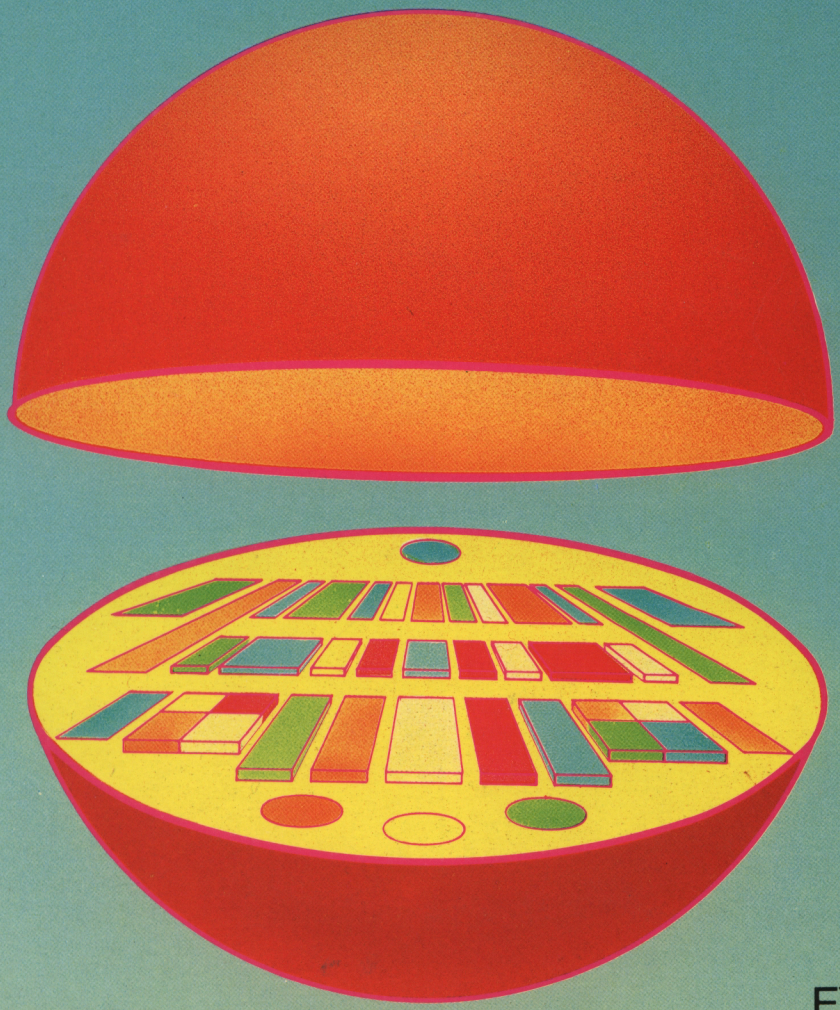
FAITES VOS JEUX AVEC ORIC



EYROLLES

CLAUDE DELANNOY

FAITES VOS JEUX AVEC ORIC

CHEZ LE MÊME ÉDITEUR

Dans la même collection

VANRYB-POLITIS — *Tout savoir sur New-Brain*, 104 p.; 1984.

GOLDSTEIN — *Le guide de l'IBM-PC*, 272 p.; 1984.

BUNN — *Obtenez le maximum de votre ATARI*, 104 p., 1984.

GARCIA — *Le Basic minimum.* 15 mots pour apprendre à programmer, 104 p., 1984.

WANNER — *Aller plus loin en BASIC TO7*, 312 p., 1984.

POLITIS-VANRYB — *Tout savoir sur ORIC*, 168 p., 1984.

VANRYB-POLITIS — *Tout savoir sur Multitech*, 152 p., 1984.

DELANNOY — *Faites vos jeux avec ORIC*, 224 p., 1984.

IFRAH — *Faites vos jeux avec CANON XO7*, 104 p., 1984.

RAVEL — *Tout savoir sur Laser 200-210*, 112 p., 1984.

VANRYB-POLITIS — *Tout savoir sur Lynx*, 176 p., 1984.

Du même auteur

DELANNOY — *Apprendre à programmer en Fortran*, 200 p.; 1983.

DELANNOY — *Apprendre à programmer en Basic*, 272 p.; 1984.

DELANNOY — *Les fichiers en Basic sur micro-ordinateur*, 168 p.; 1983. (coll. Micro-ordinateurs).

COLLECTION MICROPLUS

FAITES VOS JEUX AVEC ORIC

par

Claude DELANNOY

EYROLLES

61, boulevard Saint-Germain — 75005 PARIS
1984

Si vous désirez être tenu au courant de nos publications, il vous suffit d'adresser votre carte de visite au :

Service « Presse », Éditions EYROLLES
61, Boulevard Saint-Germain,
75240 PARIS CEDEX 05,

en précisant les domaines qui vous intéressent. Vous recevrez régulièrement un avis de parution des nouveautés en vente chez votre libraire habituel.

Avant-propos

Ce livre vous propose des jeux de très bonne qualité exploitant pleinement les capacités graphiques et sonores de l'ORIC.

Son rôle ne se limite toutefois pas à celui d'un simple recueil de programmes. Il vous offre en outre la possibilité d'adapter chacun d'entre eux suivant vos goûts et vos désirs, et cela sans même que vous n'ayez besoin de connaître le Basic. En effet, chaque jeu est accompagné de nombreuses suggestions de "personnalisation" qui vous sont expliquées dans le moindre détail.

Si vous avez quelques rudiments de Basic, ce livre peut constituer une bonne occasion de vous perfectionner en vous amusant. C'est en pensant à vous que nous avons préféré présenter les programmes par ordre de difficulté croissante, renonçant ainsi à l'habituel classement par "genre". Chacun d'entre eux est analysé, instruction par instruction et les "techniques" employées sont clairement expliquées. Vous serez ainsi parfaitement armés pour acquérir progressivement les connaissances et les

méthodes qui vous seront indispensables si vous souhaitez développer vous-mêmes vos propres jeux ou modifier profondément ceux qui vous sont proposés.

Les programmes ont été soigneusement vérifiés et leurs listes réalisées directement à partir de l'ORIC sur du matériel fiable. Vous pouvez donc les recopier en toute confiance.

Table des matières

Introduction

Afin que vous puissiez tirer le meilleur parti de ce livre, en fonction de vos désirs, nous croyons bon de vous préciser comment il est constitué.

Un **premier chapitre** vous donne des conseils pour la frappe des programmes proposés. Si vous ne connaissez pas le Basic, sa lecture vous sera très profitable. Elle vous aidera à détecter les erreurs de frappe que vous aurez pu commettre lors de la recopie d'un des jeux.

Les **chapitres 2 à 21** correspondent chacun à un jeu. Ils sont tous structurés de la même manière en cinq paragraphes :

- *paragraphe 1 : le jeu*. Vous y trouverez le scénario du jeu, ses règles et tout ce qu'il faut savoir pour y jouer avec votre ORIC.
- *paragraphe 2 : le programme*. (Il s'agit de sa liste).
- *paragraphe 3 : si vous souhaitez personnaliser ce programme*. De nombreuses possibilités d'adaptation vous sont proposées. Elles ne requièrent absolument aucune connaissance du Basic. Dans le cas où cela présente de l'intérêt, nous vous expliquons comment utiliser une poignée de jeu.

- *paragraphe 4 : description du programme.* Vous y verrez tout d'abord la liste des "techniques" utilisées dans le programme. Chacune d'entre elles fait l'objet d'une description détaillée en annexe. (Cette façon de procéder nous a évité de répéter des explications analogues d'un programme à un autre). Vient ensuite une analyse très complète du programme et des différents sous-programmes (dont nous avons fait un très large usage afin de mieux mettre en évidence le fonctionnement du jeu). Ces derniers sont décrits dans l'ordre où ils sont appelés dans le programme principal (et non pas nécessairement dans l'ordre des numéros de ligne). Notez que les structures de boucle sont mises en évidence par une typographie appropriée.

- *paragraphe 5 : liste des variables.* On y précise le rôle de chaque variable, à l'exception de quelques unes, rangées dans la rubrique "variables auxiliaires" ; ce sont là des variables n'ayant qu'un rôle secondaire et qui n'apparaissent que très localement : compteur de boucle FOR ayant peu d'instructions, variable chaîne destinée à lire une réponse O/N, etc...

Viennent enfin les différentes **annexes** dont nous avons parlé plus haut. Elles fournissent des indications sur les techniques spécifiques de programmation qui sont employées dans la réalisation des jeux et notamment sur la manière de les mettre en œuvre sur ORIC. Notez que vous y trouverez certaines informations très précieuses, qui ne figurent pas dans le manuel d'accompagnement de votre micro-ordinateur.

Bien entendu les paragraphes 4 et 5 ainsi que les annexes sont plus particulièrement destinés à ceux d'entre vous qui, ayant déjà des rudiments de Basic, souhaitent aller plus loin. Ce livre devrait vous permettre d'y parvenir en vous amusant.

1

Conseils pour la frappe des programmes

Ce petit chapitre vous donne quelques conseils qui devraient grandement faciliter cette opération souvent laborieuse qu'est la frappe des programmes.

1. Ne jamais exécuter immédiatement un programme que vous venez de taper

Bien entendu, il est nécessaire de le vérifier à l'écran (ou mieux sur imprimante si vous avez la chance d'en avoir une). Mais surtout, il est indispensable que vous en fassiez une sauvegarde sur cassette (par CSAVE) **AVANT** de tenter de l'exécuter.

En effet, il arrive (parfois) que l'exécution d'un programme erroné conduise à un "plantage" de l'ORIC tel que même le bouton RESET ne permette pas de se sortir de la situation. Dans ce cas, vous ne pouvez que débrancher puis rebrancher votre ORIC, ce qui a pour conséquence immédiate de vous faire perdre le programme que vous aviez patiemment introduit en mémoire.

Si, par contre, vous avez préalablement copié ce programme sur cassette, il vous suffit alors de le recharger par CLOAD puis d'en rechercher les anomalies.

Remarquez que cette technique peut également s'utiliser pour effectuer des sauvegardes partielles de programme. Cela peut se révéler utile lorsque :

— vous n'avez pas le temps de taper tout votre programme en une fois,
— vous craignez d'être interrompu par une "fausse manip" ou une coupure du secteur...

2. Recherche des anomalies

Il est probable que votre programme ne fonctionnera pas convenablement au premier coup. Trois types d'incidents peuvent apparaître. Nous allons voir comment remédier à chacun d'entre eux.

a) L'ORIC se plante

C'est la situation si désagréable dont nous avons parlé. Essayez d'appuyer sur le bouton RESET. Si cela ne fonctionne toujours pas, rechargez votre programme.

Il vous faut maintenant rechercher l'anomalie. Cela est, somme toutes, assez facile car, vu l'importance des dégâts, et compte-tenu de ce qu'il s'agit d'un programme BASIC, la cause ne peut être qu'une mauvaise utilisation de l'instruction POKE.

Comme celle-ci est peu utilisée dans nos programmes, vous n'aurez qu'un nombre limité de vérifications à faire. Examinez donc chacune de ces instructions en comparant avec la liste d'origine et en effectuant les corrections nécessaires.

Si vous ne trouvez aucune erreur dans les instructions POKE, il faut alors examiner les variables qui peuvent y intervenir. Par exemple dans :

POKE AM,XX

interviennent AM et XX. Vous allez alors examiner les instructions où ces variables (AM et XX) se voient attribuer une valeur, c'est-à-dire les instructions de la forme :

ou : AM = ...
XX = ...

Il est probable que l'une d'entre elles soit erronée. Sa correction devrait résoudre le problème.

b) L'exécution commence puis vous obtenez un message du type :

SYNTAX ERROR IN xxxx

où xxxx désigne un numéro de ligne.

C'est là l'erreur que vous rencontrerez fréquemment. Elle ne pose pas de problème particulier puisqu'il vous suffit de vérifier et de corriger la ligne mentionnée.

c) L'exécution commence puis vous obtenez un message du type :

ILLEGAL QUANTITY ERROR IN xxxx

Cela signifie que, dans la ligne spécifiée, vous utilisez une fonction (PLOT, DRAW, ...) avec de mauvaises valeurs pour les arguments.

Commencez par vérifier la ligne concernée. Si cela ne donne rien, c'est ce que ce sont des variables apparaissant dans l'instruction qui ont pris de mauvaises valeurs.

Si vous connaissez un peu Basic et la fonction concernée, vous pouvez faire écrire, en mode direct, le contenu des variables. Par exemple, si l'instruction est :

PLOT X,Y,C

vous pouvez, après le message d'erreur, taper simplement (sans numéro de ligne) :

```
PRINT X,Y,C
```

Vous devriez ainsi découvrir la variable coupable.

Il vous faut ensuite vérifier toutes les instructions dans lesquelles la variable (ou les variables si vous ne savez pas laquelle est en cause) reçoit une valeur. Ici, par exemple vous vérifierez toutes les instructions où apparaît quelque chose de la forme :

```
   X = ...
ou Y = ...
ou C = ...
```

D'autres erreurs sont malheureusement possibles. Celles que nous avons évoquées sont les plus courantes. Si vous prenez bien soin de relire ce que vous avez tapé et de le comparer avec la liste d'origine, il est probable que vous ne rencontrerez guère d'autres erreurs.

2

Force de frappe

1. Le jeu

Il s'agit d'un jeu d'habileté et de rapidité. Son but consiste à taper le plus vite possible sur la touche correspondant à une lettre qui apparaît dans un endroit quelconque de l'écran. La pratique de ce jeu peut vous aider à mieux connaître le clavier de votre ORIC et donc à augmenter votre vitesse de frappe.

A chaque partie, trente lettres vous sont ainsi proposées. Chacune d'entre elles est tirée au hasard parmi les vingt-six lettres de l'alphabet. Si vous tapez sur une mauvaise touche, un bruit de sonnette retentit. Par contre, si vous tapez sur la bonne touche, vous entendrez un "coup de fusil"; le programme vous proposera alors une nouvelle lettre, après un petit temps d'attente qui sera variable d'une fois à l'autre. Ce détail accroît quelque peu la difficulté du jeu car il vous empêche d'acquérir un rythme régulier, et il vous demande d'être sans cesse "à l'affût" d'une nouvelle lettre.

Le programme affiche en permanence le temps écoulé. Notez bien que seul n'est pris en compte que votre temps de réaction. Le temps d'attente (variable) avant affichage d'une lettre n'est pas considéré. Vous voyez également apparaître le nombre d'erreurs que vous avez commises.

A la fin d'une partie, le programme vous indique quel a été votre temps moyen de réaction. Il vous fournit un score qui est fonction de ce temps moyen et du nombre d'erreurs. En même temps, il vous présente le meilleur score réalisé depuis le début du jeu.

Enfin, le programme vous demande si vous désirez rejouer. Si vous répondez O (pour oui), une nouvelle partie vous est alors proposée.

2. *Le programme*

Quelques conseils pour la frappe

Dans un premier temps, vous pouvez alléger la frappe en simplifiant les instructions REM, par exemple en vous limitant au mot REM. Cela concerne les lignes 100, 4000, 5000, 6000, 8000 et 9000. Ne supprimez pas complètement ces lignes car le programme ne fonctionnerait plus.

Attention à la ligne 8040. Le nombre d'espaces situés entre les guillemets ('') doit être exact. Pour éviter de vous tromper, repérez-vous sur la ligne 8020.

