CHR
   $(t(i,j))
9  NEXT j,i
10 debut:
11 LOCATE 16,1:COLOR 7+h:PRINT a$(h)
12 LOCATE 18,1:COLOR 2:PRINT "CLIQUEZ LE PION"
13 WHILE MOUSE(0)=0:WEND
14 WHILE MOUSE(0)<>0 OR (i MOD 3)<>1 OR (j MOD 2)<>1 O
   R i>17 OR j>12
15 x=MOUSE(3):y=MOUSE(4):i=INT((x-1)/10):j=INT((y-1)/9
   )
16 a=INT((i-1)/3):b=INT((j-1)/2)
17 WEND
18 IF t(a,b)<>79-9*(h=1) THEN debut
19 LOCATE 2*b+2,3*a+2:COLOR 15:PRINT CHR$(t(a,b))
20 COLOR 2:LOCATE 18,1:PRINT "CLIQUEZ LA CASE"
21 WHILE MOUSE(0)=0:WEND
22 WHILE MOUSE(0)<>0
23 x=INT((MOUSE(3)-1)/10):y=INT((MOUSE(4)-1)/9)
24 u=INT((x-1)/3):v=INT((y-1)/2)
25 WEND
26 IF t(u,v)<>46 OR NOT((ABS(a-u)=1 AND ABS(b-v)=2) OR
   (ABS(a-u)=2 AND ABS(b-v)=1)) THEN
27 LOCATE 2*b+2,3*a+2:COLOR 7+h:PRINT CHR$(t(a,b)):GOT
   O debut
28 ELSE
29 LOCATE 2*b+2,3*a+2:COLOR 4:PRINT CHR$(46):t(a,b)=46
30 LOCATE 2*v+2,3*u+2:COLOR 7+h:PRINT CHR$(79+9*h):t(u
   ,v)=79+9*h
31 END IF
32 h=ABS(h-1):i=1:SOUND 200+500*h,3
33 GOTO debut

```

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 colorée vous permet de déplacer les lettres, et un compteur affiche en permanence le nombre d'essais : arriverez-vous à retrouver le proverbe ?

LE PROGRAMME :

Les lignes 28 à 33 impriment le terrain de jeu (sous-programme affiche appelé par la boucle I (10-13)).

La phrase à reconstituer (qui pourra bien sûr être modifiée) est placée dans une variable R\$ en ligne 4.

Le mélange des lettres et l'impression sont faits par la boucle I (5-9).

Le choix de la lettre à déplacer est fait à l'aide de la souris (lignes 15 à 20).

Les lignes 22 à 25 font glisser la série de lettres concernées jusqu'à la case vide.

La ligne 26 comptabilise le nombre d'essais et l'imprime.

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 plusieurs lettres situées sur la même ligne ou sur la même colonne que la case libre. Il suffit pour cela de venir cliquer avec le bouton de gauche de la souris sur la case que l'on désire voir se libérer. Toutes les lettres de la ligne ou de la colonne correspondante vont alors glisser jusqu'à occuper l'ancienne case vide.

Le nombre d'essais est affiché en permanence, en dessous du puzzle.

On peut arrêter à tout moment en tapant CTRL et Break.

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  SCREEN 1,320,210,4,1:WINDOW 1,"TAQUIN DE LETTRES",,
  2,1
3  RANDOMIZE TIMER:CLS:s=0:DIM t(10,10)
4  a$="":r$="DNASOUVENTBESOIND'UNPLUSPETITQUESOI "
5  FOR i=0 TO 35
6  x=INT(RND*LEN(r$)+1):a$=a$+MID$(r$,x,1):f$=LEFT$(r$,
  x-1)
7  r$=f$+MID$(r$,x+1):t((i MOD 6)+1,INT(i/6)+1)=ASC(MI
  D$(a$,i+1,1))
8  IF ASC(MID$(a$,i+1,1))=32 THEN u=(i MOD 6)+1:v=INT(
  i/6)+1
9  NEXT i
10 FOR i=1 TO 6
11 FOR j=1 TO 6
12 GOSUB affiche
13 NEXT j,i
14 test:
15 LOCATE 15,1:COLOR 2:PRINT "CLIQUEZ LA LETTRE"
16 WHILE MOUSE(0)=0:WEND
17 WHILE MOUSE(0)<>0
18 a=INT(MOUSE(3)/10):b=INT(MOUSE(4)/9)
19 x=INT(a/3):y=INT(b/2)
20 WEND
21 IF (x<>u AND y<>v) OR (a MOD 3)<>0 OR (b MOD 2)<>0
  OR a>18 OR b>12 OR b<2 THEN SOUND 800,4:GOTO test
22 IF u=x AND v<y THEN FOR f=v TO y-1:t(u,f)=t(u,f+1):
  i=u:j=f:GOSUB affiche:NEXT f
23 IF u=x AND y<v THEN FOR f=v TO y+1 STEP-1:t(u,f)=t(
  u,f-1):i=u:j=f:GOSUB affiche:NEXT f
24 IF v=y AND u<x THEN FOR f=u TO x-1:t(f,v)=t(f+1,v):
  i=f:j=v:GOSUB affiche:NEXT f
25 IF v=y AND x<u THEN FOR f=u TO x+1 STEP-1:t(f,v)=t(
  f-1,v):i=f:j=v:GOSUB affiche:NEXT f
26 t(x,y)=32:u=x:v=y:i=x:j=y:GOSUB affiche:s=s+1:LOCAT
  E 5,23:PRINT "COUP";s
27 GOTO test
28 affiche:
29 LINE(30*i-12,18*j-6)-(30*i+18,18*j+12),3,bf
30 LINE(30*i-9,18*j-4)-(30*i+15,18*j+10),11,bf
31 COLOR 8,11:LOCATE 2*j+1,3*i+1:PRINT CHR$(t(i,j))
32 SOUND INT(RND*1000+100),1
33 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 ; par contre, ses déplacements sont effectués au hasard. Arrivera-t-il à parcourir les 64 cases ?

LE PROGRAMME :

La ligne 4 définit les divers déplacements possibles, dans deux variables alphanumériques A\$ et B\$; la boucle I (6-10) dessine le terrain de jeu.

On entre les coordonnées de la case de départ (ligne et colonne) en cliquant avec la souris sur la case concernée (lignes 11 à 16).

La ligne 19 imprime le numéro du coup.

La ligne 21 permet le tirage au hasard d'un déplacement du cavalier.

La boucle Z (22-25) cherche si le déplacement est possible ; si ce n'est pas le cas, on essaie tous les autres (toujours dans le même ordre).

La ligne 24 contrôle que l'on ne sorte pas du terrain de jeu.

La ligne 24 envoie en ligne 28 si le coup est possible.

La ligne 26 gère la fin de partie (cavalier bloqué) et affiche le score.

Les lignes 29 à 32 réalisent le coup et incrémentent le score.

POUR JOUER :

On lance le programme par l'ordre RUN. Le terrain de jeu s'affiche. Il est fait appel à la souris de l'Amiga pour choisir la ligne et la colonne de départ : il suffit après avoir lancé l'exécution du programme par RUN, de cliquer avec la touche de gauche sur la case de départ choisie. Le cavalier va ensuite parcourir l'échiquier automatiquement, jusqu'au blocage par impossibilité de jouer en respectant les règles. On peut, bien sûr, arrêter à tout moment par CTRL et Break.

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  SCREEN 1,320,220,4,1:WINDOW 1,"PARCOURS DU CAVALIER
   ",,2,1
3  RANDOMIZE TIMER:CLS:s=1:t=0:DIM t(8,8)
4  a$="56653223":b$="653223"+a$+"56":a$=a$+a$
5  LINE(14,8)-(256,154),11,bf
6  FOR i=1 TO 8
7  LOCATE 1,3*i:PRINT CHR$(48+i):LOCATE 2*i+1,1:PRINT
   CHR$(64+i);
8  FOR j=1 TO 8
9  LINE(30*i-14,18*j-8)-(30*i+14,18*j+8),3*((i+j) MOD
   2)+5,bf
10 NEXT j,i
11 LOCATE 19,1:PRINT"CLIQUEZ LA CASE DE DEPART";
12 WHILE MOUSE(0)=0:WEND
13 WHILE MOUSE(0)<>0
14 x=MOUSE(3):y=MOUSE(4):a=INT((x+14)/30):b=INT((y+8)/
   18)
15 LOCATE 20,1:PRINT "LIGNE ";CHR$(64+b);:LOCATE 20,15
   :PRINT "COLONNE ";CHR$(48+a);
16 WEND
17 debut:
18 i=a:j=b:IF s>9 THEN s=0:t=t+1
19 LOCATE 2*j+1,3*i+1:COLOR 10,3*((i+j) MOD 2)+5:PRINT
   RIGHT$(STR$(s),1)
20 IF t<>0 THEN LOCATE 2*j+1,3*i:PRINT RIGHT$(STR$(t),
   1)
21 z=INT(RND*8+1):t(i,j)=1
22 FOR z=z TO z+8
23 a=i+VAL(MID$(a$,z,1))-4:b=j+VAL(MID$(b$,z,1))-4
24 IF a>0 AND b>0 AND a<9 AND b<9 THEN IF t(a,b)=0 THE
   N suite
25 NEXT z
26 COLOR 7,0:LOCATE 19,1:PRINT"TOTAL :";t*10+s;SPACE$(
   15);
27 FOR i=100 TO 1000 STEP 15:SOUND i,1:NEXT i:END
28 suite:
29 i=a:j=b
30 s=s+1:SOUND INT(RND*1000+200),1
31 FOR k=1 TO 500:NEXT k
32 GOTO debut

```

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 entrant ses coordonnées dans l'ordre ligne-colonne.