Attention, en ligne 160, ne pas laisser d'espace entre les deux guillemets :

```
100 REM ********** FORCE DE FRAPPE **********
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : FOR I=1 TO NL
150 :  GOSUB 5000
160 :  R$=KEY$: IF R$="" THEN GOSUB 6000: GOTO 160
170 :  R=ASC(R$): IF R<>C THEN PING: E=E+1: GOTO 160
180 :  GOSUB 4000
190 : NEXT I
```

```
200 : M=T/NL
220 : PRINT:PRINT:PRINT "TEMPS MOYEN DE FRAPPE :";M
230 : PRINT "AVEC ";E;" ERREURS"
240 : SC=INT(20000/M-50*E)
250 : PRINT: PRINT "VOTRE SCORE :";SC
260 : PRINT: PRINT "MEILLEUR SCORE :";MS
265 :  IF SC>MS THEN MS=SC
270 : PRINT: PRINT: PRINT "voulez vous rejouer (O/N)?"
280 : GET R$
290 UNTIL R$<>"O"
300 END
4000 REM ------ SP LETTRE CORRECTE ---------------------
4010 SHOOT
4020 PLOT X,Y,32
4030 RETURN
5000 REM ------ SP AFFICHAGE D'UNE LETTRE --------------
5010 X=X1+INT(RND(1)*(X2-X1+1))
5020 Y=Y1+INT(RND(1)*(Y2-Y1+1))
5040 C=65+INT(RND(1)*26)
5050 WAIT RND(1)*TM
5060 PLOT X,Y,C
5070 RETURN
6000 REM ------ SP MISE A JOUR, AFFICHAGE TEMPS ET ERREURS ----
6010 T=T+1
6020 PLOT 1,0,1: PLOT 2,0,"TEMPS"
6030 PLOT 8,0,STR$(T): PLOT 8,0," "
6040 PLOT 20,0,5: PLOT 21,0," ERREURS"
6050 PLOT 29,0,STR$(E): PLOT 29,0," "
6060 RETURN
8000 REM ------ SP INITIALISATION D'UNE PARTIE ---------
8010 T=0: E=0: CLS
8020 PLOT 3,0,"tapez sur une touche pour commencer"
8030 GET R$
8040 PLOT 3,0,"                                        "
8050 RETURN
9000 REM ------ SP INITIALISATIONS DU JEU ---------------
9010 POKE #26A,10
9020 PA=6: IN=4: PAPER PA: INK IN
9030 X1=2: X2=38: Y1=2: Y2=26
9040 NL=30: TM=75: MS=0
9050 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier le nombre de lettres proposées au cours d'une partie

Remplacez en 9040, le nombre 30 par une valeur de votre choix.

2. Pour modifier le temps s'écoulant avant l'apparition d'une nouvelle lettre

Remplacez, en 9040, le nombre 75 par une valeur de votre choix. Si vous souhaitez qu'il n'y ait aucun temps d'attente, vous pouvez :

— soit donner la valeur 0 à TM (en 9040),

— soit supprimer l'instruction 5050.

3. Pour que la lettre s'affiche toujours au même endroit de l'écran

Remplacez les instructions 5010 et 5020 par des instructions attribuant une valeur déterminée à X et Y. Par exemple :

```
5010 X = 20
5020 Y = 13
```

vous fourniront une lettre située au centre de l'écran.

4. Pour modifier la couleur de l'écran

Remplacez la valeur 6 de l'instruction 9020 par un nombre de votre choix (entre 0 et 7).

5. Pour modifier la couleur des lettres

Remplacez la valeur 4 (en 9020) par un nombre de votre choix (entre 0 et 7).

REMARQUE : Évitez de choisir la même valeur pour l'écran et pour les lettres car alors ces dernières seraient "invisibles".

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— HASARD
— Configuration clavier/écran

b) Le programme principal

110 appelle le sous-programme d'initialisation du jeu (en 9000).

120 et 290 permettent de répéter les instructions 130 à 280 correspondant au déroulement d'une partie, et ceci jusqu'à ce que l'utilisateur précise qu'il ne désire plus continuer à jouer :

140 à 190 proposent les trente lettres. Pour chaque lettre :

150 appelle le sous-programme 5000 qui affiche une lettre sur l'écran.

160 boucle sur elle-même jusqu'à ce qu'une touche soit pressée. Tant que cela n'a pas lieu, le sous-programme 6000 est appelé pour "incrémenter" le compteur de temps et pour en afficher sa valeur.

170-180 analysent la réponse. Si celle-ci est incorrecte, un bruit de sonnette retentit ; le compteur d'erreurs est incrémenté de un et l'on revient en 160 pour attendre une autre proposition. Si la réponse est correcte, le sous-programme 4000 fait retentir un coup de fusil et efface la lettre de l'écran.

200-230 calculent le temps moyen et l'impriment en même temps que le nombre d'erreurs.

240-265 calculent le score obtenu et l'impriment en même temps que le meilleur score réalisé jusqu'ici. Le cas échéant, le meilleur score est actualisé.

270-280 demandent à l'utilisateur s'il désire rejouer et lisent sa réponse.

c) Le sous-programme d'initialisation du jeu (9000-9050)

9010 supprime le bip du clavier et le curseur de l'écran.

9020 fixe les couleurs de fond et d'encre.

9030 fixe les limites du domaine où peuvent apparaître les lettres proposées.

9040 fixe le nombre total de lettres proposées, le temps d'attente maximum et initialise le meilleur score.

d) Le sous-programme d'initialisation d'une partie (8000-8050)

8010 initialise les différents compteurs (temps, erreurs) et efface l'écran.

8020-8040 attendent que vous frappiez une touche pour préciser que vous êtes prêt à commencer la partie.

e) Le sous-programme d'affichage d'une lettre (5000-5070)

5010-5020 déterminent au hasard les coordonnées d'affichage d'une lettre.

5040 choisit au hasard le code ASCII de l'une des vingt-six lettres de l'alphabet.

5050-5060 affichent la lettre choisie à l'emplacement déterminé, après un temps tiré lui aussi au hasard.

f) Le sous-programme lettre correcte (4000-4030)

4010 émet un bruit de "coup de fusil".

4020 efface la lettre proposée.

g) Le sous-programme d'affichage du temps et des erreurs (6000-6060)

6010 incrémente le compteur de temps.

6020-6030 affichent le temps.

6040-6050 affichent le nombre d'erreurs.

5. *Liste des variables*

PA	couleur de fond
IN	couleur d'encre (dans laquelle apparaissent les différentes lettres
X1,X2,Y1,Y2	coordonnées du domaine dans lequel les lettres pourront apparaître
NL	nombre total de lettres proposées au cours d'une partie
TM	temps maximum (exprimé en dizaines de ms) s'écoulant avant l'apparition d'une nouvelle lettre
MS	meilleur score
T	temps (cumulé) de réaction du joueur
E	compteur d'erreurs
I	numéro de la lettre courante
X,Y	coordonnées de la lettre courante
C	code ASCII de la lettre courante
R	code ASCII de la touche frappée par l'utilisateur
M	temps moyen de réaction
SC	score

Variables accessoires

R$

3

Raid de nuit

1. Le jeu

En pleine nuit, vous êtes aux commandes d'un bombardier avec pour mission d'anéantir une ville. Le temps vous est compté car l'un de vos moteurs est défaillant et vous perdez sans cesse de l'altitude. Il vous faut donc rapidement démolir les immeubles les plus élevés pour éviter de les percuter lors d'un prochain passage. D'autre part, vous ne pouvez larguer une nouvelle bombe que lorsque la précédente a atteint son but (ou le sol).

Vous voyez donc votre avion traverser l'écran de gauche à droite à diverses reprises. A chaque passage, il peut perdre ou non (suivant le hasard) un peu d'altitude. Le "suspens" en sera d'autant plus grand quand vous serez passé à ras d'un immeuble que vous aurez raté !

Vous larguez une bombe en appuyant sur la barre d'espace. Chaque fois qu'un immeuble est atteint, il est démoli sur une hauteur plus ou moins importante. Des sons d'explosion appropriés vous renseignent sur l'importance des dégâts.

Le jeu s'achève lorsque votre avion percute un immeuble ou lorsque la ville a été entièrement détruite. Dans le premier cas, votre score est fonction de l'importance des destructions opérées. Dans le second cas, il dépend de la vitesse avec laquelle vous avez atteint votre objectif (il sera toujours plus élevé dans le second cas).

Neuf niveaux différents vous sont proposés. Au niveau 1, les immeubles sont de petite taille tandis qu'au niveau 9, ils occupent une bonne partie de l'écran.

Le programme affiche en permanence le niveau auquel vous jouez et le meilleur score déjà réalisé à ce niveau : (voir photo 1 hors texte : *Vue de l'écran pendant le déroulement du jeu*).

2. Le programme

```
100 REM ********** RAID DE NUIT ******************
105 GOSUB 9000
110 REPEAT
120 : GOSUB 8000
130 : REPEAT
140 :  IF TR=1 THEN GOSUB 2000 ELSE WAIT 5
150 :  IF PEEK(#208)=#84 AND TR=0 THEN XB=XA: YB=YA: TR=1
160 :  GOSUB 3000: T=T+1
165 :  IF TR=1 AND EX=0 THEN GOSUB 2000
170 : UNTIL FI=1 OR ID=IM
180 : IF ID=IM THEN GOSUB 5000
190 : PLOT  8,26,STR$(SC): PLOT  8,26," "
195 : IF SC>MS(NV) THEN MS(NV)=SC
200 : WAIT 200
210 : PLOT 24,24,"voulez vous"
220 : PLOT 24,25,"rejouer (O/N)"
230 : GET R$: GET R$
240 UNTIL R$<>"O"
260 END
```

```
2000 REM ---- DEPLACEMENT BOMBE ------------------
2005 IF EX=1 THEN 2050
2010 PLOT XB,YB,32: YB=YB+1
2020 IF YB>Y2 THEN SHOOT: TR=0: RETURN
2030 IF SCRN(XB,YB)=32 THEN PLOT XB,YB,CB: RETURN
2040 EX=1
2050 EXPLODE: PLOT XB,YB,32: ID=ID+1: YB=YB+1
2070 IF RND(1)<PR OR YB>Y2 THEN EX=0: TR=0: RETURN
2080 RETURN
3000 REM ---- DEPLACEMENT AVION ------------------
3010 IF XA>=X2 THEN 3050
3020 IF SCRN(XA+2,YA)=CI THEN GOSUB 4000: RETURN
3030 PLOT XA,YA,"  ": XA=XA+1: PLOT XA,YA,CA$
3040 RETURN
3050 PLOT XA,YA,"  ": IF RND(1)>0.33 THEN YA=YA+1
3060 IF YA>Y2 THEN GOSUB 4000
3070 XA=X1
3080 IF SCRN(XA,YA)=CI OR SCRN(XA+1,YA)=CI THEN GOSUB 4000: RETURN
3090 PLOT XA,YA,CA$
3100 RETURN
4000 REM ---- SP FIN PARTIE PAR DESTRUCTION AVION -----
4010 SHOOT: WAIT 10:
4020 PLOT XA+2,YA,IN
4030 FOR K=1 TO 7
4040 : PLOT XA-1,YA,K: WAIT 8: EXPLODE
4050 NEXT K
4060 WAIT 50: EXPLODE: WAIT10: PLOT XA,YA,"  "
4070 SC=INT(1000*ID/IM): FI=1: RETURN
5000 REM ---- SP FIN PARTIE PAR DESTRUCTION IMMEUBLES ----
5010 WAIT 100: SC=1000+(24-YA)*100
5020 RETURN
8000 REM ---- INITIALISATION D'UNE PARTIE ----------
8010 CLS: PLOT 1,25,"Niveau choisi (1 a 9)?"
8020 GET R$: NV=VAL(R$): IF NV=0 THEN 8020
8030 PLOT 26,25,R$
8033 FOR Y=24 TO 26
8035 : PLOT 0,Y,I2
8037 NEXT Y
8040 YB=23-NV: HM=0.5*NV+5: IM=0
8050 FOR X=X1+2 TO X2-1
8060 : YM=YB-INT(RND(1)*HM)
8070 : IM=IM+Y2-YM+1
8080 : FOR Y=Y2 TO YM STEP -1
8090 :  PLOT X,Y,CI
8100 : NEXT Y
8110 NEXT X
```

```
8120 ID=0: TR=0: FI=0
8130 XA=X1: YA=Y1: XB=XA: YB=YA
8140 PLOT XA,YA,CA$
8160 PLOT 1,25,BL$
8170 PLOT 1,25,"NIVEAU SCORE  M-SCORE"
8180 PLOT 3,26,STR$(NV): PLOT 3,26," "
8185 PLOT 15,26,STR$(MS(NV)): PLOT 15,26," "
8190 RETURN
9000 REM ----- INITIALISATIONS DU JEU --------------
9010 PA=0: IN=2: I2=6
9020 PAPER PA: INK IN
9030 DIM MS(9)
9060 X1=1: X2=37: Y1=0: Y2=23
9070 CI=64+128: CB=92: C1=94: C2=95
9080 DATA 63,45,45,63,63,45,45,63
9090 C=CI-128: GOSUB 9500
9100 DATA 0,0,0,0,8,7,8,0
9110 C=CB: GOSUB 9500
9120 DATA 56,36,35,32,39,63,7,15
9130 C=C1: GOSUB 9500
9140 DATA 0,0,60,2,57,63,32,0
9150 C=C2: GOSUB 9500
9160 CA$=CHR$(C1)+CHR$(C2)
9170 POKE #26A,10
9180 PR=0.6
9190 BL$=""
9200 FOR I=1 TO 36: BL$=BL$+" ": NEXT I
9210 RETURN
9500 REM ---- INIT CARAC GRAPHIQUES ---------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier la couleur du fond (ici noir)

Remplacez, en 9010, la valeur 0 (dans PA = 0) par un nombre de votre choix (0 à 7). Notez que cette modification agit sur la couleur des immeubles.

2. Pour modifier la couleur de l'avion et de la bombe

Remplacez, en 9010, la valeur 2 (dans IN = 2) par un nombre de votre choix (0 à 7). Notez que cette modification agit sur la couleur des immeubles.

3. Pour que l'avion perde de l'altitude plus ou moins rapidement

Il vous faut savoir que la perte d'altitude est régie par l'instruction 3050. Actuellement l'avion a, à chaque passage, une chance sur trois (0.33) de ne pas descendre.

Si vous souhaitez que l'avion descende plus vite, remplacez la valeur 0.33 par un nombre plus petit, par exemple:

```
3050 PLOT XA,YA,"  ":IF RND(1)>0.15 THEN YA = YA + 1
```

Si vous voulez que l'avion descende à chaque passage, il vous suffit d'écrire:

```
3050 PLOT XA,YA,"  ":YA = YA + 1
```

Si vous souhaitez que l'avion descende plus lentement, remplacez 0.33 par un nombre plus grand, tout en étant inférieur à 1. Par exemple, avec:

```
9050 PLOT XA,YA,"  ":IF RND(1)>0.5 THEN YA = YA + 1
```

l'avion descendra, en moyenne, une fois sur deux.

Évitez de choisir des valeurs trop proches de 1 car l'avion descendrait alors trop lentement et le jeu risquerait de perdre de l'intérêt.

4. Description du programme

a) Techniques utilisées, décrites en annexe

— Hasard
— Création de caractères graphiques

— Lecture rapide du clavier
— Configuration clavier/écran

b) Le programme principal (100-260)

105 appelle le sous-programme d'initialisation du jeu.

110 et 240 répètent les instructions 120 à 230 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer.

120 appelle le sous-programme d'initialisation d'une partie.

130 et 170 répètent les instructions 140 à 165 jusqu'à la fin de la partie (écrasement de l'avion ou destruction de toute la ville).

140 assure le déplacement de la bombe si un larguage est en cours (TR = 1).

150 prend en compte une demande de larguage (barre d'espace) si aucun autre larguage n'est en cours.

160 déplace l'avion et actualise le temps.

165 déplace la bombe, s'il y a lieu.

180 appelle le sous-programme de fin de partie par destruction de la ville s'il y a lieu.

190-195 affichent votre score et actualisent le meilleur score.

200-230 vous demandent si vous désirez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9210)

9010-9020 fixent les différentes couleurs (fond, encre, lignes inférieures).

9030 réserve le tableau des meilleurs scores.

9060 fixe les limites du domaine de jeu.

9070-9160 fabriquent les caractères graphiques utilisés pour représenter l'avion, la bombe et les immeubles.

9170 supprime le bip du clavier et le curseur.

d) Le sous-programme d'initialisation de partie (8000-8190)

8010-8030 vous font choisir le niveau de jeu.

8033-8037 modifient la couleur d'encre des trois lignes inférieures.

8040-8110 dessinent les immeubles.

8120 initialise le nombre d'immeubles détruits, les indicateurs de larguage et de fin de partie.

8130-8140 dessinent l'avion dans sa position initiale.

8160-8185 affichent le niveau de jeu et le meilleur score.

e) Le sous-programme de déplacement de la bombe (2000-2080)

(Ce sous-programme gère également l'explosion d'immeuble).

2005-2040 avancent la bombe lorsqu'aucune explosion n'est en cours; examinent si elle atteint le sol ou un immeuble. Dans ce dernier cas, 2040 met en place l'indicateur "explosion en cours".