LE PROGRAMME :

La boucle I (9-11) imprime les chiffres de repérage des grilles sur les deux écrans définis par J (7-11).

La boucle principale I (14-18) imprime les symboles des deux grilles, dans un ordre différent sur la grille non visible.

La ligne 19 choisit le caractère à modifier.

Le sous-programme (34-39) s'occupe de faire l'impression à un emplacement libre (on l'appelle deux fois pour les deux grilles).

Les coordonnées du point essayé sont entrées aux lignes 23 à 30, à l'aide de la souris.

Les lignes 31 et 32 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).

Pour choisir la lettre à découvrir, il suffit de cliquer à l'aide du bouton de gauche de la souris, sur la case contenant la lettre choisie, dans la grille de droite. La ligne et la colonne sont affichées en permanence.

Le programme répondra alors BRAVO ou PERDU en donnant dans tous les cas la bonne réponse.

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  SCREEN 1,320,200,4,1:RANDOMIZE TIMER
3  WINDOW 1,"GRILLE 1",(0,0)-(150,115),2,1
4  WINDOW 2,"GRILLE 2",(145,0)-(305,115),2,1
5  WINDOW 3,"ANOMALIE",(0,130)-(305,185),2,1
6  t=5:s=t:PALETTE 0,0,0,0:CLS:PALETTE 6,0,0,0
7  FOR j=1 TO 2
8  WINDOW OUTPUT j:COLOR 2*j+1
9  FOR i=1 TO 5
10 LOCATE 1,2*i:PRINT i:LOCATE 2*i+1,1:PRINT CHR$(48+i
    )
11 NEXT i,j
12 WINDOW OUTPUT 3:DIM t(6,6,2)
13 COLOR 9:LOCATE 2,7:PRINT "ATTENDEZ UN INSTANT"
14 FOR i=1 TO 22
15 d=INT(26*RND+1)
16 FOR j=1 TO 2
17 WINDOW OUTPUT j:COLOR 14-4*j:GOSUB tir
18 NEXT j,i
19 j=2:s=0:GOSUB tir:PALETTE 6,.2,.7,.5
20 WINDOW OUTPUT 3:COLOR 11,13:CLS
21 PRINT "CLIQUEZ LA LETTRE"
22 WINDOW OUTPUT 2
23 WHILE ABS(MOUSE(0))<2 OR x<0 OR y<0 OR x>6 OR y>6
24 x=MOUSE(3):y=MOUSE(4):x=INT(x/20):y=INT(y/18)
25 IF x>0 AND y>0 AND x<7 AND y<7 THEN
26 c=t(y,x,2)
27 WINDOW OUTPUT 3:LOCATE 3,1:PRINT"LIGNE";y
28 LOCATE 3,10:PRINT"COLONNE";x:LOCATE 3,20:PRINT "LET
    TRE :";CHR$(c+64)
29 END IF
30 WEND
31 IF d=c THEN PRINT "BRAVO..." ELSE PRINT"PERDU"
32 PRINT"LA LETTRE A TROUVER ETAIT :";CHR$(d+64)
33 FOR i=100 TO 1000 STEP 10:SOUND i,1:NEXT i:END
34 tir:
35 u=INT(RND*t+1):v=INT(RND*t+1)
36 e=t(u,v,j)
37 IF (e<>0 AND s<>0) OR (e+s=0) OR e=d THEN tir ELSE
    t(u,v,j)=d
38 LOCATE 2*u+1,2*v+1:PRINT CHR$(d+64):SOUND INT(RND*1
    000+200),1
39 RETURN

```

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 6, pour recevoir les numéros des cases déjà visitées.

La boucle I (7-10) dessine le terrain de jeu.

La position initiale du locataire est tirée au hasard en ligne 11.

Le numéro de la case à visiter est donné en cliquant avec la souris (lignes 15 à 18).

Les lignes 20 à 28 contrôlent les diverses réponses possibles.

La boucle I (25-27) dessine les cases visitées.

La ligne 31 dessine le locataire, effacé en ligne 35.

Les lignes 37 et 38 assurent son déplacement.

POUR JOUER :

On lance l'exécution par RUN.

A la question CASE VISITEE, on répond en cliquant avec le bouton de gauche sur l'une des lettres 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.

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 SCREEN 1,320,200,4,1:WINDOW 1,"L' INDESIRABLE",,2,1
3 RANDOMIZE TIMER:COLOR ,0:CLS
4 DEF FNa(i)=40*(i-1-4*INT((i-1)/4))+30
5 DEF FNb(i)=36*INT((i-1)/4)+20
6 DIM b(16):LINE(10,10)-(200,160),9,bf
7 FOR i=1 TO 16
8 LINE(FNa(i),FNb(i))-(FNa(i)+29,FNb(i)+22),3,bf
9 LOCATE 3+4*INT((i-1)/4),4+4*(i-1-4*INT((i-1)/4)):CO
LOR 5:PRINT a$;:COLOR 2:PRINT CHR$(64+i);
10 NEXT i
11 x=INT(RND*16+1):s=1:k=0
12 debut:
13 FOR i=1 TO 2000:NEXT i:SOUND 400,4
14 a=1:b=1:LOCATE 19,1:PRINT SPACE$(30):PRINT SPACE$(3
0);
15 WHILE ABS(MOUSE(0))<2 OR a<>0 OR b<>0 OR f>3 OR g>3
16 LOCATE 19,1:COLOR 7:PRINT "CLIQUEZ LA CASE VISITEE
:";
17 a=INT((MOUSE(5)/10))-3:f=a/4:a=a MOD 4:b=INT((MOUSE
(6)/9))-2:g=b/4:b=b MOD 4
18 WEND
19 c=f+4*g+1:PRINT CHR$(64+c);
20 IF b(c)=1 THEN LOCATE 20,1:PRINT"DEJA VISITEE";:GOT
O debut
21 IF c=x THEN LOCATE 20,1:PRINT"TROUVE EN ";s;" COUPS
. BRAVO ";:k=1:GOTO fin
22 IF x-5=c OR x-4=c OR x-3=c OR x-1=c OR x+1=c OR x+3
=c OR x+4=c OR x+5=c THEN GOTO fin
23 LOCATE 20,1:PRINT"IL N'EST PAS LA";
24 suite:
25 s=s+1:b(c)=1:FOR i=1 TO 16
26 LOCATE 4+4*INT((i-1)/4),4+4*(i-1-4*INT((i-1)/4)):CO
LOR ,3+8*b(i):PRINT " "
27 NEXT i:COLOR ,0
28 IF s=9 THEN LOCATE 20,1:PRINT"PERDU . IL ETAIT EN "
;CHR$(x+64);:k=1:GOTO fin
29 GOTO debut
30 fin:
31 LOCATE 4+4*INT((x-1)/4),5+4*(x-1-4*INT((x-1)/4)):CO
LOR 4:PRINT CHR$(2)
32 FOR j=200 TO 1000 STEP 30:SOUND j,1:NEXT j
33 FOR i=1 TO 3000 :NEXT i
34 IF k=1 THEN FOR j=1000 TO 200 STEP-30:SOUND j,1:NEX
T j:END
35 LOCATE 4+4*INT((x-1)/4),5+4*(x-1-4*INT((x-1)/4)):CO
LOR 4:PRINT " "
36 tirage:
37 z=INT(10*RND+1)-5:IF ABS(z)=2 THEN tirage
38 x=x+z:IF x>16 OR x<1 THEN x=x-z:GOTO tirage
39 GOTO suite

```

Ordre

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : jeu de réflexion.

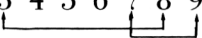
Difficulté du jeu : moyenne.

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 :

Les lignes 5 à 12 permettent de choisir le niveau en cliquant avec la souris sur la case convenable.

Les boucles emboîtées I (14-21) et J (17-19) définissent une variable alphanumérique A\$, contenant les dix premières lettres de l'alphabet dans le désordre.

La ligne 25 affiche A\$, et la ligne 26 affiche les numéros de lettres de l'alphabet pour la permutation.

On entre la permutation en cliquant la case de départ (lignes 34 à 37), puis la case d'arrivée (lignes 40 à 43).

La ligne 44 appelle une première fois le sous-programme d'inversion.

La ligne 45 appelle une deuxième fois le sous-programme d'inversion.

La ligne 46 comptabilise les essais.

La ligne 29 contrôle les résultats.

POUR JOUER :

Lancer le programme par RUN.

On choisit le niveau (de 1 à 3) en cliquant avec la souris.

Le mot initial est affiché, chaque lettre étant repérée par une lettre A, B, C, etc.

Il est fait ici utilisation de la souris pour choisir les lettres à permuter. On devra cliquer sur le bouton de gauche pour choisir la lettre de départ (qui change de couleur) et le cliquer de nouveau pour choisir celle d'arrivée (qui apparaît dans une autre couleur). 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  SCREEN 1,320,200,4,1:WINDOW 1,"ORDRE",,2,1
3  WINDOW 4,"NIVEAU", (120,60)-(210,90),16,1
4  WINDOW OUTPUT 4:RANDOMIZE TIMER:CLS:a$="":s=1
5  FOR i=1 TO 3
6  LINE (30*i-30,0)-(30*i,30),i+2,bf
7  LOCATE 2,3*i-1:PRINT CHR$(i+48)
8  NEXT i
9  WHILE MOUSE(0)=0:WEND
10 WHILE MOUSE(0)<>0 OR b<>1 OR (a MOD 3)<>1
11 a=INT((MOUSE(5)-2)/10):b=INT((MOUSE(6)-2)/9)
12 WEND
13 n=2*INT(a/3)+8:WINDOW CLOSE 4:WINDOW OUTPUT 1
14 FOR i=1 TO n
15 tirage:
16 r$=CHR$(INT(RND*n+65))
17 FOR j=1 TO i-1
18 IF r$=MID$(a$,j,1) THEN tirage
19 NEXT j
20 a$=a$+r$
21 NEXT i
22 debut:
23 p$="R E S U L T A T":co=3:fo=4:x=4:y=1:CALL affiche
(p$,co,fo,x,y)
24 LINE (30,40)-(50+10*LEN(a$),60),11,b
25 co=3:fo=7:x=6:y=4:CALL affiche(a$,co,fo,x,y)
26 p$="":FOR i=1 TO LEN(a$):p$=p$+CHR$(64+i):NEXT i
27 co=5:fo=8:x=8:y=4:CALL affiche(p$,co,fo,x,y)
28 COLOR 9,0:LOCATE 11,1:PRINT"ESSAI NUMERO ";s
29 IF a$=LEFT$("ABCDEFGHIJKLMNOP",LEN(a$)) THEN
30 COLOR 1:PRINT"TROUVE EN ";s;" COUPS"
31 FOR i=100 TO 1200 STEP 10:SOUND i,1:NEXT i:END
32 END IF
33 LOCATE 13,1:PRINT "CLIQUEZ LE DEPART"
34 WHILE MOUSE(0)=0:WEND
35 WHILE MOUSE(0)<>0 OR y<>5 OR i<1 OR i>LEN(a$)
36 i=INT(MOUSE(5)/10)-3:y=INT(MOUSE(6)/9)
37 WEND
38 LOCATE 6,i+4:COLOR 7,8:PRINT MID$(a$,i,1)
39 LOCATE 13,1:PRINT "CLIQUEZ L'ARRIVEE"
40 WHILE MOUSE(0)=0:WEND
41 WHILE MOUSE(0)<>0 OR y<>5 OR j<1 OR j>LEN(a$)
42 j=INT(MOUSE(5)/10)-3:y=INT(MOUSE(6)/9)
43 WEND
44 LOCATE 6,j+4:COLOR 11,13:PRINT MID$(a$,j,1):GOSUB i
nverse
45 IF i<>1 AND i>LEN(a$) THEN i=i-1:j=i+2:GOSUB inver
se
46 s=s+1:GOTO debut
47 inverse:
48 b$=""
49 FOR t=1 TO LEN(a$)
50 IF t=i THEN n$=MID$(a$,t,1)
51 IF t=j THEN m$=MID$(a$,t,1)
52 NEXT t
53 FOR t=1 TO LEN(a$)
54 IF t=i THEN b$=b$+n$:GOTO suite
55 IF t=j THEN b$=b$+m$:GOTO suite
56 b$=b$+MID$(a$,t,1):SOUND INT(RND*1000+100),3
57 suite:
58 NEXT t
59 a$=b$:RETURN
60 SUB affiche(r$,a,b,l,c) STATIC
61 FOR i=1 TO LEN(r$)
62 LOCATE 1,c+i:COLOR a,b:PRINT MID$(r$,i,1):SOUND INT
(RND*800+100),1
63 NEXT i
64 END SUB

```

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 utilisant la souris et les prises sont réalisées automatiquement, si elles sont possibles.

Par contre le changement de joueur étant réalisé à chaque coup, il faudra éventuellement taper la barre d'espace pour pouvoir rejouer s'il y avait eu saut ou si le coup n'était pas possible.

LE PROGRAMME :

Les boucles I (5-11) et J (7-11) dessinent le damier et définissent les positions initiales des pions noirs et des pions blancs. Les positions sont dans une table R (X, Y) et la couleur dans une autre table C (X, Y).

L'impression des pions proprement dite est effectuée par les boucles emboîtées J (13-16) et I (14-16).

Les lignes 18 et 19 indiquent le joueur qui a le trait.

Le coup est entré en cliquant le pion de départ (lignes 21 à 27), puis la case d'arrivée (lignes 29 à 34).

La ligne 22 réalise le changement de joueur (on tape un espace).

Les lignes 36 et 37 regardent si le déplacement est possible.

Le sous-programme (41-44) traite le cas d'un déplacement et le sous-programme (45-49) celui d'un saut.

La ligne 44 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 à côté pour indiquer que le trait appartient au joueur correspondant.

Après l'affichage du damier, le joueur qui commence choisit le pion à déplacer en cliquant avec le bouton de gauche (le pion apparaît de couleur grise), puis il clique sur la case d'arrivée. Le déplacement est alors réalisé (s'il est possible).

Dans le cas d'une prise, il pourra être nécessaire de taper "espace" pour imposer le changement du joueur. On quittera le programme en tapant CTRL et C.

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 à l'Amiga de se mêler avec de bonnes chances de réussite à vos passionnantes parties.


```

1  REM DAMES
2  SCREEN 1,320,220,4,1:WINDOW 1,"DAMES",,2,1
3  DIM r(10,10):DIM c(12,12):co(0)=1:co(1)=9:co(2)=14:
  CLS
4  PRINT"  A  B  C  D  E  F  G  H  I  J"
5  FOR i=1 TO 10
6  LOCATE 2*i,1:PRINT CHR$(47+i);
7  FOR j=1 TO 10
8  LINE (30*j-20,18*i-10)-(30*j+10,18*i+8),((i+j) MOD
  2)+1,BF
9  r(i,j)=0:c(i,j)=0
10 IF j<5 OR j>6 THEN r(i,j)=(i+j+1) MOD 2:c(i,j)=((i+
  j+1) MOD 2)-(j>6)
11 NEXT j,i
12 RANDOMIZE TIMER:h=INT(RND*2)
13 FOR j=1 TO 10
14 FOR i=1 TO 10
15 IF ((i+j)MOD 2)=0 THEN x=3*j+1:y=2*i:r=8:CALL cerc1
  e(x,y,r,co(c(i,j)))
16 NEXT i,j
17 debut:
18 SOUND 400+200*h,2:LOCATE 22,1:COLOR 1:PRINT "JOUEUR
  ";
19 x=10:y=21.5:r=5:CALL cercle(x,y,r,co(h+1))
20 LOCATE 22,12:PRINT "CLIQUEZ LE PION";
21 WHILE MOUSE(0)=0:r$=INKEY$
22 IF r$=" " THEN h=ABS(h-1):GOTO debut
23 WEND
24 WHILE MOUSE(0)<>0 OR c(i,j)<>h+1
25 j=INT((MOUSE(5)-10)/30)+1:i=INT((MOUSE(6)+8)/18)
26 IF c(i,j)=h+1 THEN x=3*j+1:y=2*i:r=8:g=15:CALL cerc
  le(x,y,r,g)
27 WEND
28 LOCATE 22,12:PRINT "CLIQUEZ LA CASE";
29 WHILE MOUSE(0)=0:r$=INKEY$
30 IF r$=" " THEN suite
31 WEND
32 WHILE MOUSE(0)<>0
33 k=INT((MOUSE(5)-10)/30)+1:l=INT((MOUSE(6)+8)/18)
34 WEND
35 t=-(k>j)+(k<j):s=-(l>i)+(l<i)
36 IF r(i+s,j+t)=0 THEN GOSUB avance:GOTO debut
37 IF c(i+s,j+t)=2+(h=1) AND r(i+s,j+t)=1 AND r(i+2*s,
  j+2*t)=0 THEN GOSUB saute:GOTO debut
38 suite:
39 x=3*j+1:y=2*i:r=8:g=co(h+1):CALL cercle(x,y,r,g)
40 GOTO debut
41 avance:
42 r(i,j)=0:r(i+s,j+t)=1:c(i+s,j+t)=2+(h=0):c(i,j)=0
43 x=3*j+1:y=2*i:r=8:g=1:CALL cercle(x,y,r,g):x=3*k+1:
  y=2*l:g=co(h+1):CALL cercle(x,y,r,g)
44 h=ABS(h-1):RETURN
45 saute:
46 c(i,j)=0:r(i,j)=0:r(i+s,j+t)=0:r(i+2*s,j+2*t)=1:c(i
  +s,j+t)=0:c(i+2*s,j+2*t)=2+(h=0)
47 x=3*j+1:y=2*i:r=8:g=1:CALL cercle(x,y,r,g):x=3*(j+t
  )+1:y=2*(i+s):CALL cercle(x,y,r,g)
48 x=3*(j+2*t)+1:y=2*(i+2*s):g=co(h+1):CALL cercle(x,y
  ,r,g)
49 RETURN
50 SUB cercle(a,b,c,d) STATIC
51 CIRCLE (10*a-15,9*b-1),c,d
52 PAINT (10*a-15,9*b-1),d,d
53 END SUB

```

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 lignes 5 à 11 permettent de choisir la taille du damier, avec la souris.

Les boucles emboîtées P (13-30) et Q (19-30) dessinent le terrain de jeu et la ligne 35 affiche un X ou un O, selon le joueur qui commence.

La boucle P (50-64) contrôle la validité du coup (la case est-elle libre : ligne 62), en utilisant une partie de programme qui se trouve aux lignes 67 à 74 (a-t-on pris un ou plusieurs ennemis en ligne 67 et retournement éventuel autorisé par la ligne 72 : on refait un balayage dans toutes les directions, si l'encadrement s'est révélé possible).

Les lignes 53 et 54 calculent les déplacements à effectuer pour explorer les 8 directions.

La ligne 61 réalise les déplacements.

La ligne 74 assure le changement de joueur, s'il y a eu retournement (contrôlé en ligne 73).

POUR JOUER :

Après RUN, on commence par donner la taille du terrain de jeu en cliquant sur l'une des tailles proposées. Puis, le joueur désigné (x ou o) devra prendre la souris et cliquer avec la touche de gauche sur la case où il désire jouer. Le coup est réalisé s'il est possible, et le trait passe alors au joueur suivant. Si aucun coup n'est possible, le changement de joueur se fait en appuyant sur la barre d'espace. On peut aussi arrêter la partie en tapant CTRL et C.

EXTENSIONS POSSIBLES :

Avant tout, bien sûr, améliorer l'affichage avec des couleurs, une grille plus aérée.

On pourra aussi imaginer un jeu machine pour l'Amiga.