2050-2070 exécutées lorsqu'une explosion est en cours, ces instructions incrémentent le compteur d'immeubles détruits, effacent un "morceau" d'immeuble et déterminent si l'explosion doit ou non se poursuivre sur la partie inférieure de l'immeuble.

f) Le sous-programme de déplacement d'avion (3000-3100)

3010-3040 déplacent l'avion s'il n'a pas atteint le bord droit de l'écran, en regardant s'il percute un immeuble. Dans ce dernier cas le sous-programme 4000 est appelé.

3050-3100 réaffichent l'avion à gauche de l'écran, après avoir éventuellement modifié son altitude.

g) Les sous-programmes de fin de partie (4000-4070 et 5000-5020)

4010-4060 explosion (couleurs et sons) de l'avion.

4070 calcul du score (en cas de fin par percution d'immeuble).

5010 calcul du score (en cas de fin par ville détruite).

5. Liste des variables

PA	couleur du fond
IN	couleur de l'encre pour le domaine de jeu

I2	couleur de l'encre pour les trois lignes inférieures
MS(9)	tableau contenant le meilleur score relatif à chacun des neuf niveaux
X1,X2,Y1,Y2	coordonnées du domaine de jeu
CI	code du caractère utilisé pour dessiner les immeubles
CB	code du caractère utilisé pour dessiner la bombe
C1	code du caractère utilisé pour dessiner la partie gauche de l'avion
C2	code du caractère utilisé pour dessiner la partie droite de l'avion
CA$	chaîne de deux caractères représentant l'avion
PR	probabilité pour que l'explosion d'une partie d'un immeuble entraîne l'explosion de la partie inférieure
BL$	chaîne de 36 caractères "blanc" utilisée pour l'effacement d'une ligne d'écran
NV	niveau de jeu
IM	nombre d'immeubles (plus précisément de "cases" représentant les immeubles)
ID	nombre de "cases immeubles" détruites
TR	si 1 : larguage en cours (une bombe est en train de tomber) si 0 : aucun larguage en cours
FI	indicateur de fin de partie : 0 : cas normal 1 : fin de partie
XA,YA	coordonnées de l'avion
XB,YB	coordonnées de la bombe
EX	indicateur explosion en cours : 0 : pas d'explosion 1 : explosion en cours
SC	score

Variables auxiliaires

R$ X Y C I
HM,YB (dans le sous-programme 8000)

4

FLIP FLOP

1. Le jeu

Vous disposez de neuf pions réversibles disposés suivant une grille 3 × 3. L'une de leur face est bleu clair tandis que l'autre est recouverte d'un quadrillage rouge et jaune.

Au départ, le programme dispose les pions dans la grille, la face visible étant tirée au hasard. Le but du jeu consiste à aboutir, en un minimum de coups à la situation suivante :

— le pion central est bleu

— les autres pions sont rouges/jaunes

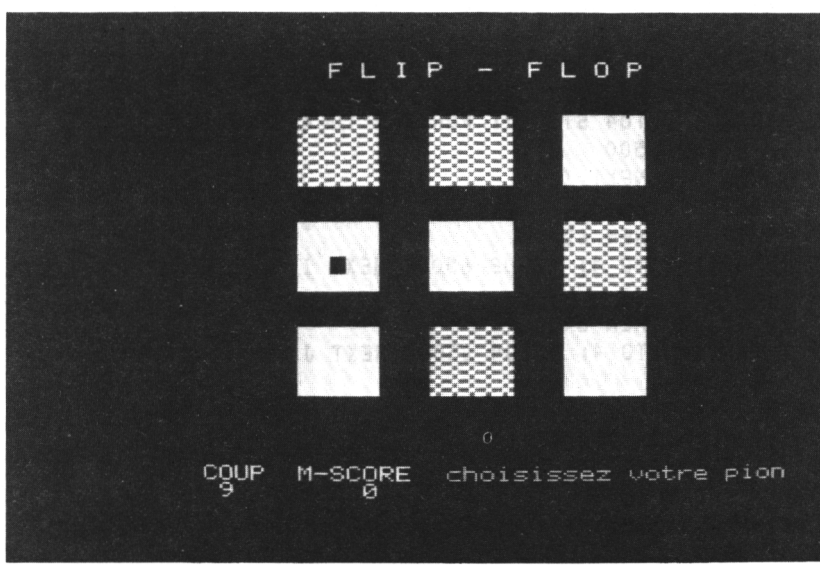
A chaque coup, vous choisissez le pion sur lequel vous voulez agir. Le programme vous retourne alors, non seulement ce pion, mais également un certain nombre d'autres, suivant les règles suivantes :

Pion choisi	*Pions retournés en plus du pion choisi*
Le pion central.	Les quatre pions des coins.
Un pion situé en milieu d'une rangée horizontale ou verticale.	Les deux pions voisins situés sur la même rangée.
Un pion situé dans un coin.	Les trois voisins immédiats.

Malgré une lointaine ressemblance avec le "rubik's cube", vous verrez que ce jeu est beaucoup plus facile à pratiquer.

Vous choisissez un pion en y amenant le curseur représenté par un petit carré bleu foncé situé à l'intérieur d'un pion. Ce "curseur" se déplace à l'aide des quatre touches fléchées.

Le programme vous donne en permanence le numéro du coup que vous jouez et le meilleur score déjà obtenu.



Vue de l'écran pendant le déroulement du jeu

2. Le programme

```
100 REM ********** FLIP FLOP ***************************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : REPEAT
150 :  NC=NC+1: PLOT 3,25,STR$(NC): RLOT 3,25," "
160 :  GOSUB 7000: GOSUB 6000: GOSUB 5000
170 : UNTIL NN=8 AND T(3,3)=1
175 : IF NC<MS OR MS=0 THEN MS=NC
180 : SHOOT: PLOT 24,24,"GAGNE": WAIT 100
190 : PLOT 20,25,12: PLOT 21,25,"rejouez vous?"
200 : GET R$: GET R$
210 UNTIL R$<>"O"
220 END
5000 REM ----- SP ANALYSE D'UN COUP ---------------
5010 NN=0
5020 FOR I=2 TO 4: FOR J=2 TO 4
5030 : IF T(I,J)=-1 THEN NN=NN+1
5040 NEXT J: NEXT I
5050 RETURN
6000 REM ----- SP JEU DU COUP PROPOSE -----------------
6008 IP=I: JP=J
6020 IF IP<>3 OR JP<>3 THEN 6100
6030 : GOSUB 6500
6040 : FOR I=2 TO4 STEP 2: FOR J=2 TO 4 STEP 2
6050 :  GOSUB 6500
6060 : NEXT J: NEXT I
6070 : RETURN
6100 IF IP<>3 THEN 6200
6110 : FOR I=2 TO 4: GOSUB 6500: NEXT I
6120 : RETURN
6200 IF JP<>3 THEN 6400
6210 : FOR J=2 TO 4: GOSUB 6500: NEXT J
6230 : RETURN
6400 REM
6410 : FOR I=IP-1 TO IP+1
6420 :  FOR J=JP-1 TO JP+1
6430 :   GOSUB 6500
6440 : NEXT J: NEXT I
6450 RETURN
```

```
6500 REM ----- SP INVERSION D'UN PION -----------------
6510 T(I,J)=-T(I,J)
6520 GOSUB 8500
6530 RETURN
7000 REM ----- SP LECTURE COUP PROPOSE ---------------
7010 PLOT 17,24,12: PLOT 18,24,"choisissez votre pion"
7020 I=2: J=2
7030 X=8*(I-2)+11: Y=6*(J-2)+6
7040 SC=SCRN(X,Y)
7050 PLOT X,Y,CC
7060 CL=PEEK(#208): IF CL=#38 THEN 7060
7070 IF CL=#84 THEN 7200
7080 I=I-(CL=#AC)*(I>2)+(CL=#BC)*(I<4)
7090 J=J-(CL=#9C)*(J>2)+(CL=#B4)*(J<4)
7100 PLOT X,Y,SC: WAIT 20: GOTO 7030
7200 PING: WAIT 100: PLOT X,Y,SC
7210 PLOT 17,24,"                        "
7220 RETURN
8000 REM ----- SP INITIALISATION D'UNE PARTIE ----------
8010 CLS: PLOT 10,1," F L I P  -  F L O P "
8015 NC=0
8020 NN=0
8030 FOR I=2 TO 4: FOR J=2 TO 4
8040 : T(I,J)=1: IF RND(1)>0.5 THEN T(I,J)=-1: NN=NN+1
8050 NEXT J: NEXT I
8060 IF T(3,3)=1 AND NN=8 THEN 8020
8070 PLOT 3,24,"COUP  M-SCORE"
8080 PLOT 12,25,STR$(MS): PLOT 12,25," "
8090 FOR I=2 TO 4: FOR J=2 TO 4
8100 : GOSUB 8500
8110 NEXT J: NEXT I
8120 RETURN
8500 REM ----- SP DESSIN PION I,J SI VALIDE -----------
8510 IF I<2 OR I>4 OR J<2 OR J>4 THEN RETURN
8520 XD=8*(I-2)+9: YD=6*(J-2)+4
8530 CH$=CP$: IF T(I,J)=-1 THEN CH$=CN$
8540 FOR Y=YD TO YD+3
8550 : PLOT XD,Y,CH$
8560 NEXT Y
8570 RETURN
9000 REM ----- SP INITIALISATION DU JEU ---------------
9010 IN=6: PA=4: CLS: PAPER PA: INK IN
9020 DIM T(5,5): MS=0
9030 CN=95+128: CP=96: CC=32
9040 DATA 51,51,12,12,51,51,12,12
```

```
9050 C=CN-128: GOSUB 9500
9060 DATA 63,63,63,63,63,63,63,63
9070 C=CP: GOSUB 9500
9080 CN$="": FOR I=1 TO 5: CN$=CN$+CHR$(CN): NEXT I
9090 CP$="": FOR I=1 TO 5: CP$=CP$+CHR$(CP): NEXT I
9100 POKE #26A,10
9110 RETURN
9500 REM ----- SP INIT CARAC GRAPHIQUES --------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

Pour modifier la couleur des pions

Il vous faut savoir que :

- le côté pile est représenté par la couleur d'encre placée en 9010, ici 6, c'est-à-dire cyan (bleu ciel)
- le côté face est représenté par un quadrillage en "vidéo inverse". Autrement dit, les deux couleurs qui le constituent sont :

  — le complémentaire de la couleur de fond (4), c'est-à-dire 7 — 4 = 3 (jaune)

  — le complémentaire de la couleur de papier (6), c'est-à-dire 7 — 6 = 1 (rouge)

Par exemple, si vous remplacez 9010 par :

```
IN=3: PA=0: CLS: PAPER PA: INK IN
```

vous obtiendrez :

— un fond noir (0)
— du jaune (3) pour le côté pile
— un quadrillage bleu (7 — 3 = 4) et blanc (7 — 7 = 0) pour le côté face.

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard
— Création de caractères graphiques
— Lecture rapide du clavier
— Configuration clavier/écran

b) Le programme principal (100-220)

110 appelle le sous-programme d'initialisation du jeu.

120 et 210 répètent les instructions 130 à 200 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

130 appelle le sous-programme d'initialisation d'une partie.

140 et 170 répètent les instructions 150 et 160 (jeu d'un coup) jusqu'à ce que vous ayez gagné:

150 incrémente le compteur de coups et affiche sa valeur,

160 appelle les sous-programmes: lecture du coup, jeu du coup (inversion des pions correspondants), analyse du coup.

180-200 vous disent que vous avez gagné, actualisent le meilleur score et vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9110)

9010 fixe les couleurs de fond et de papier.

9030-9090 fabriquent les caractères graphiques représentant les deux faces d'un pion.

9100 supprime le bip du clavier et le curseur d'écran.

d) Le sous-programme d'initialisation d'une partie (8010-8120)

8010-8015 effacent l'écran, affichent le nom du jeu et initialisent le compteur de coups.

8020-8060 choisissent au hasard la position (pile ou face) de chaque pion et comptent le nombre de côtés "face". On évite la situation gagnante.

8070-8080 affichent le meilleur score.

8090-8110 dessinent les pions en utilisant le sous-programme 8500.

e) Le sous-programme de dessin d'un pion (8500-8570)

Il détermine la position sur l'écran d'un pion de coordonnées I et J et le dessine. Notez que, pour des raisons de simplification du programme, on utilise une grille de 5 × 5. Seules les cases correspondant à I = 2,3,4 et J = 2,3,4 représentent effectivement des pions.

f) Le sous-programme de lecture du coup proposé (7000-7220)

7010 affiche en clignotant "choisissez votre pion".

7020-7100 vous permettent de déplacer le curseur grâce aux touches fléchées. Ces instructions sont répétées jusqu'à ce que vous frappiez la barre d'espace.

7200-7210 émettent un son de clochette pour montrer que votre coup a été pris en compte et effacent le curseur.

g) Le sous-programme de jeu du coup proposé (6000-6450)

6030-6070 cas du pion central.

6100-6120 cas d'un pion situé au centre d'une colonne.

6200-6230 cas d'un pion situé au centre d'une ligne.

6400-6450 cas d'un pion situé dans un coin.

h) Le sous-programme d'analyse d'un coup (5000-5050)

Il compte le nombre de pions en position "face" (rouge et jaune).

5. *Liste des variables*

IN couleur d'encre

PA couleur de fond

T(5,5) tableau représentant la situation de chaque pion :
1 : côté pile (bleu)
— 1 : côté face (rouge et jaune)
Notez que pour des raisons de simplification, nous employons un tableau 5 × 5 et non 3 × 3. Seuls les éléments T(i,j) ($2 \leqslant i \leqslant 4$; $2 \leqslant j \leqslant 4$) correspondent aux pions effectivement dessinés sur l'écran.

MS meilleur score

CN code ASCII du caractère servant à dessiner les pions, côté face

CP code ASCII du caractère servant à dessiner les pions, côté pile

CC code ASCII du caractère servant à dessiner le curseur

CN$ chaîne de cinq caractères représentant une partie d'un pion, côté face

CP$ chaîne de caractères représentant une partie d'un pion, côté pile

NC numéro du coup

NN nombre de pions côté face

XD,YD coordonnées écran du coin haut gauche d'un pion

I,J coordonnées d'un pion (varient entre 2 et 4)

IP,JP coordonnées du pion choisi (varient entre 2 et 4)

X,Y coordonnées du curseur

SC score

Variables auxiliaires

C CH$ CL

5

Drôles de mines

1. Le jeu

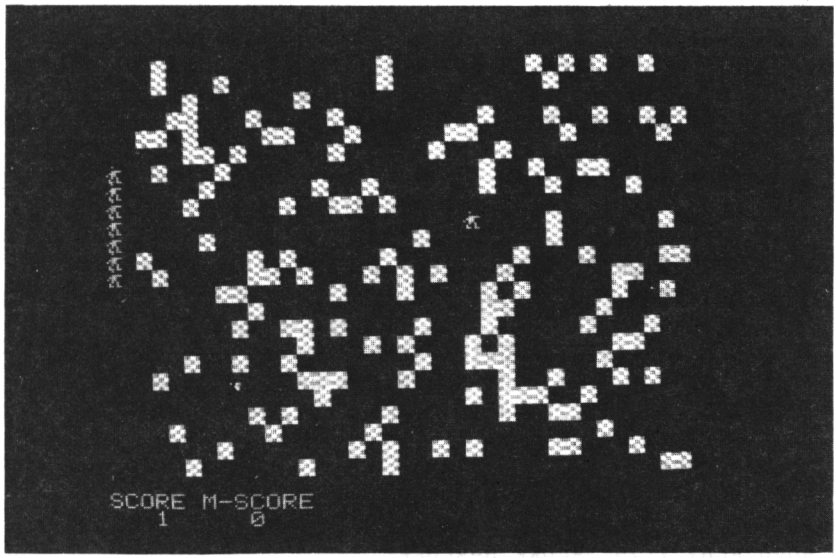
Ce sont effectivement de drôles de mines qu'il va vous falloir déjouer. En effet, contrairement aux mines habituelles, elles sont capables de "sauter" une multitude de fois.

Votre mission : faire parvenir un maximum d'espions en territoire ennemi. Pour cela, ils doivent obligatoirement traverser un domaine infesté de mines. Vous disposez de douze hommes, prêts à risquer leur vie et qui suivent à la lettre l'itinéraire que vous leur indiquez par radio.

Vos douze espions sont alignés sur la gauche de l'écran. Le premier attend que vous le dirigiez à l'aide des quatre touches fléchées (ou, le cas échéant, de la poignée de jeu). Vous devez tenter de le faire parvenir à droite de l'écran (à n'importe quelle hauteur). S'il saute sur une mine (avec sons et décors appropriés), vous verrez aussitôt l'espion suivant se présenter au départ.

Les mines sont, bien entendu, invisibles. Par contre, de nombreux obstacles, infranchissables, vous serviront de repère.

Bonne chance et n'oubliez pas que ces mines sont "drôles".



Vue de l'écran pendant le déroulement du jeu

2. *Le programme*