```

1 REM OTHELLO
2 SCREEN 1,320,220,4,1:WINDOW 1,"OTHELLO",,2,1
3 WINDOW 4,"CLIQUEZ LA TAILLE", (90,60)-(225,90),16,1
4 WINDOW OUTPUT 4:RANDOMIZE TIMER:CLS
5 FOR i=3 TO 6
6 LINE (30*i-90,0)-(30*i-60-16*(i=6),30),i,bf
7 LOCATE 2,3*i-7:IF i<6 THEN PRINT CHR$(46+2*i) ELSE
PRINT "10"
8 NEXT i
9 WHILE ABS(MOUSE(0))<2 OR b<>1 OR (a MOD 3)<>1
10 a=INT(MOUSE(3)/10):b=INT(MOUSE(4)/9)
11 WEND
12 WINDOW CLOSE 4:WINDOW OUTPUT 1
13 a=2*INT((a-1)/3)+4:DIM t(a+2,a+2):j$(0)="X":j$(1)="
0"
14 FOR i=1 TO a+1
15 LINE (15,18*i-5)-(15+20*a,18*i-5):LINE (20*i-5,13)-(
20*i-5,13+18*a)
16 NEXT i
17 FOR p=1 TO a
18 LOCATE 1,2*p+1:COLOR 2:PRINT RIGHT$(STR$(p),1)
19 FOR q=1 TO a
20 LOCATE 2*p+1,1:COLOR 2:PRINT RIGHT$(STR$(p),1)
21 COLOR 11:LOCATE 2*p+1,2*q+1:PRINT "."
22 IF q=a/2+1 AND p=a/2 THEN
23 LOCATE 2*p+1,2*q-1:COLOR 1:PRINT "0"
24 LOCATE 2*p+1,2*q+1:COLOR 4:PRINT "X"
25 END IF
26 IF q=a/2+1 AND p=a/2+1 THEN
27 LOCATE 2*p+1,2*q-1:COLOR 4:PRINT "X"
28 LOCATE 2*p+1,2*q+1:COLOR 1:PRINT "0"
29 END IF
30 NEXT q,p
31 k=INT(RND*2):so=2:sx=2:t(a/2,a/2)=79:t((a/2)+1,a/2)
=88
32 t(a/2,(a/2)+1)=88:t((a/2)+1,(a/2)+1)=79
33 debut:
34 LOCATE 9,24:COLOR 2:PRINT"JOUEUR"
35 LOCATE 11,27:COLOR 1-3*(k=1):PRINT j$(1-k)
36 LOCATE 13,24:PRINT "CLIQUEZ":LOCATE 14,24:PRINT "LA
CASE"
37 WHILE ABS(MOUSE(0))<2
38 r$=UCASE$(INKEY$):IF r$=" " THEN k=ABS(k-1):GOTO de
but
39 x=INT(MOUSE(5)/10):y=INT(MOUSE(6)/9):u=x:v=y
40 LOCATE 15,24:PRINT "LIGNE";INT(v/2)
41 LOCATE 16,24:PRINT "COL. ";INT(u/2)
42 WEND
43 IF ABS(MOUSE(0))<2 OR (x MOD 2)<>0 OR (y MOD 2)<>0
THEN debut
44 GOSUB change:l=MOUSE(0)
45 LOCATE 5,24:COLOR 5:PRINT sx;"X "
46 LOCATE 6,24:PRINT so;"0 ":SOUND 300+400*k,3
47 IF sx+so<>a THEN debut
48 FOR i=100 TO 1000 STEP 15:SOUND i,1:NEXT i:END
49 change:
50 h=1:p=1:s=0
51 essai:
52 w=0:t=u:r=v
53 f=(p>3 AND p<7) - (p<3 OR p>7)
54 g=(p>1 AND p<5) - (p>=6)
55 q=46-42*(s=1)-33*(s=2)
56 IF s=0 THEN suite
57 so=so+(c=88)-(c=79):sx=sx+(c=79)-(c=88)
58 LOCATE r+1,t+1:c=q:COLOR -1*(k=0)-4*(k=1)
59 PRINT CHR$(q):t(r/2,t/2)=c
60 suite:
61 t=t+2*f:r=r+2*g:c=t(r/2,t/2)
62 IF c=79 OR c=88 THEN tirage
63 boucle:
64 p=p+1:IF p<9 THEN s=0:GOTO essai ELSE IF h<>0 THEN
RETURN
65 so=so-(k=0):sx=sx-(k=1):k=ABS(k-1):RETURN
66 tirage:
67 IF (c=79 AND k=1) OR (c=88 AND k=0) THEN w=1
68 IF s<>0 THEN
69 LOCATE r+1,t+1:COLOR -1*(k=0)-4*(k=1):PRINT CHR$(q)
:t(r/2,t/2)=q
70 so=so-(c=88)+(c=79):sx=sx+(c=88)-(c=79)
71 END IF
72 IF w=1 AND ((c=79 AND k=1) OR (c=88 AND k=0)) THEN
suite
73 IF w=0 OR s<>0 THEN boucle
74 s=1-(k=0):h=-h*(s=0):GOTO essai

```

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 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 des instructions AREA et AREAFILL. La couleur permet de distinguer chaque camp.

LE PROGRAMME :

Les lignes 4 à 7 lisent les DATA contenant les indications de dessin de pièces.

Les lignes 8 à 16 dessinent l'échiquier. Les lignes 14, 15 et 17, 18 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 J (20-23) et I (21-23) dessinent les pièces en appelant une procédure "dessin" en ligne 42.

Les lignes 26 à 30 permettent de choisir la pièce à déplacer avec la souris. Le contrôle est fait en ligne 31.

Les lignes 32 à 36 permettent de choisir la case d'arrivée avec la souris, avec contrôle en ligne 37.

La ligne 40 note la nouvelle position.

La ligne 41 assure le changement de joueur.

POUR JOUER :

On lance l'exécution par RUN.

L'échiquier est dessiné, ainsi que les pièces. Puis le joueur actif est affiché et il choisit la pièce à déplacer en cliquant. La pièce apparaît alors en grisé, et le joueur choisit la case d'arrivée en cliquant de nouveau. Le déplacement est alors effectué et le trait passe au joueur suivant. On peut abandonner à tout moment en tapant CTRL et C.

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 à l'Amiga de se comporter en partenaire honorable. Bon courage !

```

1  REM ECHECS
2  SCREEN 1,320,210,4,1:WINDOW 1,"ECHECS",,2,1
3  DIM r(12,12),c(12,12),t(6,2,15):RANDOMIZE TIMER:COL
  OR 2,1:CLS
4  FOR i=0 TO 5
5  FOR j=1 TO 14
6  READ t(i,1,j),t(i,2,j)
7  NEXT j,i
8  LINE(10,10)-(256,162),11,bf
9  LOCATE 1,1:PRINT " A B C D E F G H"
10 FOR i=1 TO 8
11 LOCATE 2*i+1,1:PRINT CHR$(48+i)
12 FOR j=1 TO 8
13 LINE(30*i-18,18*j-4)-(30*i+12,18*j+14),((i+j) MOD 2
  )+13,bf
14 r(i,j)=6:IF i=2 OR i=7 THEN r(i,j)=0
15 c(i,j)=3:IF i=7 OR i=8 THEN c(i,j)=4
16 NEXT j,i
17 r(1,1)=1:r(1,2)=3:r(1,3)=2:r(1,4)=4:r(1,5)=5:r(1,6)
  =2:r(1,7)=3:r(1,8)=1
18 FOR i=1 TO 8:r(8,i)=r(1,i):NEXT i
19 h=INT(RND*2):a$(0)="JOUER BLEU ":a$(1)="JOUER ROU
  GE"
20 FOR j=1 TO 8
21 FOR i=1 TO 8
22 IF r(i,j)<>6 THEN CALL dessin (i,j,c(i,j),r(i,j))
23 NEXT i,j
24 debut:
25 COLOR 4-h:LOCATE 19,1:PRINT a$(h);
26 COLOR 2:LOCATE 20,1:PRINT "CLIQUEZ LA PIECE";
27 WHILE MOUSE(0)=0:WEND
28 WHILE MOUSE(0)<>0
29 x=INT((MOUSE(5)+16)/30):y=INT((MOUSE(6)+3)/18)
30 WEND
31 IF c(x,y)<>4-h THEN debut ELSE z=1:CALL dessin(x,y,
  z,r(x,y))
32 COLOR 2:LOCATE 20,1:PRINT "CLIQUEZ LA CASE ";:SOUND
  800,1
33 WHILE MOUSE(0)=0:WEND
34 WHILE MOUSE(0)<>0
35 u=INT((MOUSE(5)+16)/30):v=INT((MOUSE(6)+3)/18)
36 WEND
37 IF u<1 OR v<1 OR u>8 OR v>8 THEN z=4-h:CALL dessin(
  x,y,z,r(x,y)):GOTO debut
38 LINE (30*x-18,18*y-4)-(30*x+12,18*y+14),((x+y) MOD
  2)+13,bf
39 LINE (30*u-18,18*v-4)-(30*u+12,18*v+14),((u+v) MOD
  2)+13,bf
40 CALL dessin(u,v,4-h,r(x,y)):c(u,v)=4-h:r(u,v)=r(x,y)
  :c(x,y)=((x+y) MOD 2)+13:r(x,y)=6
41 h=ABS(h-1):SOUND 200+300*h,3:GOTO debut
42 SUB dessin(a,b,c,d) STATIC
43 SHARED t()
44 f=30*a+3:g=18*b+6
45 AREA(f,g):FOR i=14 TO 1 STEP-1:AREA STEP (t(d,1,i),
  t(d,2,i)):NEXT i
46 COLOR c:AREAFILL
47 END SUB
48 DATA 4,2,0,2,-2,2,3,2,3,-2,-2,-2,0,-2,4,-2,-10,0,0,
  0,0,0,0,0,0,0,0
49 DATA 5,3,0,2,-2,0,0,2,3,0,0,-2,2,0,0,2,3,0,0,-2,-2,
  0,0,-2,5,-3,-14,0
50 DATA 5,2,0,2,-3,2,5,2,5,-2,-3,-2,0,-2,5,-2,-14,0,0,
  0,0,0,0,0,0,0,0
51 DATA 5,2,0,3,-4,-2,-1,2,5,2,5,-2,-3,-1,0,-2,5,-2,-1
  2,0,0,0,0,0,0,0,0
52 DATA 5,3,-3,2,0,2,2,-2,2,2,2,-2,2,2,0,-2,-3,-2,5,-3
  ,-12,0,0,0,0,0,0,0
53 DATA 5,2,-3,3,2,2,-3,1,8,0,-3,-1,2,-2,-3,-3,5,-2,-1
  0,0,0,0,0,0,0,0

```

Poker

REFERENCES :

Difficulté de programmation : difficile.

Catégorie de jeu : jeu contre l'Amiga.

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, l'Amiga. 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 de l'Amiga de jouer.

LE PROGRAMME :

Les lignes 6 à 13 définissent les cartes. Une variable N est dimensionnée à 16, en ligne 3, pour recevoir dix cartes joueur et cinq cartes Amiga.

Le tirage sera effectué par une boucle I (20-24) et le contrôle par la ligne 23.

L'affichage des cartes est réalisé par les lignes 16 à 19 en appelant le sous-programme rectangle.

Les cartes du joueur sont retournées en appelant le sous-programme par la ligne 26.

Les lignes 31 à 36 permettent de retourner à nouveau les cartes refusées.

Le tirage définitif est fait par la ligne 45.

Les lignes 47 à 50 gèrent une nouvelle partie.

POUR JOUER :

L'ordre RUN lance l'exécution du programme, et les 5 premières cartes du joueur sont alors affichées, d'abord à l'envers, puis à l'endroit.

Le programme utilise la souris de l'Amiga pour choisir les cartes à retirer. A cet effet, on devra cliquer avec le bouton de gauche sur les cartes refusées qui sont retournées. On annule le choix en cliquant une deuxième fois. On appuie sur "espace" pour valider les choix effectués et provoquer un retrait éventuel, suivi du jeu machine. On appuie sur une touche quelconque pour faire une autre partie, et sur Ø pour finir.

EXTENSIONS POSSIBLES :

Elles seront, bien sûr, très nombreuses, depuis un affichage graphique en couleur des cartes jusqu'à l'amélioration du jeu du PC, l'introduction des règles sur les paris et même, pourquoi pas, des possibilités de bluff.

```

1  REM POKER
2  SCREEN 1,640,210,4,2:WINDOW 1,"POKER",,2,1
3  RANDOMIZE TIMER:DIM carte%(320),n(16),t(33),d%(7)
4  v$=" 7 8 9 10 VALET DAME ROI AS "
5  COLOR 2,7:CLS:DEFINT i,j,k
6  FOR i=1 TO 4
7  IF i=1 THEN RESTORE carreau ELSE IF i=2 THEN RESTOR
   E coeur
8  IF i=3 THEN RESTORE trefle ELSE IF i=4 THEN RESTORE
   pique
9  GOSUB lecture
10 LINE(96*i,80)-(96*i+16,87),i+2,bf
11 GET (96*i,80)-(96*i+16,87),carte%(80*(i-1))
12 NEXT i
13 RESTORE decor:GOSUB lecture
14 debut:
15 COLOR 2,0:CLS:FOR i=0 TO 32:t(i)=0:NEXT i
16 FOR i=1 TO 5
17 x=8*i-4:a=0:y=2:LOCATE y-1,13*i-7:PRINT i:GOSUB rec
   tangle
18 a=10:c1=3:GOSUB rectangle
19 NEXT i
20 FOR i=1 TO 16
21 choix:
22 n(i)=INT(RND*32)
23 IF t(n(i))=0 THEN t(n(i))=1 ELSE choix
24 NEXT i
25 LOCATE 10,1:COLOR 7:PRINT "CLIQUEZ SUR VOS CARTES P
   OUR CHANGER, PUIS ESPACE"
26 RESTORE efface:GOSUB lecture:a=0:GOSUB tir
27 attends:
28 WHILE MOUSE(0)=0
29 r$=INKEY$:IF r$=" " THEN suite
30 WEND
31 WHILE MOUSE(0)<>0
32 x=INT((MOUSE(5)-44)/64):y=INT((MOUSE(6)-10)/64)
33 WEND
34 IF y<>0 OR (x MOD 2<>0) THEN attends ELSE i=INT(x/2
   )+1:x=8*i-4:a=0
35 IF c(i)=0 THEN
36 RESTORE decor:GOSUB lecture:c(i)=1:c1=3:GOSUB recta
   ngle
37 ELSE
38 RESTORE efface:GOSUB lecture:GOSUB tircarte
39 END IF
40 GOTO attends
41 suite:
42 FOR i=1 TO 5
43 IF c(i)=1 THEN n(i)=n(i+5)
44 NEXT i

```

```

45 GOSUB tir:a=a+10:GOSUB tir
46 LOCATE 21,1:COLOR 9,0
47 PRINT "APPUYEZ UNE TOUCHE POUR CONTINUER, 0 POUR FI
  NIR";
48 r$=INPUT$(1):IF r$<>"0" THEN debut
49 FOR i=100 TO 1200 STEP 12:SOUND i,2:NEXT i
50 END
51 tir:
52 FOR i=1 TO 5:GOSUB tircarte:NEXT i
53 RETURN
54 tircarte:
55 c(i)=0:c1=7:x=8*i-4:GOSUB rectangle
56 FOR k=1 TO 500:NEXT k:SOUND INT(RND*800+200),2
57 co=INT(n(i+a)/8):f=5*(n(i+a) MOD 8)+1:s$=MID$(v$,f,
  5)
58 FOR j=1 TO 4
59 PUT (16*x-16+32*((j+1) MOD 2),10*a+15-44*(j>2)),car
  te%(80*co),PSET
60 NEXT j
61 LOCATE a+5-(a>5),13*i-8:COLOR co+3,7:PRINT s$
62 RETURN
63 lecture:
64 FOR j=0 TO 7:READ d%(j):NEXT j
65 PATTERN ,d%
66 RETURN
67 rectangle:
68 LINE(16*x-26,10*a+10)-(16*x+44,10*a+70),c1,bf
69 LINE(16*x-27,10*a+9)-(16*x+45,10*a+71),11,b
70 RETURN
71 efface:
72 DATA &hffff,&hffff,&hffff,&hffff,&hffff,&hffff,&hff
  ff,&hffff
73 carreau:
74 DATA &h0100,&h0380,&h07c0,&h0fe0,&h07c0,&h0380,&h01
  00,&h0000
75 coeur:
76 DATA &h0c60,&h1ef0,&h1ff0,&h1ff0,&h0fe0,&h07c0,&h03
  80,&h0100
77 trefle:
78 DATA &h0100,&h0380,&h0100,&h0540,&h0fe0,&h0540,&h01
  00,&h0380
79 pique:
80 DATA &h0100,&h0380,&h07c0,&h0fe0,&h07c0,&h0100,&h03
  80,&h07c0
81 decor:
82 DATA &h8888,&h5555,&h2222,&h5555,&h8888,&h5555,&h22
  22,&h5555

```


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 10 icônes.

LE PROGRAMME :

Les lignes 5 à 8 définissent les dix icônes en appelant un programme de dessin qui utilise les lignes 31 à 47.

Les boucles I et J de la ligne 5 lisent les DATA des dessins.

Les lignes 9 à 21 permettent de choisir l'icône, la largeur, la hauteur et la couleur du dessin à l'aide de la souris.

Les lignes 22 à 27 permettent de déposer l'icône à l'emplacement choisi.

POUR JOUER :

On lance le programme par RUN.

Les icônes sont affichées dans des carrés en haut à gauche : on déplace la souris pour choisir l'icône et on clique sur le bouton de gauche.

On peut, en outre, choisir la largeur, la hauteur et la couleur des diverses icônes proposées. On modifie les paramètres de dessin (couleur, largeur et hauteur) en cliquant sur les signes + et - disposés juste à côté des noms.

On déplace ensuite le curseur à l'emplacement voulu de l'écran et on clique pour déposer l'icône.

On quitte le programme en tapant CTRL et C.

EXTENSIONS POSSIBLES :

On pourra augmenter le nombre d'icônes disponibles, améliorer le choix des couleurs, l'agrandissement des icônes, etc.