```
100 REM ********** DROLE DE MINE ********************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : FOR I=1 TO NV
150 :  X=XI: Y=YI: PLOT X,Y,CB: PLOT 1,I,32
160 :  CL=PEEK(#208)
170 :  IF CL=#38 THEN 160
180 :  XN=X-(CL=#AC)*(X>XI)+(CL=#BC)*(X<38)
```

```
190 :  YN=Y-(CL=#9C)*(Y>Y1)+(CL=#B4)*(Y<Y2)
200 :  SX=SCRN(XN,YN)
210 :  IF SX=CM THEN GOSUB 3000: GOTO 260
215 :  PLOT X,Y,32
220 :  IF SX<>CO THEN X=XN: Y=YN
230 :  PLOT X,Y,CB
240 :  IF X=38 THEN GOSUB 2000: GOTO 260
250 :  WAIT 20: GOTO 160
260 : NEXT I
265 : IF SC>MS THEN MS=SC
270 : PLOT 10,25,"voulez vous rejouer (O/N)"
280 : GET R$: GET R$
290 UNTIL R$<>"O"
310 END
2000 REM ---- SP TRAVERSEE REUSSIE -----------------
2010 ZAP: SC=SC+1
2020 PLOT 3,26,STR$(SC): PLOT 3,26," "
2030 WAIT 50: PLOT X,Y,32
2040 RETURN
3000 REM ---- SP EXPLOSION MINE -------------------
3010 PLOT X,Y," ": PLOT XN,YN,CB: WAIT 20: EXPLODE
3020 PLOT XN,YN,C2: WAIT 40: PLOT XN,YN,C3: WAIT 60
3030 PLOT XN,YN,CM
3040 RETURN
8000 REM ---- SP INIT D'UNE PARTIE -------------------
8010 CLS: SC=0
8020 PLOT 1,25,"SCORE M-SCORE"
8030 PLOT 9,26,STR$(MS): PLOT 9,26," "
8040 FOR I=1 TO NO
8050 : X=X1+INT(RND(1)*(X2-X1+1))
8060 : Y=Y1+INT(RND(1)*(Y2-Y1+1))
8070 : IF SCRN(X,Y)<>32 THEN 8050
8080 : PLOT X,Y,CO
8090 NEXT I
8100 FOR I=1 TO NM
8110 : X=X1+INT(RND(1)*(X2-X1+1))
8120 : Y=Y1+INT(RND(1)*(Y2-Y1+1))
8130 : IF SCRN(X,Y)<>32 THEN 8110
8140 : PLOT X,Y,CM
8150 NEXT I
8160 FOR I=1 TO NV
8170 : PLOT 1,I,CB
8180 NEXT I
8190 RETURN
```

```
9000 REM ---- SP INITIALISATIONS --------------------
9010 IN=1: PA=3: INK IN: PAPER PA
9020 X1=3: X2=36: Y1=0: Y2=23: XI=2: YI=15
9030 NV=12: NO=150: NM=40
9040 CB=64: CO=94+128: CM=IN: C2=95: C3=96
9050 DATA 8,0,15,28,44,10,18,27
9060 C=CB: GOSUB 9500
9070 DATA 51,51,12,12,51,51,12,12
9080 C=CO-128: GOSUB 9500
9090 DATA 20,23,0,12,8,2,34,59
9100 C=C2: GOSUB 9500
9110 DATA 2,16,1,4,32,1,20,1
9120 C=C3: GOSUB 9500
9130 POKE #26A,10
9140 RETURN
9500 REM ---- SP INIT CARAC GRAPHIQUES ---------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier le nombre d'espions

Remplacez le nombre 12 de l'instruction 9030 par une valeur de votre choix ne dépassant pas 24.

2. Pour modifier le nombre d'obstacles

Remplacez le nombre 150 de l'instruction 9030 par la valeur de votre choix.

3. Pour modifier le nombre de mines

Remplacez le nombre 40 de l'instruction 9030 par la valeur de votre choix.

Notez bien qu'un trop grand nombre de mines risque de rendre le jeu difficile, voir impossible. Un trop petit nombre risque, par contre, de le rendre trop facile.

4. Pour que les mines n'explosent qu'une seule fois

Remplacez 3030 par:

```
3030 PLOT XN,YN," "
```

5. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1" comme expliqué en annexe. Branchez la poignée du côté droit et remplacez les instructions 160 à 190 par les suivantes:

```
160 :  C=PEEK(#400)
170 :  IF C=191 THEN 160
180 :  XN=X-((C AND 1)=0)*(X>XI)+((C AND 2)=0)*(X<38)
190 :  YN=Y-((C AND 16)=0)*(Y>Y1)+((C AND 8)=0)*(Y<Y2)
```

4. *Description du programme*

a) Techniques utilisées décrites en annexe

— Hasard
— Création de caractères graphiques
— Lecture rapide du clavier
— Configuration clavier/écran
— Poignées de jeu

b) Le programme principal

110 appelle le sous-programme d'initialisation du jeu.

120 et 290 répètent les instructions 130 à 280 (déroulement d'une partie)

jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

130 appelle le sous-programme d'initialisation d'une partie.

140-260 répètent douze fois la tentative de traversée d'un espion. Pour chaque tentative:

150 dessine l'espion au départ, et efface un espion de la file d'attente.

160-250 sont répétées jusqu'à ce que l'espion ait sauté sur une mine ou qu'il ait réussi sa traversée:

160-190 attendent qu'une touche soit enfoncée et déterminent la position correspondante.

200-230 si une mine se trouve à cette nouvelle position, le sous-programme 3000 simule l'explosion de la mine. Dans le cas contraire et s'il ne s'agit pas d'un obstacle, l'espion est dessiné dans sa nouvelle position après avoir été effacé de l'ancienne.

240 appelle le sous-programme traversée "réussie" si l'espion atteint le bord droit.

270-280 vous demandent si vous désirez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9140)

9010 fixe les couleurs d'encre et de fond.

9020 fixe les limites du domaine qui contiendra les obstacles et les mines.

9030 fixe le nombre d'espions, le nombre d'obstacles et le nombre de mines.

9040-9120 fabriquent les caractères graphiques utilisés pour représenter les espions et les obstacles.

9130 supprime le "bip" et le curseur.

d) Le sous-programme d'initialisation d'une partie (8000-8190)

8010-8030 initialisent le score, effacent l'écran et affichent le meilleur score.

8040-8090 dessinent au hasard les différents obstacles.

8100-8150 placent les bombes au hasard.

8160-8180 dessinent la file d'attente d'espions.

e) Le sous-programme d'explosion d'une mine (3000-3030)

3010 dessine d'abord l'espion à l'emplacement voulu et, après une brève attente, fait retentir une explosion.

3020 simule la "destruction" de l'espion en faisant se succéder trois caractères différents: "morceaux", "poussières", "espace".

f) Le sous-programme "traversée réussie" (2000-2040)

2010 émet un son "zap" et incrémente le score.

2020 affiche le score.

2030 efface l'espion qui vient de traverser.

5. Liste des variables

IN	couleur d'encre
PA	couleur de fond
X1,X2,Y1,Y2	coordonnées du domaine dans lequel se trouvent les obstacles et les mines
XI,YI	coordonnées de la position de départ d'un espion
NV	nombre d'espions
NO	nombre d'obstacles
NM	nombre de mines
CB	code du caractère représentant un espion
CO	code du caractère représentant un obstacle
CM	code du caractère représentant une mine (il doit être "invisible") (Nous avons choisi le code de la couleur de l'encre)
C2,C3	codes des caractères représentant l'espion (ou plutôt ses "restes") au cours d'une explosion
SC	score (nombre d'espions traversés)
I	numéro de l'espion en cours de traversée

X,Y	coordonnées de l'espion en cours de traversée
CL	code du caractère frappé au clavier
XN,YN	coordonnées du prochain emplacement de l'espion
MS	meilleur score

Variables auxiliaires

X,Y (de 8040 à 8150) coordonnées d'un obstacle ou d'une mine
C SX R$
I (dans le sous-programme 8000)

6

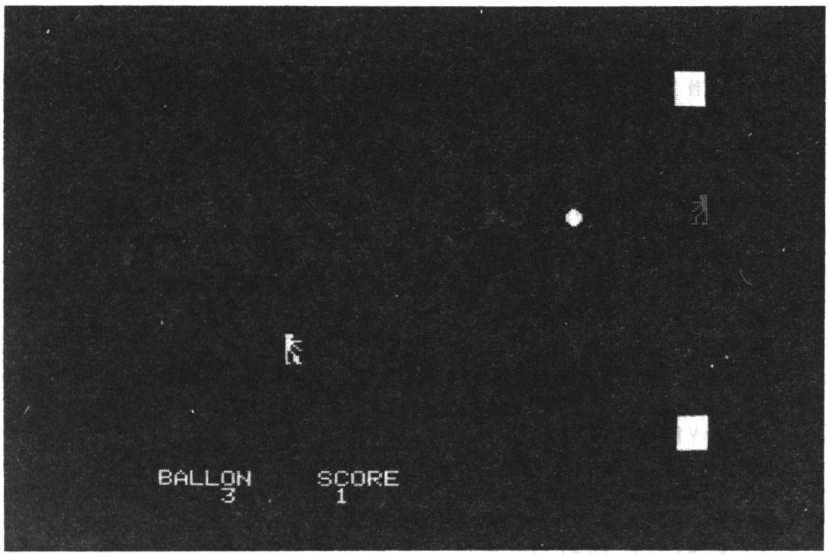
GOAL

1. Le jeu

Vous venez d'être sélectionné comme goal dans une grande équipe de football. Saurez-vous être à la hauteur de la situation en interceptant tous les tirs de vos adversaires qui se montrent particulièrement adroits.

Les buts sont matérialisés par deux poteaux sur la droite de l'écran. Vous déplacez le goal, entre les deux poteaux à l'aide des deux touches " ↑ " et " ↓ ". Douze tirs au but vont avoir lieu. Pour chaque tir, vous voyez apparaître en un endroit quelconque de l'écran, un joueur accompagné d'un ballon. Au bout d'un court instant (variable d'une fois à l'autre), le joueur "shoote" dans le ballon et il vous faut l'intercepter. Vous ne pouvez déplacer le goal que pendant le déplacement du ballon. Comme vous le constaterez, vos adversaires sont très habiles puisqu'ils tirent toujours dans les buts.

Le programme affiche le nombre de coups tirés et le nombre de ballons interceptés.



Vue de l'écran pendant le déroulement du jeu

2. *Le programme*

```
100 REM *********** GOAL ************************
110 SC=0: GOSUB 9000
120 FOR IC=1 TO NC
130 : GOSUB 5000
140 : IF PEEK(768)=176 THEN WAIT 6:GOTO 190
150 : DL=PEEK(735)-135
160 : PLOT XG,YG,"  ": PLOT XG,YG+1,"  "
170 : YG=YG+(DL=4)*(YG>P1+1)*(YG<>Y OR XG-1<>X)
175 : YG=YG-(DL=3)*(YG<P2-2)*(YG<>Y OR XG-1<>X)
180 PLOT XG,YG,G1$: PLOT XG,YG+1,G2$
190 : PLOT X,Y,32
200 : X=X+2: Y=Y+DY
```

```
210 : IF X>=XG THEN GOSUB 4000: GOTO 240
215 : IF SCRN(X,Y)<>32 THEN GOSUB 4000: GOTO 240
220 : PLOT X,Y,CB
230 : GOTO 140
240 NEXT IC
250 PRINT CHR$(6);CHR$(17)
260 END
4000 REM ... SP FIN D'UN COUP .....................
4008 IF SCRN(X,Y)=32 THEN ZAP: GOTO 4050
4020 SC=SC+1
4030 SOUND 4,25,0: PLAY 0,1,1,500
4040 PLOT 14,25,STR$(SC): PLOT 14,25," "
4050 WAIT 100
4060 PLOT XJ,YJ,32: PLOT XJ,YJ+1,32
4070 WAIT 100
4080 RETURN
5000 REM ... SP INITIALISATION D'UN COUP .............
5010 PLOT 7,25,STR$(IC): PLOT 7,25," "
5020 X=X1+2*INT(RND(1)*(X2-X1+1)/2)
5030 Y=Y1+INT(RND(1)*(Y2-Y1)+1)
5040 DY=INT(RND(1)*6)-2
5050 YI=Y+(38-X)*DY/2
5060 IF YI<=P1 OR YI>=P2 THEN 5020
5065 XJ=X-1: YJ=Y-1
5070 PLOT X,Y,CB
5080 PLOT X-1,Y-1,J1
5090 PLOT X-1,Y,J3
5095 PLOT X,Y,CB
5100 PLOT 25,25,12: PLOT 26,25,"ATTENTION"
5110 WAIT RND(1)*TM+100
5120 PLOT 25,25,"           "
5125 PLOT X-1,Y,J2
5130 SOUND 4,25,0: PLAY 0,1,1,600
5140 RETURN
9000 REM .... SP INITIALISATIONS ...................
9010 PA=3: IN=4
9020 PAPER PA: INK IN
9030 PP=5: IG=1
9040 XG=36: YG=11: TM=100
9050 NC=12
9060 X1=3: X2=15
9070 Y1=1: Y2=23
9080 P1=2: P2=21
9090 CB=36: G1=64: G2=94
9100 J1=95: J2=96: J3=123
9110 DATA 0,7,7,3,63,7,11,19
9120 C=G1: GOSUB 9500
```

```
9130 DATA 35,5,9,17,17,17,51,0
9140 C=G2: GOSUB 9500
9150 DATA 0,56,56,48,63,56,52,50
9160 C=J1: GOSUB 9500
9170 DATA 49,40,36,38,38,38,51,0
9180 C=J2: GOSUB 9500
9183 DATA 12,30,63,63,63,63,30,12
9185 C=CB: GOSUB 9500
9186 DATA 49,40,40,40,40,40,60,0
9188 C=J3: GOSUB 9500
9190 G1$=CHR$(IG)+CHR$(G1)
9200 G2$=CHR$(IG)+CHR$(G2)
9210 PRINT CHR$(6);CHR$(17);
9230 CLS
9240 PLOT XG,YG,G1$
9250 PLOT XG,YG+1,G2$
9260 PLOT 4,24,"BALLON    SCORE"
9270 FOR J=0 TO 1
9280 : FOR I=0 TO 1
9290 :  PLOT XG+I,P1-J,PP+16
9300 :  PLOT XG+I,P2+J,PP+16
9310 : NEXT I
9315 : PLOT XG+2,P1-J,PA+16
9317 : PLOT XG+2,P2+J,PA+16
9320 NEXT J
9330 RETURN
9500 AM=46080+8*C
9510 FOR I=0 TO 7
9520 :  READ XX: POKE AM+I,XX
9530 NEXT I
9540 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour changer la couleur du fond (du terrain)

Remplacez en 9010, le chiffre 3 (dans PA = 3), par une valeur de votre choix (0 à 7). Par exemple :

```
9010 PA=2: IN=4
```

vous donnera un fond vert.

2. Pour changer la couleur du joueur et du ballon

Remplacez en 9010, le chiffre 4 (dans IN = 4), par une valeur de votre choix. Par exemple:

```
9010 PA=3: IN=1
```

donnera la couleur rouge au joueur et au ballon.

REMARQUE: Évitez de donner la même valeur pour PA et pour IN car, dans ce cas, le joueur et le ballon deviendraient de la même couleur que le fond. Ils seraient alors "invisibles".

3. Pour modifier le nombre de coups tirés lors d'une partie

Remplacez, en 9050, la valeur 12 par un nombre de votre choix.

4. Pour modifier la taille des buts

Modifiez, en 9080, les valeurs de P1 (ici 2) et de P2 (ici 21) qui fixent la position des poteaux.

Ainsi:

```
9080 P1=8: P2=18
```

vous fourniront de petits buts,

tandis qu'avec:

```
9080 P1=1: P2=25
```

ils occuperont toute la hauteur de l'écran.

5. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1" comme expliqué en annexe. Branchez la poignée du côté droit et remplacez les lignes 140,150,170,175 par:

```
140 : DL=PEEK(#400)
150 : IF DL=191 THEN WAIT 6: GOTO 190
170 : YG=YG+((DL AND 16)=0)*(YG>P1+1)*(YG<>Y OR XG-1<>X)
175 : YG=YG-((DL AND 8)=0)*(YG<P2-2)*(YG<>Y OR XG-1<>X)
```

Notez que l'instruction 160 n'est pas modifiée.

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.
— Poignées de jeu.

b) Le programme principal (100-260)

110 initialise le score et appelle le sous-programme d'initialisation du jeu.

120-240 répètent douze fois le tir d'un ballon. Pour chaque coup, les instructions 130 à 220 sont répétées jusqu'à ce que le ballon atteigne la ligne des buts :

130 appelle le sous-programme d'initialisation d'un coup (en 5000).

140-180 déplacent le goal si l'utilisateur l'a demandé en appuyant sur l'une des touches " ↑ " ou " ↓ ".

190-220 déplacent le ballon. Si celui-ci atteint le niveau des buts, il y a appel du sous-programme de fin d'un coup (en 4000).

c) Le sous-programme d'initialisation du jeu (9000 à 9330)

9010-9080 fixent les couleurs du fond, du goal, du joueur, des buts et du ballon ainsi que les limites du terrain et de la zone à l'intérieur de laquelle le joueur peut apparaître.

9090-9200 fabriquent les caractères graphiques utilisés pour représenter le joueur, le goal, le ballon et les buts. Ces instructions emploient le sous-programme placé en 9500.

9210 supprime le "bip" du clavier et le curseur de l'écran.

9240-9250 dessinent le goal au milieu des buts.

9270-9320 dessinent les deux poteaux des buts.

d) Le sous-programme d'initialisation d'un coup (5000 à 5140)

5010 affiche le numéro du coup qui va être joué.

5020-5030 choisissent au hasard un emplacement pour le joueur.

5040-5060 choisissent au hasard une direction de tir telle que la trajectoire du ballon passe entre les deux poteaux.

5065-5100 dessinent le ballon et le joueur "prêt à shooter" et font clignoter le mot "ATTENTION".

5110-5130 dessinent le joueur "shootant" dans le ballon et émettent un son approprié.

e) Le sous-programme de fin d'un coup (4000-4080)

4008 fait entendre un "zap" si le goal a raté le ballon.

4020-4040 augmentent de un le score et l'affichent dans le cas où le goal a intercepté le ballon.

4050-4070 effacent le joueur.

5. Liste des variables

PA	couleur de fond (papier)
IN	couleur d'encre employée pour le joueur et le ballon
PP	couleur des poteaux des buts
IG	couleur d'encre employée pour le goal
XG,YG	coordonnées du goal
NC	nombre de coups prévus
X1,X2,Y1,Y2	coordonnées du domaine dans lequel peut se trouver le joueur
P1	ordonnée du premier poteau
P2	ordonnée du second poteau
CB	code ASCII du caractère représentant le ballon
G1,G2	codes ASCII des deux caractères représentant le goal

J1	code ASCII du caractère représentant la moitié supérieure du joueur
J2	code ASCII du caractère représentant la moitié inférieure du joueur en position de repos
J3	code ASCII du caractère représentant la moitié inférieure du joueur lorsqu'il shoote dans le ballon
G1$,G2$	chaînes de caractères représentant le goal (compte-tenu de sa couleur différente de la couleur d'encre du reste de l'écran)
X,Y	coordonnées du ballon
DY	indique la direction de déplacement du ballon. Sa valeur représente le déplacement vertical correspondant à un déplacement horizontal de deux unités
YI	ordonnée de l'intersection de la trajectoire du ballon avec la ligne des buts
XJ,YJ	coordonnées du joueur
SC	score (nombre de ballons interceptés)
IC	numéro du coup courant

Variables auxiliaires

I J AM XX DL

7

Reconstitution

1. Le jeu

Un crime vient d'être commis. La victime a été coupée en morceaux ! Et le pire, c'est que vous êtes chargé de la reconstitution. De la reconstitution des circonstances du crime, pensez-vous ! Vous n'y êtes pas... il s'agit de la reconstitution de la victime...

Dure affaire, pensez-vous. En fait nous jouons sur les mots puisque la victime n'est autre qu'un mot qui va se présenter à vous avant qu'une explosion n'en disperse les lettres sur l'écran. Votre mission consiste à les ramasser le plus vite possible et dans le bon ordre.

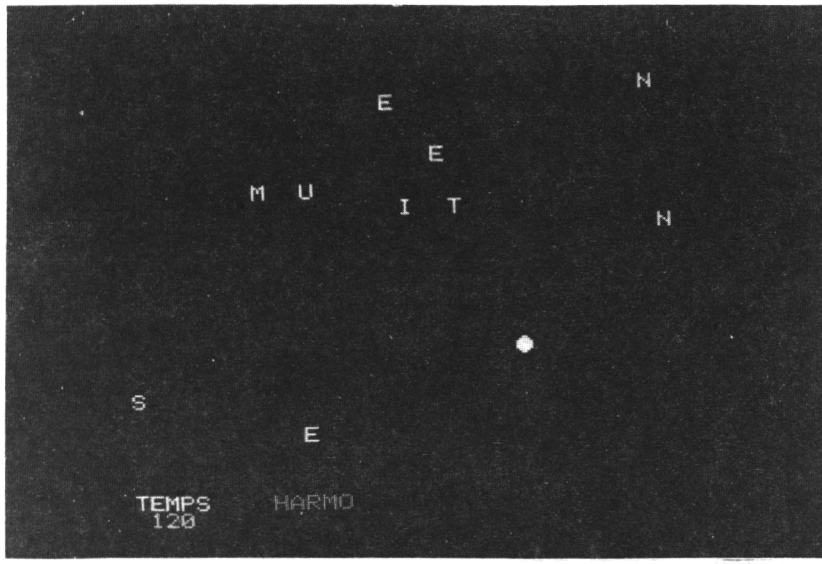
Vous ramassez les lettres grâce à une "boule" que vous promenez sur l'écran à l'aide des quatre touches fléchées. Au début de la partie, le programme vous laisse examiner à loisir l'état de la situation : position des lettres, de la boule, meilleur chemin à suivre... Quand vous pensez être prêt, vous appuyez sur une touche du clavier. A partir de là, dès que vous aurez

mis votre boule en mouvement à l'aide de l'une des quatre flèches, plus rien ne pourra l'arrêter.

Notez bien que pour votre boule, l'écran est "sphérique". Autrement dit, si la boule disparaît à gauche, elle réapparaît aussitôt à droite au même niveau. De même, si elle disparaît en haut, elle réapparaît en bas de l'écran, au-dessus des trois lignes servant à l'affichage d'informations.

Au fur et à mesure que vous ramassez les lettres, vous voyez s'inscrire, en bas de l'écran, en rouge, le mot partiellement reconstitué. Le programme affiche en permanence le temps écoulé.

Si vous gagnez, vous voyez apparaître, en clignotant, le mot "GAGNE" et votre temps moyen (par lettre) ainsi que le meilleur score. Si vous perdez, vous ne verrez que le mot "PERDU".



Vue de l'écran pendant le déroulement du jeu

2. Le programme

```
100 REM ********** RECONSTITUTION *************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
135 : FOR NL=1 TO L
140 REM
150 : IF PEEK(768)<>176 THEN DN=PEEK(735)-135
160 : IF DN>0 AND DN<5 THEN DL=DN
170 : XN=X+(DL=1)-(DL=2): YN=Y+(DL=4)-(DL=3)
180 : IF XN>X2 THEN XN=X1
190 : IF XN<X1 THEN XN=X2
200 : IF YN>Y2 THEN YN=Y1
210 : IF YN<Y1 THEN YN=Y2
220 : T=T+1: PLOT 2,26,STR$(T): PLOT 2,26," "
225 : IF DL=0 THEN 140
230 : SX=SCRN(XN,YN)
240 : PLOT X,Y,32: X=XN: Y=YN: PLOT X,Y,CB
242 : IF SX=32 THEN 140
250 : IF CHR$(SX)<>MID$(MO$,NL,1) THEN 360
270 : PLOT 10,25,CHR$(I2)+LEFT$(MO$,NL)
280 : NEXT NL
300 : PING: PLOT 2,24,12: PLOT 3,24,"GAGNE"
310 : TM=INT(T/L): PLOT 11,24,"TEMPS MOYEN"
320 : PLOT 27,24,STR$(TM): PLOT 27,24," "
330 : PLOT 11,25,"MEILLEUR TEMPS   "
340 : PLOT 27,25,STR$(MT): PLOT 27,25," "
350 : IF TM<MT OR MT=0 THEN MT=TM: GOTO 370
360 : ZAP: PLOT 2,24,12: PLOT 3,24,"PERDU"
370 : PLOT 15,26,12: PLOT 15,26,"rejouez vous (O/N)"
380 : GET R$: GET R$
400 UNTIL R$<>"O"
410 END
8000 REM ---- SP INITIALISATION D'UNE PARTIE ----------
8010 T=0
8060 K=1+INT(RND(1)*NM)
8068 MO$=MO$(K)
8080 CLS
8090 PLOT 3,10,12: PLOT 4,10,MO$
8100 WAIT 200: EXPLODE
8110 CLS
8120 L=LEN(MO$)
8130 FOR I=1 TO L+1
8140 : X=X1+INT(RND(1)*(X2-X1+1))
```

```
8150 : Y=Y1+INT(RND(1)*(Y2-Y1+1))
8160 : IF SCRN(X,Y)<>32 THEN 8140
8170 : IF I<L+1 THEN PLOT X,Y,MID$(MO$,I,1)
8180 : IF I=L+1 THEN PLOT X,Y,CB
8190 NEXT I
8200 DL=0
8210 PLOT 2,25,I2: PLOT 3,25,"pour commencer, tapez une touche"
8220 GET R$
8230 PLOT 1,25,BL$
8240 PLOT 2,25,"TEMPS"
8250 RETURN
9000 REM ---- SP INITIALISATION DU JEU ----------------
9010 PA=0: IN=3: I2=1: PAPER PA: INK IN
9020 MT=0
9025 X1=2: X2=38: Y1=0: Y2=23
9030 CB=64: AM=46080+8*CB
9035 DATA 12,30,63,63,63,63,30,12
9040 FOR I=0 TO 7
9050 : READ XX: POKE AM+I,XX
9060 NEXT I
9070 BL$=""
9080 FOR I=1 TO 36: BL$=BL$+" ": NEXT I
9090 DATA 9
9100 DATA HARMONIEUSEMENT,MEGALOMANIE,SCIENTIFIQUE
9110 DATA RICHISSIME,CACOPHONIE,DRAMATIQUE
9120 DATA PLANIFICATION,DEMOCRATISATION,RADICALISATION
9130 READ NM
9140 DIM MO$(NM)
9150 FOR I=1 TO NM
9160 : READ MO$(I)
9170 NEXT I
9180 POKE #26A,10
9190 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier la couleur du fond

Remplacez, en 9010, la valeur 0 par un nombre de votre choix (entre 1 et 7).

2. Pour modifier la couleur de la boule et des lettres

Remplacez, en 9010, la valeur 3 par un nombre de votre choix (entre 0 et 7). Évitez d'utiliser la même couleur que pour le fond car, alors, lettres et boule deviendraient "invisibles".

3. Pour jouer sur un domaine plus petit

Agissez sur la ligne 9025 en augmentant les valeurs de X1 et Y1 et en diminuant celles de X2 et Y2. Essayez, par exemple:

```
9025 X1=10: X2=30: Y1=5: Y2=20
```

4. Pour ajouter de nouveaux mots ou pour modifier les mots proposés

Il vous faut savoir que les instructions "DATA" des lignes 9090 à 9120 fournissent:

— le nombre total de mots (ici 9)

— la liste de ces mots. Chaque instruction peut contenir un nombre quelconque de mots; ceux-ci doivent simplement être séparés par des virgules.

Notez bien que vous pouvez placer, non seulement des mots ayant un sens, mais éventuellement n'importe quelles suites de caractères (lettres, chiffres,...). Cela peut rendre le jeu quelque peu plus compliqué.

5. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1" comme décrit en annexe. Branchez la poignée du côté droit et remplacez les instructions 150,160,170 par:

```
150 : C=PEEK(#400): IF C<>191 THEN DL=C
160 : XN=X+((DL AND 1)=0)-((DL AND 2)=0)
170 : YN=Y+((DL AND 16)=0)-((DL AND 8)=0)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.
— Poignées de jeu (facultatives).

b) Le programme principal (100-410)

110 appelle le sous-programme d'initialisation du jeu.

120 et 400 répètent les instructions 130 à 380 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer :

130 appelle le sous-programme d'initialisation d'une partie.

135 et 280 répètent, pour chaque lettre du mot choisi, les instructions 140 à 270 qui correspondent au ramassage d'une lettre :

140-160 déterminent, dans DL, le nouveau déplacement de la boule (codé de 1 à 4), en fonction de l'ancien déplacement ou de la touche enfoncée.

170-210 déterminent les nouvelles coordonnées de la boule.

220 actualise le temps.

225 est utile pour le début de partie (la boule est alors immobile).

230-250 déplacent la boule, examinent le contenu de l'emplacement où elle arrive.

270 affiche la lettre rencontrée, s'il s'agit de la bonne.

300-350 cas où le mot a été entièrement reconstitué. On affiche le mot "GAGNE", le temps moyen et le meilleur temps que l'on actualise ensuite.

360 cas où une erreur de lettre a été commise pendant la reconstitution. On affiche le mot "PERDU".

370-380 vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9190)

9010 fixe les couleurs de fond et d'encre.

9020 initialise le meilleur temps.

9025 fixe les limites du domaine de jeu.

9030-9060 fabriquent le caractère graphique représentant la boule.

9070-9080 préparent une chaîne de 36 caractères "blanc" utilisée pour l'effacement.

9090-9170 remplissent, grâce aux DATA des lignes 9100 à 9120, le tableau MO$ des mots parmi lesquels se fera le tirage au hasard.

9180 supprime le bip du clavier et le curseur d'écran.

d) Le sous-programme d'initialisation d'une partie (8000-8250)

8010 initialise le compteur de temps.

8060-8090 choisissent un mot au hasard (dans MO$), l'affichent à l'écran pendant deux secondes.

8100-8110 font retentir une explosion et effacent l'écran.

8120-8190 affichent chaque lettre du mot et la boule en un emplacement quelconque de l'écran.

8200 initialise le déplacement de la boule.

8210-8240 attendent que vous tapiez sur une touche.

5. Liste des variables

PA	couleur de fond
IN	couleur d'encre employée pour les lettres et la boule
I2	couleur d'encre employée pour le message "pour commencer..."

MT	meilleur temps
X1,X2,Y1,Y2	coordonnées du domaine de jeu
CB	code ASCII du caractère représentant la boule
BL$	chaîne de 36 caractères "blanc" servant à l'effacement de message
NM	nombre total de mots parmi lesquels on en choisira un au hasard
MO$(NM)	tableau contenant les mots parmi lesquels on en tirera un au hasard
T	compteur de temps écoulé
MO$	mot tiré au hasard pour la partie en cours
L	longueur de ce mot
DL	déplacement courant de la boule 0 : boule immobile (seulement en début de partie) 1 : gauche 2 : droite 3 : bas 4 : haut
NL	numéro de la prochaine lettre à ramasser
X,Y	coordonnées courantes de la boule (sauf dans le sous-programme 8000)
XN,YN	prochain emplacement de la boule
SX	code du caractère situé sur le prochain emplacement de la boule
TM	temps moyen (par lettre)

Variables auxiliaires

AM I XX K R$

X et Y (dans le sous-programme 8000)

8

Phosphore

1. Le jeu

Le but de ce jeu de mémoire consiste à retenir des suites de lettres les plus longues possibles. Dix essais vous sont proposés. Pour chacun d'entre eux, le programme commence par vous montrer pendant un court instant une première lettre choisie au hasard. Il vous demande ensuite quelle était la lettre affichée. Si votre réponse est correcte, le programme vous gratifie d'un petit compliment et vous montre à nouveau cette lettre suivie d'une seconde. Cela se poursuit ainsi avec trois lettres, puis quatre... jusqu'à ce que vous vous trompiez. Vous marquez alors un nombre de points égal au nombre de lettres que vous avez réussi à retenir.

Le programme affiche en permanence le numéro de l'essai que vous êtes entrain de jouer, le nombre de lettres que vous avez déjà mémorisées pour l'essai en cours, votre score et le meilleur score.

Chaque lettre proposée est accompagnée d'un son (qui reste le même pour une lettre donnée) qui peut faciliter la mémorisation.

REMARQUE: Bien attendre que le programme vous pose la question: "quelles étaient les lettres proposées" avant de commencer à taper votre réponse.

2. *Le programme*