```

1  REM FAIS MOI UN DESSIN
2  SCREEN 1,640,210,4,2:WINDOW 1,"FAIS MOI UN DESSIN",
   ,2,1
3  DIM t(10,37):e=0:LINE (0,40)-(640,40)
4  LOCATE 2,59:PRINT "+ -":LOCATE 3,59:PRINT "+ -":LOC
   ATE 4,59:PRINT "+ -"
5  FOR i=0 TO 10:FOR j=1 TO 34:READ t(i,j):NEXT j,i
6  FOR k=0 TO 10
7  m(1)=1:m(2)=1:x=40*k+24:y=22:LINE(x-8,y-15)-(x+22,y
   +5),,b:i=k:m(3)=4:GOSUB dessin
8  NEXT k
9  debut:
10 LOCATE 2,46:PRINT "Largeur *" m(1):LOCATE 3,46:PRIN
   T "Hauteur *" m(2)
11 LOCATE 4,46:PRINT "Couleur :" m(3)
12 WHILE MOUSE(0)=0:WEND
13 WHILE MOUSE(0)<>0
14 xo=INT((MOUSE(5)-11)/10):yo=INT((MOUSE(6))/9)
15 IF (xo=57 OR xo=59) AND (yo=1 OR yo=2 OR yo=3) THEN
16 g=xo-57:m(yo)=ABS(m(yo)+(1 AND g=0)-(1 AND g=2))
17 IF yo=3 THEN m(yo)=m(yo) MOD 16
18 WHILE MOUSE(0)<>0:WEND
19 END IF
20 WEND
21 IF e=0 AND ((xo MOD 4)=0 OR yo<1 OR yo>4 OR xo>44)
   THEN debut ELSE xo=INT(xo/4)
22 IF e=1 THEN
23 y=MOUSE(6):IF y<40 AND MOUSE(5)<480 THEN e=0:x=p1:G
   OSUB param
24 IF y>40 THEN e=0:x=MOUSE(5):GOSUB dessin:x=p1:GOSUB
   param:GOTO debut
25 END IF
26 IF e=0 AND y<40 THEN x=40*xo+24:p1=x:i=xo:m(3)=2:GO
   SUB dessin:e=1:GOTO debut
27 GOTO debut
28 param:
29 y=22:x=p1:m(1)=1:m(2)=1:m(3)=4:GOSUB dessin
30 RETURN
31 dessin:
32 m=m(1):p=-m(2):COLOR m(3)
33 a1=x+m*t(i,1):b1=y+p*t(i,2):a2=a1+m*t(i,3):b2=b1+p*
   t(i,4):a3=a2+m*t(i,5)
34 b3=b2+p*t(i,6):a4=a3+m*t(i,7):b4=b3+p*t(i,8):a5=a4+
   m*t(i,9):b5=b4+p*t(i,10)
35 a6=a5+m*t(i,11):b6=b5+p*t(i,12):a7=a6+m*t(i,13):b7=
   b6+p*t(i,14)
36 a8=a7+m*t(i,15):b8=b7+p*t(i,16):a9=a8+m*t(i,17):b9=
   b8+p*t(i,18)

```

```

37 a10=a9+m*t(i,19):b10=b9+p*t(i,20)
38 a11=a10+m*t(i,21):b11=b10+p*t(i,22):a12=a11+m*t(i,2
3)
39 b12=b11+p*t(i,24):a13=a12+m*t(i,25):b13=b12+p*t(i,2
6)
40 a14=a13+m*t(i,27):b14=b13+p*t(i,28)
41 a15=a14+m*t(i,29):b15=b14+p*t(i,30)
42 a16=a15+m*t(i,31):b16=b15+p*t(i,32):a17=a16+m*t(i,3
3):b17=b16+p*t(i,34)
43 AREA(a1,b1):AREA(a2,b2):AREA(a3,b3):AREA(a4,b4):ARE
A(a5,b5)
44 AREA(a6,b6):AREA(a7,b7):AREA(a8,b8):AREA(a9,b9):ARE
A(a10,b10)
45 AREA(a11,b11):AREA(a12,b12):AREA(a13,b13):AREA(a14,
b14):AREA(a15,b15)
46 AREA(a16,b16):AREA(a17,b17):AREAFILL
47 m(1)=1:m(2)=1:m(3)=4:COLOR 7:RETURN
48 DATA 2,1,0,5,-3,-3,-1,1,4,4,-2,3,8,0,-2,-3,4,-4,-1,
-1,-3,3,0,-5,2,-1,-3,0,-1,3,-1,-3,-3,0
49 DATA 2,3,0,4,-3,-2,2,4,-2,-2,3,3,3,-3,-2,2,2,-4,-3,
2,0,-4,2,-3,-4,0,0,0,0,0,0,0,0,0
50 DATA 4,4,3,-3,1,1,1,-1,3,3,3,-3,-3,2,-3,-3,1,-1,-5,
0,1,1,-3,3,-3,-3,0,0,0,0,0,0,0,0
51 DATA 0,6,2,2,5,0,0,2,2,0,0,-2,2,0,2,-2,0,-6,-3,0,0,
4,-3,0,0,-4,-7,0,0,0,0,0,0,0,0
52 DATA 3,7,6,0,2,-3,3,-1,3,-2,-3,0,-2,-2,-2,2,-5,0,-2
,-2,-2,2,-2,0,0,0,0,0,0,0,0,0,0,0
53 DATA 0,9,1,1,1,-1,0,-2,5,0,0,-2,-5,0,0,-3,5,0,0,-2,
-5,0,0,-1,-2,0,0,0,0,0,0,0,0,0
54 DATA 0,8,11,0,0,-4,4,0,0,-4,-2,0,-2,-2,-1,0,-2,2,-2
,0,-2,-2,-1,0,-2,2,-1,0,0,0,0,0,0,0,0
55 DATA 1,3,-3,2,2,0,2,1,2,2,-3,0,3,2,1,2,1,-4,3,-1,-3
,-1,-3,-3,-2,-3,2,0,-3,0,0,0,0,0,0
56 DATA 3,2,-2,3,3,-2,3,2,-1,-3,2,-2,-3,0,-2,-2,0,2,-2
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
57 DATA 0,2,14,0,0,-2,-14,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0
58 DATA 0,10,3,0,0,-10,-3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0

```

Animation d'images

REFERENCES :

Difficulté de programmation : moyenne.

Catégorie de jeu : animation.

Difficulté du jeu : difficile.

LE JEU :

Huit grilles de dessin sont affichées sur l'écran. Un menu propose de choisir la couleur de chaque point et l'on peut cliquer avec la souris sur un emplacement d'une des grilles pour allumer le point dans la couleur désirée.

Les images obtenues sont affichées en taille réelle en bas de l'écran.

Lorsque l'on a fini, on appuie sur la barre d'espace. Un exemple d'animation utilisant ces huit images est alors proposé.

LE PROGRAMME :

Les lignes 4 et 5 définissent les menus en lisant les DATA des lignes 38 et 39.

Les boucles des lignes 6 à 14 dessinent les grilles.

La ligne 19 affiche un rectangle donnant la couleur actuelle.

La ligne 22 permet le choix du point à colorier.

Les lignes 25 et 26 colorent un rectangle sur la grille et un point sur l'image taille réelle.

Les lignes 29 à 36 s'occupent de l'animation.

POUR JOUER :

On lance le programme par RUN. Les grilles s'affichent.

On choisit la couleur sur le menu en cliquant avec le bouton de droite. On colore en cliquant avec le bouton de gauche sur la case choisie.

Une fois les huit dessins préparés, on clique sur la barre d'espace pour observer l'animation.

L'appui sur une touche quelconque permet de refaire une animation.

On devra taper CTRL et C pour arrêter.

EXTENSIONS POSSIBLES :

On pourra envisager de sauver sur disquette les images réalisées pour pouvoir les récupérer dans un autre programme. On peut aussi utiliser cette technique pour préparer des SPRITES ou des BOBS.