```
100 REM *** PHOSPHORE ***
110 GOSUB 9000
120 FOR CC=1 TO NC
130 : XA=CC: XD=XF(1): GOSUB 6000
140 : LE$=""
150 :   GOSUB 8000
160 :   GOSUB 7000
170 :   IF OK=1 THEN 150
175 : IF CC=NC THEN 200
180 : PLOT 0,26,20: PLOT 1,26,3
185 : PLOT 2,26,"tapez sur une touche pour continuer"
190 : R$=KEY$: IF R$="" THEN 190
200 NEXT CC
210 IF SC>MS THEN MS=SC
220 XA=MS: XD=XF(4): GOSUB 6000
230 CLS: PLOT 5,12,"voulez vous rejouer (O/N)"
240 GET R$: IF R$="O" THEN 120
250 END
6000 REM ... SP AFFICHAGE VALEUR NUMERIQUE ........
6010 CH$=STR$(XA)+" " : GOSUB 6500
6020 CH$=" "      : GOSUB 6500
6030 RETURN
6500 REM ... SP DEPOS CHAINE CARAC EN MEM GRAPH ..........
6510 LE=LEN(CH$)
6520 FOR KK=1 TO LE
6530 : POKE XD+KK-1,ASC(MID$(CH$,KK,1))
6540 NEXT KK
6550 RETURN
7000 REM ...SP LECTURE DE LA REPONSE ..............
7010 CLS: PAPER 6: INK 4
7020 NL=LEN(LE$)
```

```
7030 FOR I=1 TO 4: PRINT: NEXT I
7040 IF NL=1 THEN PRINT "quelle etait la lettre?"
7050 IF NL>1 THEN PRINT "quelles etaient les ";NL;" lettres?"
7060 PRINT
7070 FOR I=1 TO NL
7080 : R$=KEY$: IF R$="" THEN 7080
7090 : PRINT R$;
7100 : IF R$<>MID$(LE$,I,1) THEN 7200
7110 NEXT I
7115 PRINT: PRINT: PRINT: WAIT 50
7120 PRINT R$(INT(RND(1)*NR+1))
7130 OK=1
7140 WAIT 100
7150 RETURN
7200 PRINT : PRINT: PRINT N$(RND(1)*NN+1)
7210 ZAP: WAIT 50 : PRINT
7215 PRINT "la bonne reponse etait :"
7220 GOSUB 8500
7230 GOSUB 7500
7240 OK=0
7245 WAIT 150
7250 RETURN
7500 REM ... SP MISE A JOUR ET AFFICHAGE DU SCORE .......
7510 P=LEN(LE$)-1
7520 PLOT 1,15,"vous marquez": PLOT 13,15,1: PLOT 14,15,STR$(P)
7525 PLOT 14,15," ": PLOT 18,15,4: PLOT 19,15,"points"
7530 WAIT 30
7540 SC=SC+P
7550 XA=SC: XD=XF(3): GOSUB 6000
7560 WAIT 50
7570 RETURN
8000 REM ... SP AFFICHAGE DES LETTRES .........
8010 CLS: PAPER 6: INK 4
8020 PLOT 10,10,"REGARDEZ"
8030 WAIT 150:CLS
8040 IN=EN(INT(RND(1)*NE+1))
8050 PA=PA(INT(RND(1)*NP+1))
8060 PAPER PA: INK IN
8070 C=65+INT(RND(1)*26)
8080 LE$=LE$+CHR$(C)
8090 XA=LEN(LE$): XD=XF(2): GOSUB 6000
8100 GOSUB 8500
8110 RETURN
```

```
8500 REM ... SP AFFICHAGE SONORE DES LETTRES ...........
8510 PLAY 1,0,0,0
8520 FOR I=1 TO LEN(LE$)
8530 : C$=MID$(LE$,I,1)
8540 : S=6*(ASC(C$)-40)
8550 : SOUND 1,S,6
8560 : PLOT I,12,C$
8570 WAIT 50
8580 NEXT I
8590 PLAY 0,0,0,0 : WAIT 100
8600 RETURN
9000 REM ... SP INITIALISATIONS ..............
9008 NC=10: AD=48000
9010 PRINT CHR$(17);
9020 FOR I=1 TO 39: POKE AD+I,32: NEXT I
9040 DIM C$(4), XD(4), XF(4), IN(4)
9050 XD(1)=AD+1: C$(1)="COUP"    : IN(1)=2: XF(1)=XD(1)+5
9060 XD(2)=AD+10:C$(2)="LETTRES": IN(2)=6: XF(2)=XD(2)+8
9070 XD(3)=AD+22:C$(3)="SC    :"  : IN(3)=3: XF(3)=XD(3)+3
9075 XD(4)=AD+30: C$(4)="M-SC": IN(4)=3: XF(4)=XD(4)+4
9080 PS=17: POKE AD,PS
9090 FOR I=1 TO 4
9100 : CH$=CHR$(IN(I))+C$(I)
9110 : XD=XD(I)
9120 : GOSUB 6500
9130 NEXT I
9160 DATA 3,2,3,6
9170 READ NP: DIM PA(NP)
9180 FOR I=1 TO NP: READ PA(I): NEXT I
9190 DATA 3,1,4,5
9200 READ NE: DIM EN(NE)
9210 FOR I=1 TO NE: READ EN(I): NEXT I
9220 DATA 9
9230 DATA OK,OUI,EXACT,BIEN,JUSTE,CORRECT,BRAVO,PARFAIT,VRAI
9240 READ NR: DIM R$(NR)
9250 FOR I=1 TO NR: READ R$(I): NEXT I
9260 DATA 5
9270 DATA NON,FAUX,INEXACT,INCORRECT,ERREUR
9280 READ NN: DIM N$(NN)
9290 FOR I=1 TO NN: READ N$(I): NEXT I
9300 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier le temps qui s'écoule entre l'affichage de deux lettres successives

Remplacez, en 8570, le nombre 50 par la valeur de votre choix (exprimée en dizaines de ms).

2. Pour modifier le temps d'attente à la fin de la présentation d'une suite de lettres

Remplacez, en 8590, le nombre 100 par la valeur de votre choix.

3. Pour supprimer le message "REGARDEZ" qui apparaît pendant 1,5 s,

Supprimez les instructions 8020,8030.

4. Pour modifier le nombre d'essais proposés

Remplacez, en 9008, la valeur 10 par celle de votre choix.

5. Pour modifier les couleurs employées pour la présentation des lettres, sachez que :

— l'instruction DATA de la ligne 9160 contient le nombre de couleurs de fond (ici 3), suivi des codes de ces couleurs (ici 2, 3 et 6).

— l'instruction DATA de la ligne 9190 contient le nombre de couleurs d'encre (ici 3) suivi des codes de ces couleurs (ici 1, 4 et 5).

REMARQUE: Évitez de placer deux codes de couleurs identiques dans chacune de ces instructions car, de temps en temps, les lettres proposées seraient invisibles.

6. Pour obtenir des suites de chiffres à la place des suites de lettres

Remplacez la ligne 8070 par :

```
8070 C=48+INT(RND(1) * 10)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Affichage ligne supérieure.
— Configuration clavier/écran.
— Écriture directe en mémoire d'écran.

b) Le programme principal (100-250)

110 appelle le sous-programme d'initialisation du jeu (en 9000).

120-200 proposent les dix essais. Pour chaque essai:

130 affiche le numéro du coup.

140 initialise la suite de lettres proposée.

150-170 ajoutent une lettre à la suite proposée (sous-programme 8000) et lisent votre réponse (sous-programme 7000) jusqu'à ce que celle-ci soit fausse.

175-190 vous permettent de "souffler" entre deux essais en vous demandant de frapper une touche lorsque vous êtes prêt à poursuivre. L'instruction 175 évite que cette question ne soit posée à la fin du dernier essai.

210-220 actualisent et affichent le meilleur score.

230-240 vous demandent si vous désirez faire une nouvelle partie.

c) Le sous-programme d'initialisation du jeu (9000-9300)

9008 fixe le nombre d'essais d'une partie et l'adresse de début de la ligne supérieure.

9010 supprime le curseur d'écran.

9020 efface la ligne supérieure de l'écran (celle où apparaît le mot "CAPS".

9040-9075 déterminent les emplacements et les différents textes que l'on souhaite voir figurer en ligne supérieure, ainsi que leur couleur.

9080 fixe la couleur de fond de la ligne supérieure.

9090-9130 affichent les textes voulus en ligne supérieure.

9160-9210 déterminent les différentes couleurs de fond et d'encre qui seront utilisées lors de la présentation des suites de lettres.

9220-9290 fixent les différents messages qui seront formulés par le programme, après votre proposition.

d) Le sous-programme de préparation d'une suite de lettres (8000-8110)

8010-8030 affichent le message "REGARDEZ" pendant 1,5 s.

8040-8060 choisissent une couleur de fond et une couleur d'encre.

8070-8080 choisissent une lettre au hasard et l'ajoutent à la suite déjà proposée.

8090 affiche, en ligne supérieure, le nombre de lettres proposées.

8100 appelle le sous-programme d'affichage des lettres.

e) Le sous-programme d'affichage d'une suite de lettres (8500-8600)

8510-8580 affichent chacune des lettres de la suite, en émettant un son (dont la fréquence dépend de la lettre), et en attendant une demi-seconde entre chaque lettre.

8590-8600 suppriment le son et attendent une seconde.

f) Le sous-programme de lecture de la réponse du joueur (7000-7250)

7000-7060 vous demandent quelles étaient les lettres proposées.

7070-7110 lisent votre réponse, lettre par lettre, en la comparant avec la suite proposée et en s'arrêtant dès qu'une erreur est rencontrée.

7115-7150 exécutées en cas de réponse correcte: impriment un petit message de compliments.

7200-7250 exécutées en cas de réponse incorrecte: impriment un petit message d'erreur, émettent un "zap", affichent (en musique) la réponse exacte et comptabilisent vos points (sous-programme 7500).

g) Sous-programmes d'affichage en ligne supérieure (6000-6500)

Le sous-programme 6500 affiche le contenu de la chaîne CH$, à partir de la colonne de rang XD de la ligne supérieure.

Le sous-programme 6000 affiche la valeur contenue dans XA, à partir de la colonne de rang XD de la ligne supérieure.

5. Liste des variables

NC	nombre d'essais proposés
AD	adresse mémoire correspondant au début de la ligne supérieure de l'écran
PS	couleur de fond employée pour la ligne supérieure
C$(4)	tableau contenant les textes inscrits en ligne supérieure
XD(4)	tableau contenant les emplacements de ces textes (dans la ligne)
XF(4)	tableau contenant les emplacements des valeurs correspondant à ces textes
IN(4)	couleur d'encre utilisée pour chacun de ces textes
NP	nombre de couleurs différentes employées pour le fond lors de la présentation des lettres
PA(NP)	tableau contenant les valeurs correspondant à ces différentes couleurs
NE	nombre de couleurs différentes employées pour l'écriture des lettres, lors de leur présentation
EN(NE)	tableau contenant les valeurs correspondant à ces différentes couleurs
NR	nombre de messages différents utilisés pour encourager le joueur lorsqu'il a trouvé la bonne suite de lettres
R$(NR)	tableau contenant ces différents messages
NN	nombre de messages différents utilisés en cas de mauvaise réponse
N$(NN)	tableau contenant ces différents messages

CC	numéro de l'essai courant
LE$	chaîne contenant la suite de lettres proposées
IN	couleur d'encre (courante) utilisée pour l'affichage des lettres
PA	couleur de papier utilisée lors de l'affichage des lettres
C	code ASCII de la dernière lettre ajoutée à la suite proposée
NL	nombre de lettres proposées
P	nombre de points marqués à la fin d'un essai
SC	score (nombre total de points)
OK	prend la valeur 1 quand la réponse du joueur est correcte, 0 dans le cas contraire
MS	meilleur score

Variables auxiliaires

XD XA I R$ CH$ LE KK

9

Les allumettes suédoises

1. Le jeu

Nous vous proposons ici une version simplifiée du célèbre jeu, connu également sous le nom de " Marienbad ". Ce programme vous donnera une bonne idée des dessins que l'on peut réaliser avec l'ORIC, sans même utiliser le mode haute résolution graphique (HIRES).

Le jeu proposé se joue à deux. Ici, vous jouez contre l'ordinateur (ou plutôt contre le programme). Au départ, vous êtes en présence de 36 allumettes. Chaque joueur, à tour de rôle, choisit d'enlever un nombre d'allumettes compris entre 1 et 7. Celui qui est obligé d'enlever la dernière allumette a perdu.

Le programme commence par dessiner les 36 allumettes puis choisit au hasard celui qui jouera en premier. Quand c'est au programme de jouer, il vous dit: "j'enlève x allumettes". Vous voyez ensuite les allumettes concernées clignoter puis disparaître en musique (chaque allumette égrène

une note de la gamme). Quand c'est à vous de jouer, vous voyez apparaître la question "combien d'allumettes retirez-vous ? Vous répondez en tapant sur le chiffre souhaité. Notez que toute touche ne correspondant pas à un nombre compris entre 1 et 7 est purement et simplement ignorée.

Le programme affiche en permanence le nombre d'allumettes restant (RESTE), le numéro de la partie que vous êtes entrain de jouer ainsi que le nombre de parties gagnées par vous et par lui. (Voir photo 2 hors texte : *Vue de l'écran pendant le déroulement du jeu*).

2. Le programme

```
100 REM ********** LES ALLUMETTES SUEDOISES **********
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : REPEAT
150 :  IF JO=1 THEN GOSUB 5000 ELSE GOSUB 4000
160 :  GOSUB 3000
170 :  N=N-P
180 :  PLOT 3,2,STR$(N)+" ": PLOT 3,2," "
190 :  JO=-JO
200 : UNTIL N<=1
210 : IF N=1 AND JO=1 THEN GOSUB 2000 ELSE GOSUB 2500
220 : GOSUB 8500
230 : JD=-JD
240 UNTIL R$<>"O"
250 END
2000 REM ... SP JOUEUR GAGNANT
2010 PJ=PJ+1
2020 PLOT 18,2,STR$(PJ): PLOT 18,2," "
2030 RETURN
2500 REM ... SP ORDINATEUR GAGNANT ..............
2510 PO=PO+1
2520 PLOT 31,2,STR$(PO): PLOT 31,2," "
2530 RETURN
```

```
3000 REM ... SP EFFACEMENT P ALLUMETTES .............
3010 RD=1: KD=N-P+1
3020 IF N-P>=18 THEN RD=2: KD=KD-18
3030 RF=1: KF=N
3040 IF N  >=18 THEN RF=2: KF=KF-18
3050 YD=10*(RD-1)+5
3060 X=2*KD-1
3070 FOR Y=YD TO YD+7
3080 : PLOT X,Y,12
3090 NEXT Y
3100 IF RD=RF THEN 3140
3110 FOR Y=15 TO 22
3120 : PLOT 1,Y,12
3130 NEXT Y
3140 WAIT 200
3150 BS=1: KN=0: OC=3
3160 IF RD=RF THEN 3300
3170 FOR X=2*KD-1 TO 36
3180 : FOR Y=5 TO 12
3190 :  PLOT X,Y,32
3200 : NEXT Y
3210 : GOSUB 3500
3220 NEXT X
3230 FOR X=1 TO 2*KF
3240 : FOR Y=15 TO 22
3250 :  PLOT X,Y,32
3260 : NEXT Y
3270 : GOSUB 3500
3280 NEXT X
3290 GOTO 3400
3300 FOR X=2*KD-1 TO 2*KF
3310 : FOR Y=YD TO YD+7
3320 :  PLOT X,Y,32
3330 : NEXT Y
3340 : GOSUB 3500
3350 NEXT X
3400 RETURN
3500 REM ... SP EMISSION NOTE LORS EFFACEMENT .......
3510 BS=-BS
3520 IF BS=1 THEN KN=KN+1 ELSE 3590
3530 MUSIC 1,OC,NO(KN),0
3560 PLAY 1,0,1,500
3570 WAIT 20
3580 IF KN=7 THEN KN=0: OC=OC+1
3590 RETURN
```