```

1  REM ANIMATION D'IMAGES
2  DIM co$(16),t%(2400):SCREEN 1,640,210,4,2:WINDOW 1,
   "",,2,1
3  COLOR 0,1:CLS:DEFINT a-z
4  FOR i=3 TO 15:READ co$(i):MENU 1,i-3,1," "+co$(i):
   NEXT i
5  MENU 1,1,2," "+co$(4):f=1:co=1:MENU 2,0,1," "
6  FOR i=70 TO 574 STEP 8
7  FOR j=0 TO 155 STEP 5
8  LINE(i,j)-(i+8,j+5),15,b
9  NEXT j,i
10 FOR i=70 TO 574 STEP 128
11 FOR j=0 TO 155 STEP 80
12 LINE(i,j)-(i+128,j+80),2,b
13 NEXT j,i
14 FOR i=10 TO 380 STEP 48:LINE(i,170)-(i+17,187),2,b:
   NEXT i
15 LOCATE 20,38:PRINT "- ESPACE - POUR FINIR";
16 debut:
17 WHILE MOUSE(0)=0
18 IF MENU(0)=1 THEN co=MENU(1):MENU 1,f,1:MENU 1,co,2
   :f=co
19 LINE(14,140)-(34,154),co+2,bf
20 r$=INKEY$:IF r$=" " THEN fin
21 WEND
22 WHILE MOUSE(0)<>0:x=INT(MOUSE(5)/8):y=INT(MOUSE(6)/
   5):WEND
23 IF x<9 OR x>72 OR y>31 THEN debut
24 u=2*(x-9)-128*(y>15):v=y MOD 16
25 LINE(8*x-1,5*y+1)-(8*x+5,5*y+4),co+2,bf
26 PSET(11+32*INT(u/32)+(u/2),v+171),co+2
27 GOTO debut
28 fin:
29 FOR i=10 TO 380 STEP 48
30 j=INT((i-10)/48):GET(i+1,171)-(i+15,186),t%(300*j)
31 NEXT i
32 test:
33 CLS
34 FOR i=0 TO 600
35 j=i MOD 8:PUT(i,80),t%(300*j),PSET
36 NEXT i
37 r$=INPUT$(1):GOTO test
38 DATA COULEUR,ROUGE,BLEU F,ROSE,BLEU C,BLANC,MARRON
39 DATA ORANGE,VERT C,JAUNE,BLEU CIEL,VIOLET,VERT

```

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 :

Les lignes 3 à 5 dessinent le vaisseau spatial, et les lignes 6 et 7 dessinent les météorites.

La ligne 14 permet de faire apparaître le vaisseau.

La ligne 16 fait apparaître les météorites, et la ligne 17 affiche le minerai d'urane.

Les lignes 21 à 23 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 (lignes 24 à 26).

Le déplacement du vaisseau est géré par les lignes 19 et 20, et l'avance des objets sur l'écran par la ligne 18.

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 numériques 1, 3, 4 et 6 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. Les touches 1 et 3 donnent un déplacement lent, et 4 et 6 un déplacement plus rapide.

On pourra aussi arrêter à tout moment en tapant CTRL et Break.

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  DEFINT a-z: DIM des1%(46), des2%(46): s=0: COLOR ,1: CLS
3  COLOR 2: FOR i=1 TO 10: READ a,b: AREA(a,b): NEXT i: ARE
  AFILL
4  CIRCLE(13,4),3,3: PAINT(13,4),3,3
5  GET(0,0)-(26,10), des1%: CLS: COLOR 2
6  FOR i=1 TO 12: READ a,b: AREA(a,b): NEXT i: AREA FILL
7  GET(0,0)-(26,10), des2%
8  CLS: RANDOMIZE TIMER
9  SCREEN 1,640,200,4,2: WINDOW 1,"CAPTURES DANS L'ESPA
  CE",,2,1: COLOR 2,1: CLS
10 s1$=MKL$(0)+MKL$(0)+MKL$(des1%(2))+MKL$(des1%(0))+M
  KL$(des1%(1))
11 s1$=s1$+MKI$(24)+MKI$(3)+MKI$(0)
12 FOR i=3 TO 46: s1$=s1$+MKI$(des1%(i)): NEXT i
13 OBJECT.SHAPE 1,s1$: a=320: b=80: OBJECT.X 1,a: OBJECT.Y
  1,b
14 OBJECT.ON: OBJECT.START
15 debut:
16 IF RND<.4 THEN PUT(16*INT(RND*37+1),180), des2%, PSET
17 IF RND<.3 THEN x=16*INT(RND*36+2): CIRCLE(x,180),6,1
  1: PAINT(x,180),11,11
18 SCROLL(0,70)-(639,194),0,-2
19 r$=INKEY$: f=(r$="1")-(r$="3")+2*(r$="4")-2*(r$="6")
20 a=a+2*f: OBJECT.X 1,a
21 e=POINT(a+4,b+12): f=POINT(a+13,b+13): g=POINT(a+22,b
  +12)
22 IF e+f+g=3 THEN debut
23 IF e=11 OR f=11 OR g=11 THEN s=s+1: SOUND 800,2: GOTO
  debut
24 FOR i=10 TO 2000 STEP 20: SOUND i,1: NEXT i
25 PRINT "C'EST FINI...."
26 PRINT: PRINT "score : "; s: END
27 DATA 8,4,6,7,8,8,7,10,10,9,16,9,19,10,18,8,20,7,17,
  4
28 DATA 12,1,11,4,8,5,11,6,10,9,13,7,16,9,15,6,18,5,15
  ,4,14,1,13,4

```

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 :

Les lignes 4 à 7 préparent les divers dessins.

La position de chaque météorite est déterminée par les lignes 18 et 19.

L'engin spatial est affiché par la ligne 22.

L'avance des météorites est réalisée par la ligne 20.

Les lignes 23 et 24 vérifient si vous avez ou non heurté une météorite.

Les lignes 13 et 14 dessinent votre vaisseau spatial, et la ligne 21 permet de le déplacer.

Le compteur se trouve en ligne 8.

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.

Le déplacement du joueur vers la droite ou vers la gauche se fait en déplaçant la souris dans la direction voulue et en cliquant.

On peut arrêter la partie à tout moment en tapant CTRL et Break. 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  DEFINT a-z: DIM des1%(46), des2%(46): COLOR ,1: CLS
3  COLOR 2: FOR i=1 TO 10: READ a,b: AREA(a,b): NEXT i: ARE
  AFILL
4  CIRCLE(13,4),3,3: PAINT(13,4),3,3
5  GET(0,0)-(26,10), des1%: CLS: COLOR 2
6  FOR i=1 TO 12: READ a,b: AREA(a,b): NEXT i: AREAFILL
7  GET(0,0)-(26,10), des2%
8  CLS: RANDOMIZE TIMER: t=TIMER
9  SCREEN 1,640,200,4,2: WINDOW 1,"DANGER SPATIAL",,2,1
  : COLOR 2,1: CLS
10 s1$=MKL$(0)+MKL$(0)+MKL$(des1%(2))+MKL$(des1%(0))+M
  KL$(des1%(1))
11 s1$=s1$+MKI$(24)+MKI$(3)+MKI$(0)
12 FOR i=3 TO 46: s1$=s1$+MKI$(des1%(i)): NEXT i
13 OBJECT.SHAPE 1,s1$: a=320: b=100: OBJECT.X 1,a: OBJECT.
  Y 1,b
14 OBJECT.ON: OBJECT.START
15 ON COLLISION GOSUB fin
16 COLLISION ON
17 debut:
18 IF RND<.3 THEN PUT(16*INT(RND*37+1),180), des2%, PSET
19 IF RND<.2 THEN PUT(16*INT(RND*37+1),180), des2%, PSET
20 SCROLL(0,0)-(639,200),0,-1: z=ABS(MOUSE(0))
21 x=MOUSE(5): y=MOUSE(6): f=2*(x<a)+1: g=2*(y<b)+1
22 a=a+z*f: b=b+z*g: OBJECT.X 1,a: OBJECT.Y 1,b
23 e=POINT(a+1,b): f=POINT(a+1,b+13): g=POINT(a+23,b): h=
  POINT(a+23,b+13)
24 IF e+f+g+h=4 THEN debut
25 fin:
26 FOR i=10 TO 2000 STEP 20: SOUND i,1: NEXT i
27 PRINT "C'EST FINI...."
28 PRINT: PRINT"TEMPS : "; INT(TIMER-t): END
29 DATA 8,4,6,7,8,8,7,10,10,9,16,9,19,10,18,8,20,7,17,
  4
30 DATA 12,1,11,4,8,5,11,6,10,9,13,7,16,9,15,6,18,5,15
  ,4,14,1,13,4

```

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, grâce aux flèches de mouvement, et vous allez essayer d'amener l'appareil ennemi au centre de votre viseur.

Vous tirez sur lui en frappant 5 ou 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 GAGNE).

LE PROGRAMME :

Les lignes 3 à 15 préparent les dessins en utilisant les DATA des lignes 46 et 47.

Le lancement des projectiles est effectué par les lignes 28 à 30.

Les lignes 24 à 26 vous permettent le pilotage de votre appareil.

Les lignes 16 à 18 dessinent le viseur de votre appareil.

Le sous-programme, gérant le tir, va de la ligne 32 à la ligne 42 et le test de fin de partie est effectué en ligne 34.

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 des touches du clavier numérique (2, 4, 6 et 8) et vous pouvez tirer à tout moment en pressant la touche 5 ou 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 GAGNE, si vous touchez l'avion ennemi.

On peut aussi arrêter à tout moment en frappant CTRL et Break.

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 DEFINT a-z:DIM des1%(46),des2%(46):COLOR ,0:CLS
3 COLOR 3:FOR i=1 TO 7:READ a,b:AREA(a,b):NEXT i:AREA
  FILL
4 GET(0,0)-(26,10),des1%:COLOR 2:CLS
5 FOR i=1 TO 11:READ a,b:AREA(a,b):NEXT i:AREAFILL
6 GET(0,0)-(26,10),des2%
7 CLS:RANDOMIZE TIMER:SCREEN 1,640,210,4,2
8 WINDOW 1,"COMBAT AERIEN",,2,1:COLOR 2,0:CLS
9 s1$=MKL$(0)+MKL$(0)+MKL$(des1%(2))+MKL$(des1%(0))+M
  KL$(des1%(1))
10 s1$=s1$+MKI$(24)+MKI$(3)+MKI$(0)
11 s2$=MKL$(0)+MKL$(0)+MKL$(des2%(2))+MKL$(des2%(0))+M
  KL$(des2%(1))
12 s2$=s2$+MKI$(24)+MKI$(3)+MKI$(0)
13 FOR i=3 TO 46:s1$=s1$+MKI$(des1%(i)):s2$=s2$+MKI$(d
  es2%(i)):NEXT i
14 OBJECT.SHAPE 1,s1$:OBJECT.SHAPE 2,s2$:OBJECT.SHAPE
  3,1
15 OBJECT.PLANES 1,15:OBJECT.PLANES 3,15
16 LINE(300,80)-(340,120),3,b:CIRCLE(320,100),6,11
17 LINE(290,100)-(310,100),9:LINE(330,100)-(350,100),9
18 LINE (320,70)-(320,90),9:LINE(320,110)-(320,130),9
19 a=50:b=320:OBJECT.X 2,b:OBJECT.Y 2,a:OBJECT.ON 2
20 debut:
21 ON COLLISION GOSUB controle:COLLISION ON
22 bouge:
23 a=a+3*(2*INT(RND*2)-1):b=b+2*(2*INT(RND*2)-1):OBJEC
  T.X 2,b:OBJECT.Y 2,a
24 a$=INKEY$:b=b-5*(a$="6")+5*(a$="4"):a=a-4*(a$="2")+
  4*(a$="8")
25 IF b<1 OR b>610 THEN b=ABS(b-610)
26 IF a<3 OR a>190 THEN a=ABS(a-190)
27 IF (a$<>" " AND a$<>"5") THEN bouge
28 OBJECT.X 1,206:OBJECT.Y 1,190:OBJECT.VX 1,50:OBJECT
  .VY 1,-50
29 OBJECT.X 3,406:OBJECT.Y 3,190:OBJECT.VX 3,-50:OBJEC
  T.VY 3,-50
30 OBJECT.ON 1:OBJECT.ON 3:OBJECT.START 1:OBJECT.START
  3
31 GOTO bouge
32 controle:
33 a=OBJECT.X(2):b=OBJECT.Y(2):c=COLLISION(0):d=COLLIS
  ION(c)
34 IF a>300 AND a<340 AND b>80 AND b<120 THEN PRINT "B
  RAVO":GOTO fin
35 COLLISION OFF:SOUND 1000,1
36 IF c=0 THEN
37 OBJECT.START 1:OBJECT.START 3
38 FOR i=1 TO 30:NEXT i:GOTO debut
39 ELSE
40 WHILE c<>0:c=COLLISION(0):d=COLLISION(c):WEND
41 END IF
42 GOTO controle
43 fin:
44 FOR i=50 TO 1200 STEP 20:SOUND i,1:NEXT i
45 END
46 DATA 13,2,12,4,12,7,10,9,16,9,14,7,14,4
47 DATA 7,6,10,7,14,6,18,7,21,6,20,5,16,5,15,4,13,4,12
  ,5,8,5