```
4000 REM ... SP LECTURE CHOIX DU JOUEUR .............
4010 PLOT 1,25,"Combien d'allumettes retirez vous?"
4020 R$=KEY$: IF R$="" THEN 4020
4030 P=ASC(R$)-48
4040 IF P>0 AND P<=PM THEN 4080
4050 PLOT 1,26,"entre 1 et"
4060 PLOT 13,26,STR$(PM): PLOT 13,26," "
4070 GOTO 4020
4080 PLOT 37,25,R$
4090 WAIT 50
4100 PLOT 1,26,BL$
4110 RETURN
5000 REM ... SP CHOIX ORDINATEUR NB ALLUMETTES ......
5010 Q=INT(N/(PM+1))
5020 R=N-Q*(PM+1)
5030 P=R-1: IF P<=0 THEN P=P+PM+1
5032 IF P>PM THEN P=INT(RND(1)*PM+1)
5034 IF N<=PM+1 THEN P=N-1
5035 PLOT 1,25,BL$
5040 PLOT 2,25,"J'enleve"
5050 PLOT 10,25,STR$(P): PLOT 10,25," "
5060 IF P=1 THEN PLOT 13,25,"allumette"
5070 IF P>1 THEN PLOT 13,25,"allumettes"
5080 WAIT 200
5090 RETURN
7000 REM ... SP DESSIN DES ALLUMETTES ..............
7010 ND=18: IF NX<18 THEN ND=NX
7020 FOR X=2 TO 2*ND STEP 2
7030 : PLOT X,5,C1
7040 : PLOT X,6,C2
7050 : FOR Y=7 TO 12
7060 :  PLOT X,Y,CT
7070 : NEXT Y
7080 NEXT X
7090 ND=NX-ND
7100 IF ND<=0 THEN 7200
7110 FOR X=2 TO 2*ND STEP 2
7120 : PLOT X,15,C1
7130 : PLOT X,16,C2
7140 : FOR Y=17 TO 22
7150 :  PLOT X,Y,CT
7160 : NEXT Y
7170 NEXT X
7200 RETURN
```

```
8000 REM ... SP INITIALISATION D'UNE PARTIE .........
8010 N=NX: NP=NP+1
8020 PLOT 9,2,STR$(NP): PLOT 9,2," "
8030 PLOT 3,2,STR$(N)+" ": PLOT 3,2," "
8035 GOSUB 7000
8040 IF JD=1  THEN PLOT 2,24,"je joue le premier"
8050 IF JD=-1 THEN PLOT 2,24,"vous jouez le premier"
8060 JO=JD
8070 WAIT 100
8080 PLOT 2,26,"pour commencer, tapez sur une touche"
8090 R$=KEY$: IF R$="" THEN 8090
8100 PLOT 1,24,BL$
8110 PLOT 1,26,BL$
8130 RETURN
8500 REM ... SP FIN DE PARTIE .......................
8505 WAIT 200
8510 FOR Y=5 TO 15
8520 : PLOT 2,Y,32
8530 NEXT Y
8540 PLOT 2,25,"Voulez vous rejouer (O/N)?"
8550 R$=KEY$: IF R$="" THEN 8550
8560 PLOT 1,25,BL$
8570 RETURN
9000 REM ... SP INITIALISATIONS ..................
9010 P1=3: I1=1: P2=6: IS=1: IT=3: P3=5: I3=7
9020 NX=36: PM=7: NP=0: PO=0: PJ=0
9030 CLS: PAPER P2: INK IT
9100 AM=48040
9110 FOR I=0 TO 3
9120 : POKE AM  +40*I,P1+16
9130 : POKE AM+1+40*I,I1
9140 NEXT I
9150 FOR I=24 TO 26
9160 : POKE AM  +40*I,P3+16
9170 : POKE AM+1+40*I,I3
9180 NEXT I
9190 FOR I=5 TO 6
9200 : POKE AM+1+40*I,IS
9210 NEXT I
9220 FOR I=15 TO 16
9230 : POKE AM+1+40*I,IS
9240 NEXT I
9250 PLOT 1,1,"RESTE  PARTIE  VOTRE SCORE  MON SCORE"
9255 CT=64: C1=95: C2=96
9260 DATA 30,30,30,30,30,30,30,30
9270 C=CT: GOSUB 9500
```

```
9280 DATA 0,0,0,0,12,30,63,63
9290 C=C1: GOSUB 9500
9300 DATA 63,63,63,63,63,30,30,30
9310 C=C2: GOSUB 9500
9320 IF NX>36 THEN NX=36
9330 JD=1: IF RND(1)>0.5 THEN JD=-1
9340 BL$="": FOR I=1 TO 38: BL$=BL$+" ": NEXT I
9350 DATA 1,3,5,6,8,10,12
9360 DIM NO(7)
9370 FOR I=1 TO 7
9380 : READ NO(I)
9390 NEXT I
9395 POKE #26A,10
9400 RETURN
9500 REM ... SP INIT GRAPHIQUES ..................
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour changer la couleur de la zone principale où sont dessinées les allumettes

Remplacez, en 9010, la valeur 6 (dans P2 = 6) par un nombre de votre choix, compris entre 0 et 7 (mais différent de 1 et de 3).

2. Pour modifier les couleurs en général

Reportez-vous à la liste des variables et voyez le rôle de : P1, I1, P2, IS, IT, P3, I3. Modifiez la ligne 9010 en fonction de vos desiderata.

3. Pour modifier le nombre total d'allumettes

Remplacez, en 9020, la valeur 36 par un nombre de votre choix ne dépassant pas 36.

4. Pour modifier le nombre maximum d'allumettes que l'on peut supprimer en une fois

Remplacez, en 9020, la valeur 7 par un nombre de votre choix, compris entre 1 et 9.

4. Description du programme

a) Techniques utilisées décrites en annexe

— Création de caractères graphiques.
— Écriture directe en mémoire d'écran.
— Configuration clavier/écran.

b) Le programme principal

110 appelle le sous-programme d'initialisation du jeu.

120 et 240 répètent les instructions 130 à 230 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne désirez plus rejouer :

130 appelle le sous-programme d'initialisation d'une partie.

140 et 200 répètent le jeu d'un coup (par vous ou par le programme) jusqu'à ce qu'il ne reste plus qu'une allumette.

150 permet le choix du nombre d'allumettes à retirer, soit par vous (sous-programme 4000), soit par l'ordinateur (sous-programme 5000). Le joueur est indiqué par la variable JO (1 : ordinateur ; — 1 : vous).

160 appelle le sous-programme qui fait "disparaître" les allumettes concernées.

170-180 actualisent le nombre d'allumettes restantes.

190 change de joueur.

210 analyse le résultat de la partie et appelle, suivant le cas, l'un des sous-programmes "joueur gagnant" ou "ordinateur gagnant".

220 appelle le sous-programme de fin de partie.

230 change le joueur qui doit commencer la partie.

c) Le sous-programme d'initialisation du jeu (9000-9400)

9010 fixe les différentes couleurs intervenant sur l'écran.

9020 fixe le nombre d'allumettes, la prise maximum autorisée et initialise les compteurs.

9030-9240 donnent les couleurs voulues aux différentes zones de l'écran. Les couleurs principales sont attribuées par PAPER et IN à l'ensemble de l'écran. Les autres couleurs sont déterminées par l'écriture directe (POKE) d'attributs en mémoire d'écran.

9250 affiche les titres de la zone supérieure.

9255-9310 fabriquent les caractères graphiques servant à représenter la tige et la tête des allumettes.

9330 choisit, au hasard, le joueur qui commencera la première partie.

9340-9390 placent dans le tableau NO les codes (employés par MUSIC) des notes de la gamme.

d) Le sous-programme d'initialisation d'une partie (8000-8130)

8010-8030 initialisent le nombre total d'allumettes et actualisent le numéro de partie.

8035 appelle le sous-programme de dessin des allumettes.

8040-8110 écrivent un message précisant qui joue en premier, initialisent le numéro du joueur courant et attendent que vous tapiez sur une touche.

e) Le sous-programme de dessin des allumettes (7000-7200)

7010-7080 dessinent les allumettes de la rangée supérieure (dix-huit au maximum).

7090-7170 dessinent, s'il y a lieu, les allumettes de la rangée inférieure (lorsque le nombre total d'allumettes est supérieur à dix-huit).

f) Le sous-programme de choix du joueur (4000-4110)

4010-4030 vous demandent combien d'allumettes vous souhaitez enlever et lisent votre réponse.

4040 vérifie que votre réponse est correcte.

4050-4070 vous précisent les valeurs autorisées lorsque votre réponse est incorrecte.

4080-4100 affichent la réponse lue puis effacent la question, après un court instant.

g) Le sous-programme de choix de l'ordinateur (5000-5090)

5010-5034 déterminent le nombre d'allumettes à enlever.

5035-5080 affichent ce nombre.

h) Le sous-programme d'effacement de "P" allumettes (3000-3400)

P contient le nombre d'allumettes à effacer.

3010-3060 déterminent les emplacements des allumettes à effacer.

3070-3130 font clignoter pendant 2 secondes les allumettes à effacer.

3150 initialise le processus d'effacement.

3160 examine si les allumettes à effacer sont ou non dans la même rangée.

3170-3350 effacent les allumettes en utilisant le sous-programme 3500 pour émettre une note différente pour chaque allumette effacée :

3170-3280 correspondent au cas où les allumettes à effacer sont situées dans deux rangées différentes.

3300-3350 correspondent au cas où elles sont situées dans une même rangée.

REMARQUE : Par souci de simplification, ce sous-programme traite de la même manière une "colonne" contenant une allumette qu'une colonne séparant deux allumettes.

i) Le sous-programme d'émission d'une note (3500-3590)

Il est appelé pour chaque "colonne" effacée. L'indicateur BS permet de n'émettre une note que dans le cas où l'on a effectivement effacé une allumette.

j) Les sous-programmes joueur gagnant (2000-2030) et ordinateur gagnant (2500-2530)

Ils actualisent le nombre de parties gagnées.

k) Le sous-programme de fin de partie (8500-8570)

8505-8530 effacent la dernière allumette, après un temps d'attente.

8540-8560 vous demandent si vous voulez rejouer.

5. Liste des variables

P1	couleur de fond de la zone supérieure
I1	couleur d'encre employée dans la zone supérieure
P2	couleur de fond de la zone principale (où sont dessinées les allumettes)
IS	couleur de la tête des allumettes
IT	couleur de la tige des allumettes
P3	couleur de fond de la zone inférieure
I3	couleur d'encre employée dans la zone inférieure
NX	nombre total d'allumettes
PM	nombre maximum d'allumettes que l'on peut retirer en une fois
NP	nombre de parties jouées
PO	nombre de parties gagnées par l'ordinateur
PJ	nombre de parties gagnées par le joueur
AM	adresse mémoire du début de l'écran
CT	code ASCII du caractère utilisé pour représenter la tige des allumettes
C1	code ASCII du caractère représentant la partie supérieure de la tête des allumettes
C2	code ASCII du caractère représentant la partie inférieure de la tête des allumettes

JD joueur commençant la partie : (1 : ordinateur ; —1 : joueur)

BL$ chaîne de 38 caractères "blanc" employée pour effacer des messages

NO(7) tableau contenant les numéros (employés par l'instruction MUSIC) des sept notes de la gamme (1 pour DO, 3 pour RE, 5 pour MI, 6 pour FA, etc...)

N nombre d'allumettes restantes

JO joueur courant (1 : ordinateur ; —1 : joueur)

ND nombre d'allumettes à dessiner dans une rangée

X,Y coordonnées courantes d'un emplacement écran

P nombre d'allumettes à enlever (choix du joueur ou de l'ordinateur)

BS indicateur employé dans le sous-programme d'émission d'une note. Quand il vaut +1, une note doit être émise.

OC octave de la note émise

KN pointeur dans le tableau NO. NO(KN) donne le code de la note à émettre.

RD rangée (1 ou 2) contenant la première allumette à effacer

KD numéro, dans la rangée, de la première allumette à effacer

RF rangée contenant la dernière allumette à effacer

KF numéro, dans la rangée, de la dernière allumette à effacer.

Variables auxiliaires

I R$ YD

C AM XX Q R

10
MASTER MIND

1. Le jeu

Il s'agit là d'un grand classique des jeux de réflexion. Si vous en connaissez déjà les règles, vous retrouverez un univers familier. Par rapport au jeu manuel vous rencontrerez essentiellement deux différences :

- Le nombre de positions peut atteindre 6 tandis que le nombre de couleurs peut atteindre 9.
- Manuellement, vous preniez un pion pour le déposer dans un trou du plateau de jeu. Ici, vous sélectionnez la couleur de votre pion grâce à un curseur que vous déplacez à l'aide des touches → et ←. Cette façon de procéder évite la sempiternelle question "entrez les couleurs choisies" qui oblige toujours à coder les couleurs (chiffres, lettres,...). Ici, on choisit naturellement sa couleur en la désignant.

Si vous n'avez jamais pratiqué ce jeu, vous allez pouvoir vous y exercer tranquillement sans risquer d'être l'objet de commentaires désobligeants

des plus entraînés. Votre compagnon ordinateur ne marquera aucune impatience. Et quand bien même vous ne parviendriez pas à trouver en 22 coups une simple combinaison de 2 couleurs sur 2 positions, il se contentera simplement de vous montrer la solution, sans commentaires.

a) Le but du jeu

Il consiste à découvrir une combinaison de couleurs, tirée en secret par le programme. Pour celà, vous formulez des propositions de combinaison. Pour chacune de ces combinaisons, le programme vous renseigne sur:

— le nombre de couleurs exactes, situées en bonne position,

— le nombre de couleurs exactes, mais situées en mauvaise position.

b) Déroulement d'une partie

Au départ, le programme vous fait choisir le nombre de positions (2 à 6) et le nombre de couleurs (2 à 9). Vous pouvez ainsi choisir vous-même la difficulté du jeu. Ce choix vous sera rappelé, tout au long de la partie, en bas à gauche de l'écran. Par exemple 4P signifie 4 positions et 9C signifie 9 couleurs. En même temps, les couleurs avec lesquelles vous pourrez jouer (compte tenu de votre choix) apparaissent en bas à droite sous formes de ronds de couleur. Jusqu'à 5 couleurs, il s'agit de ronds "pleins", au delà, vous retrouvez les mêmes couleurs sous forme de "cercles".

Le programme choisit alors, en secret, une combinaison de couleurs (les répétitions sont possibles). Pour deviner cette combinaison inconnue, vous faites une proposition en choisissant parmi les couleurs affichées. Pour celà, vous amenez la flèche noire sous la couleur voulue grâce aux deux touches de déplacement horizontal. Vous sélectionnez la couleur en appuyant sur la barre d'espace (un son vous confirme que votre couleur a été enregistrée). Un pion de cette couleur apparaît alors sur l'écran (en haut) à l'emplacement qui était signalé par une croix.

Cette croix apparait maintenant un peu plus loin pour vous montrer que le programme attend que vous choisissiez une seconde couleur. Celà se poursuit jusqu'à ce que vous ayez choisi autant de couleurs qu'il y a de positions prévues.

L'ordinateur vous fournit alors sa réponse sous forme de petits pions accompagnés de musique. Le rythme d'apparition de ces pions est irrégulier, ce qui donne l'impression que l'ordinateur "réfléchit" avant de répondre. La signification des pions est la suivante:

- une couleur à sa place pour chaque petit pion noir cerclé de blanc ;
- une bonne couleur, non à sa place, pour chaque petit pion blanc.