```

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 :

Les lignes 3 à 16 préparent les dessins en lisant les DATA des lignes 45 à 47.

La ligne 22 permet à l'Amiga de choisir la profondeur du sous-marin.

La ligne 25 dessine votre bâtiment.

La ligne 20 dessine la partie supérieure de l'écran.

La ligne 23 dessine le sous-marin.

Le tir est effectué par un programme spécifique placé aux lignes 34 et 35.

Le déplacement (très rapide) de votre navire est réalisé par les lignes 29 à 31.

Le test, permettant de savoir si l'on a touché ou non, est effectué dans la partie contrôle des lignes 36 à 44.

POUR JOUER :

Taper RUN pour commencer la partie.

Le bâtiment sera déplacé grâce aux touches B et V (déplacement lent) ou N et C (déplacement rapide).

Le tir est effectué en appuyant sur une touche de 1 à 9.

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 faire CTRL et Break 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 DEFINT a-z: DIM des1%(46), des2%(46), des3%(46): COLOR
,0:CLS
3 COLOR 3:FOR i=1 TO 14:READ a,b:AREA(a+3,b+3):NEXT i
:AREAFILL
4 GET(0,0)-(26,10),des1%:COLOR 3:CLS
5 FOR i=1 TO 9:READ a,b:AREA(a,b):NEXT i:AREAFILL
6 GET(0,0)-(26,10),des2%:COLOR ,1:CLS
7 FOR i=1 TO 12:READ a,b:AREA(a,b):NEXT i:AREAFILL
8 GET(0,0)-(26,10),des3%
9 CLS:RANDOMIZE TIMER:SCREEN 1,640,200,4,2
10 WINDOW 1,"SOUS-MARIN",,2,1:COLOR 2,0:CLS
11 s1$=MKL$(0)+MKL$(0)+MKL$(des1%(2))+MKL$(des1%(0))+M
KL$(des1%(1))
12 s1$=s1$+MKI$(24)+MKI$(3)+MKI$(0)
13 s2$=MKL$(0)+MKL$(0)+MKL$(des2%(2))+MKL$(des2%(0))+M
KL$(des2%(1))
14 s2$=s2$+MKI$(24)+MKI$(3)+MKI$(0)
15 FOR i=3 TO 46:s1$=s1$+MKI$(des1%(i)):s2$=s2$+MKI$(d
es2%(i)):NEXT i
16 OBJECT.SHAPE 1,s1$:OBJECT.SHAPE 2,s2$
17 ON COLLISION GOSUB controle:COLLISION ON
18 debut:
19 p!=0:c=35:s=6:t$=STRING$(70," ")
20 CLS:COLOR ,1:FOR i=1 TO 6:PRINT t$:NEXT
21 nouveau:
22 vs=INT(RND*45+8):s=INT(RND*13+7):p!=p!+.01:OBJECT.X
1,600:OBJECT.Y 1,9*s
23 OBJECT.VX 1,-vs:OBJECT.VY 1,0:OBJECT.ON 1:OBJECT.ST
ART 1:OBJECT.OFF 2
24 avance:
25 PUT(10*c,43),des3%,PSET:d=c:a$="":COLLISION ON
26 LOCATE 19,1:COLOR 7,0:PRINT "SCORE :";:PRINT USING
"###.###";p!
27 IF p!>99 THEN LOCATE 19,1:PRINT "C'EST FINI...":EN
D
28 entree:
29 a$=UCASE$(INKEY%):c=c-(((a$="B") AND c<62)-((a$="V"
) AND c>2))
30 c=c-3*(((a$="N") AND c<62)-((a$="C") AND c>2))
31 IF d<c THEN COLOR ,1:LOCATE 6,d+1:PRINT " "
32 IF a$="" THEN FOR i=1 TO 40:NEXT i:GOTO entree
33 IF VAL(a$)=0 THEN avance
34 OBJECT.X 2,10*c:OBJECT.Y 2,54:OBJECT.VX 2,0:OBJECT.
VY 2,8*VAL(a$)
35 OBJECT.ON 2:OBJECT.START 2:COLLISION ON:GOTO avance
36 controle:
37 SOUND 400,4:a=COLLISION(0):b=COLLISION(a):COLLISION
OFF
38 IF b=-3 THEN OBJECT.OFF 2:GOTO avance
39 IF b>=0 THEN
40 p!=p!+1:LOCATE s+1,c:COLOR 11,0:PRINT "***";:OBJECT
.START 2
41 FOR i=1000 TO 100 STEP -100:SOUND i,1:NEXT i:OBJECT
.START 1
42 LOCATE s+1,c:PRINT " ";:GOTO avance
43 END IF
44 GOTO nouveau
45 DATA 4,5,8,6,17,6,19,5,21,6,21,3,19,4,17,3,12,3,12,
0,11,0,11,3,8,3,5,4
46 DATA 10,1,11,2,11,4,9,6,12,9,14,6,13,4,13,2,14,1
47 DATA 3,6,5,10,20,10,22,6,19,6,19,4,17,4,17,2,16,2,1
6,4,12,4,12,6

```


Votre avis nous intéresse

Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques sur le présent ouvrage.

— **Titre de votre livre :**

— **Ce livre vous donne-t-il toute satisfaction ?**

.....
.....

— **Y a-t-il un aspect du problème que vous auriez aimé voir abordé ?**

.....
.....

Si vous souhaitez des éclaircissements techniques, écrivez-nous, ou laissez un message à notre service Minitel (3615 01 PSI), nous ne manquerons pas de vous répondre directement.

Avez-vous déjà acquis des livres P.S.I. ?

lesquels ?

qu'en pensez-vous ?

.....

Nom Prénom Age

Adresse

Profession

Centre d'intérêt

Je désire recevoir un catalogue gratuit et être informé régulièrement des nouveaux livres publiés.

☐ oui ☐ non

Veuillez renvoyer cette page, dûment remplie, aux

Editions PSI
5, place du Colonel Fabien
75491 PARIS CEDEX 10



Apprendre tout en se distrayant, tel est l'objectif de ce livre : ces 102 programmes de jeux vous guideront progressivement dans l'exploration de l'AmigaBasic sur Amiga 500, 1000 ou 2000. Niveau par niveau vous assimilerez de nouvelles instructions pour arriver à une plus grande maîtrise du Basic. Chaque niveau commence par une présentation pédagogique du jeu d'instructions utilisé.

Tous les jeux sont décrits et les programmes sont analysés afin d'en faciliter la modification.

DES LIVRES
POUR VOTRE
ORDINATEUR

PSI.

ISBN: 2-86595-474-9
135 FF



Couverture GIBRAT



102 PROGRAMMES AMIGAS

PAR JACQUES DECONCHAT