Si, par malchance (ou peut-être par chance !), votre proposition ne comporte aucune bonne couleur, vous entendrez simplement "zap".

Le programme vous demandera ensuite de formuler une autre proposition (à moins que vous ne soyez tombé juste au premier coup !). Le Jeu se poursuivra ainsi jusqu'à ce que vous trouviez la bonne combinaison ou que vous dépassiez le nombre maximum (22) de propositions. Dans ce dernier cas, le programme vous fournira sa combinaison cachée et vous pourrez observer tout à loisir l'écran rempli de pions de couleurs et éventuellement analyser votre jeu (peut-être, même, vérifier que le programme n'a pas menti ! eh ! on ne sait jamais...).

REMARQUE: Il peut vous arriver de vous tromper pendant que vous faites une proposition. Dans ce cas, vous pouvez annuler la proposition en cours en appuyant deux fois de suite sur la touche "return" (vous verrez les pions déjà proposés s'effacer). (Voir photo 3 hors texte : *Vue de l'écran pendant le déroulement du jeu*).

2. *Le programme*

```
100 REM ********** MASTER MIND **********
150 GOSUB 9000
160 REPEAT
170 : GOSUB 8000
180 : CP=0
190 : REPEAT
200 :   CP=CP+1
210 :   GOSUB 5000
220 :   GOSUB 4000
230 : UNTIL PE=NP OR CP>=NX
240 : IF PE=NP THEN GOSUB 2000 ELSE GOSUB 3000
250 : PLOT XD+1,26,"rejouez vous (O/N)"
260 : GET R$: GET R$
270 UNTIL R$<>"O"
290 END
```

```
2000 REM ----- SP GAGNE ------------------------------
2010 FOR Y=24 TO 26: FOR X=XD TO 38
2020 : PLOT X,Y,32
2030 NEXT X:  NEXT Y
2040 PLOT XD,24,"VOUS AVEZ TROUVE EN"
2045 PLOT XD,25,CHR$(12)+CHR$(3)+STR$(CP)+" COUPS"
2050 PLOT XD+2,25," "
2060 WAIT 300
2070 RETURN
3000 REM ----- SP PERDU -----------------------------
3008 FOR Y=24 TO 26: FOR X=XD TO 38
3020 : PLOT X,Y,32
3030 NEXT X: NEXT Y
3035 ZAP
3040 PLOT XD-1,24,12: PLOT XD,24,"LA COMBINAISON ETAIT"
3050 X=XD: Y=25
3060 FOR I=1 TO NP
3065 : CA=P1
3070 : IC=TI(I): IF IC>5 THEN CA=P2: IC=IC-5
3080 : PLOT X,Y,CO(IC): PLOT X+1,Y,CA
3085 : X=X+2
3090 NEXT I
3100 WAIT 200
3120 RETURN
4000 REM ----- SP ANALYSE DE LA COMBINAISON PROPOSEE -------
4010 FOR I=1 TO NP: TP(I)=TI(I): NEXT I
4020 PE=0
4030 FOR I=1 TO NP
4040 : IF TP(I)=PR(I) THEN PE=PE+1: TP(I)=0: PR(I)=-1
4050 NEXT I
4055 CE=0
4060 FOR I=1 TO NP
4070 : FOR J=1 TO NP
4080 :  IF TP(J)=PR(I) THEN CE=CE+1: TP(J)=0: GOTO 4100
4090 : NEXT J
4100 NEXT I
4110 PLOT X+1,Y,7: X=X+2
4120 IF PE=0 THEN 4200
4130 FOR I=1 TO PE
4140 : WAIT TM*RND(1)
4150 : MUSIC 1,02,N2,0: PLAY 1,0,1,200
4160 : PLOT X,Y,R1
4170 : X=X+1
4180 NEXT I
4200 IF CE=0 THEN 4300
```

```
4210 FOR I=1 TO CE
4220 : WAIT TM*2*RND(1)
4230 : MUSIC 1,03,N3,0: PLAY 1,0,1,200
4240 : PLOT X,Y,R2
4250 : X=X+1
4260 NEXT I
4300 WAIT 200
4310 IF PE=0 AND CE=0 THEN ZAP
4320 RETURN
5000 REM ----- SP LECTURE DE LA COMBINAISON PROPOSEE -----
5010 PLOT XD,24,12: PLOT XD+1,24,"faites votre choix"
5015 PLOT 8,26,STR$(CP): PLOT 8,26," "
5020 IP=XD
5030 PLOT IP,26,0: PLOT IP+1,26,94
5040 FOR I=1 TO NP
5050 : PLOT 12,26,STR$(I): PLOT 12,26," "
5060 : X=2*I-1: IF CP>11 THEN X=X+19
5070 : Y=2*CP-1: IF CP>11 THEN Y=Y-22
5080 : PLOT X-1,Y,7: PLOT X,Y,43
5090 : CL=PEEK(#208)
5100 : IF CL=#38 THEN 5090
5110 : IF CL=#84 THEN 5200
5112 : IF CL<>#AF THEN 5120
5115 : PLOT X-1,Y,"              ": PLOT IP,26," "
5117 : GOTO 5040
5120 : PLOT IP,26,"  "
5130 : IP=IP-2*(CL=#AC)*(IP>XD)+2*(CL=#BC)*(IP<XF)
5140 : PLOT IP,26,0: PLOT IP+1,26,94
5145 : WAIT 20
5150 : GOTO 5090
5200 : K=(IP-XD)/2+1
5210 : PR(I)=K
5220 : MUSIC 1,01,N1,0: PLAY 1,0,1,400
5230 : CA=P1: IC=K
5240 : IF IC>5 THEN IC=IC-5: CA=P2
5250 : PLOT X-1,Y,CO(IC): PLOT X,Y,CA
5260 : WAIT 40
5270 NEXT I
5275 PLOT IP,26,"  "
5300 PLOT 18,24,"                    "
5310 RETURN
```

```
8000 REM ----- SP INITIALISATION D'UNE PARTIE ----------
8010 CLS: PAPER 6: INK4
8020 PLOT 2,10,"COMBIEN DE POSITIONS (2 A 6)?"
8030 GET R$: NP=VAL(R$): IF NP<2 OR NP>6 THEN 8030
8040 PLOT 32,10,R$
8050 PLOT 2,15,"COMBIEN DE COULEURS (2 A 9)?"
8060 GET R$: NC=VAL(R$): IF NC<2 OR NC>10 THEN 8060
8070 PLOT 32,15,R$
8080 WAIT 300: CLS: PAPER PA: INK IN
8085 XF=XD+2*(NC-1)
8090 ID=XD: CA=P1
8100 FOR I=1 TO NC
8110 : IC=I: IF IC>5 THEN IC=IC-5: CA=P2
8120 : PLOT ID,25,CO(IC)
8130 : PLOT ID+1,25,CA
8140 : ID=ID+2
8150 NEXT I
8160 FOR I=0 TO 2
8170 : PLOT 1,24+I,PB+16
8180 : PLOT 6,24+I,PC+16
8190 NEXT I
8200 PLOT XD-1,25,16
8210 PLOT 2,25,STR$(NP): PLOT 2,25," ": PLOT 4,25,"P"
8220 PLOT 2,26,STR$(NC): PLOT 2,26," ": PLOT 4,26,"C"
8230 PLOT 7,25,"COUP POSIT"
8240 FOR I=1 TO NP
8250 : TI(I)=INT(RND(1)*NC+1)
8260 NEXT I
8270 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU ---------------
9010 PA=0: IN=4: PB=3: PC=6
9020 DIM CO(5), TI(9),PR(9),TP(9)
9030 NX=22: TM=100: XD=19
9040 O1=3: N1=1: O2=3: N2=8: O3=4: N3=1
9050 POKE #26A,10
9060 DATA 1,3,5,7,4
9070 FOR I=1 TO 5: READ CO(I): NEXT I
9080 P1=64: P2=123: R1=95: R2=96
9090 DATA 12,30,63,63,63,63,30,12
9100 C=P1: GOSUB 9500
9110 DATA 12,30,51,51,51,51,30,12
9120 C=P2: GOSUB 9500
9130 DATA 0,0,12,18,18,12,0,0
9140 C=R1: GOSUB 9500
9150 DATA 0,0,12,30,30,12,0,0
9160 C=R2: GOSUB 9500
9170 RETURN
```

```
9500 REM ----- SP INIT CARAC GRAPHIQUES ---------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier les couleurs des pions

Remplacez les cinq nombres de l'instruction 9060 par cinq valeurs de votre choix. Veillez à ce qu'elles soient toutes différentes:

Par exemple:

9060 DATA 7,1,2,4,5

vous fournira, dans cet ordre, les couleurs: blanc, rouge, vert, bleu et violet.

2. Pour modifier les sons

Remplacez les valeurs de l'instruction 9040 par des nombres de votre choix, sachant que O1, O2 et O3 correspondent à des numéros d'octave (donc de 0 à 6) et N1, N2, N3 à des numéros de note (donc de 1 à 12). D'autre part:

O1,N1 correspondent à la note émise lors de la sélection d'une couleur.
O2,N2 correspondent à la note émise pour chaque "bonne position".
O3,N3 correspondent à la note émise pour chaque "bonne couleur".

3. Pour avoir un moyen de connaître la combinaison secrète, sans devoir finir la partie

Ajoutez la ligne.

5111 IF CL=≠AD THEN CP=NX:RETURN

et complétez 210 comme ceci:

```
210 GOSUB 5000: IF CP > =NX THEN 230
```

Il suffira alors que vous pressiez la touche "DEL" pendant la phase de sélection des couleurs pour que la partie se termine en faisant apparaître la combinaison cachée. Notez bien qu'il ne vous sera plus possible de poursuivre cette partie ainsi interrompue.

4. Description du programme

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.

b) Le programme principal (100-299)

150 appelle le sous programme d'initialisation du jeu.

160 et 270 répètent les instructions 170 à 240 (déroulement d'une partie) jusqu'à ce que vous disiez que vous ne souhaitez plus rejouer:

170 appelle le sous-programme d'initialisation d'une partie.

180 initialise le compteur de coups joués.

190 et 230 répètent les instructions 200 à 220 (jeu d'un coup) jusqu'à ce que vous ayez gagné ou que le nombre de coups joués atteigne 22:

200 incrémente le compteur de coups.

210 appelle le sous-programme qui "lit" la combinaison que vous proposez.

220 appelle le sous-programme qui analyse cette proposition.

240 appelle, suivant le cas, le sous-programme "gagné" ou le sous-programme "perdu".

250-260 vous demandent si vous voulez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9170)

9010 fixe les diverses couleurs intervenant dans le programme.

9020 réserve les tableaux.

9030-9040 fixent le nombre maximum de coups, les sons employés au cours de la partie,... (voir variables).

9050 supprime le bip du clavier et le curseur d'écran.

9060-9070 détermine (à partir du DATA 9060) les cinq couleurs servant à représenter les pions. Notez qu'on obtiendra 9 "couleurs" effectives en utilisant deux formes de pions.

9080-9160 fabriquent les caractères graphiques: pion plein, pion cerclé, petit pion cerclé, petit pion plein.

d) Le sous-programme d'initialisation d'une partie (8000-8270)

8010-8080 vous font choisir le nombre de positions et le nombre de couleurs (le dialogue se fait en bleu sur fond cyan).

8085-8150 dessinent, en bas à droite, les différents pions correspondants au nombre de couleurs choisies.

8160-8190 fixent les couleurs de fond des trois lignes du bas de l'écran.

8200-8230 affichent en bas d'écran, le nombre de positions et le nombre de couleurs.

8240-8260 tire au hasard la combinaison qu'il vous faudra deviner. Notez que chaque "couleur" du jeu est représentée par un entier compris entre 1 et 9.

e) Le sous programme de lecture de la combinaison proposée (5000-5310)

5010-5030 affichent le message clignotant "faites votre choix", le numéro du coup et affichent la flèche de sélection de couleurs en position initiale.

5040 à 5270 lisent les couleurs proposées. Pour chaque couleur:

5050-5080 affichent le numéro du pion que vous êtes en train de choisir et font apparaître "+" à l'emplacement correspondant.

5090-5260 examinent le clavier:

5110 détecte la frappe de la barre d'espace.

5112 détecte la frappe de la touche "return".

5115-5117 : cas où "return" a été frappée.

5120-5150 : cas d'une touche autre que espace ou return (cas usuel) : on calcule la nouvelle position de la flèche de sélection et on l'affiche.

5200-5260 : cas ou espace a été frappé : on range le numéro de la couleur proposée dans le tableau PR ; on émet un son montrant que la proposition a été enregistrée et on affiche le pion ainsi choisi à l'emplacement correspondant.

5275-5300 effacent la flèche et le message.

f) Le sous-programme d'analyse de la combinaison proposée (4000-4320)

4010 recopie dans le tableau de travail TP, la combinaison tirée par l'ordinateur. Celà permet d'effectuer certaines modifications dans TP (lors de l'analyse de la proposition) sans altérer le tableau TI.

4020-4050 comptent le nombre (PE) de couleurs bien placées.

4055-4100 comptent le nombre (CE) de couleurs présentes dans la combinaison cachée, mais mal placées. Notez bien qu'il a fallu éviter de prendre en considération les couleurs déjà comptées dans PE.

4110-4180 affichent, s'il y a lieu, en musique un nombre de petits pions cerclés égal au nombre de couleurs à leur place.

4200-4260 affichent, s'il y a lieu, en musique, un nombre de pions blancs égal au nombre de bonnes couleurs mal placées.

4310 émet un "zap" si vous n'avez aucune bonne couleur dans votre proposition.

g) Le sous-programme "gagné" (2000-2070)

Il affiche, en bas, à droite le message "vous avez trouvé en" suivi du nombre de coups.

h) Le sous-programme "perdu" (3000-3120)

Après effacement de la partie appropriée de l'écran, il vous affiche la combinaison choisie.

5. *Liste des variables*

PA couleur du fond (ici noir).

IN couleur d'encre utilisée pour les trois lignes inférieures.

PB couleur de fond de la partie gauche des trois lignes inférieures.

PC couleur de fond des trois lignes inférieures (à l'exception de la partie gauche).

CO(5) tableau contenant les couleurs utilisées pour les pions (on obtient jusqu'à 9 "couleurs" en jouant sur l'aspect des pions).

TI(9) tableau contenant la combinaison tirée par le programme.

PR(9) tableau contenant la proposition du joueur.

TP(9) tableau de travail employé dans l'analyse de la proposition.

NX nombre maximum de coups.

TM temps d'attente maximum utilisé lors de la réponse du programme à une proposition.

XD abscisse de début de l'emplacement où s'affichent les couleurs possibles (en bas à droite).

O1,N1 note (octave + numéro dans l'octave) de la note jouée lorsque vous sélectionnez une couleur.

O2,N2 note (octave + numéro dans l'octave) de la note jouée en accompagnement de chaque petit pion noir cerclé de blanc.

O3,N3 note (octave + numéro dans l'octave) de la note jouée en accompagnement de chaque petit pion blanc.

P1 code ASCII du caractère représentant un pion "plein".

P2 code ASCII du caractère représentant un pion "cerclé".

R1 code ASCII du caractère représentant un petit pion noir cerclé de blanc.

R2 code ASCII du caractère représentant un petit pion blanc.

NP nombre de positions choisies pour la partie en cours.

NC nombre de couleurs choisies pour la partie en cours.

XF abscisse de fin de la zone servant à afficher les différentes couleurs en vue de leur sélection.

ID,IP indices servant à décrire les positions des différentes couleur de la zone de sélection.

CP numéro du coup.

X,Y coordonnées du pion courant (pion "ordinaire" ou petit pion).

K numéro de la couleur sélectionnée.

PE nombre de couleurs bien placées dans la proposition courante (tableau PR).

CE nombre de bonnes couleurs, mal placées, dans la proposition courante (tableau PR).

Variables auxiliaires

C R$ I CA IC

11
Gobe mouches

1. Le jeu

Imaginez que vous soyez une petite araignée obligée de chasser les mouches pour se nourrir.

Heureusement celles-ci pullulent dans votre environnement et vous n'avez que l'embarras du choix. Toutefois vous ne pouvez vous déplacer que le long d'un fil (de votre toile).

Vous vous déplacez le long du bas de l'écran grâce aux touches fléchées :

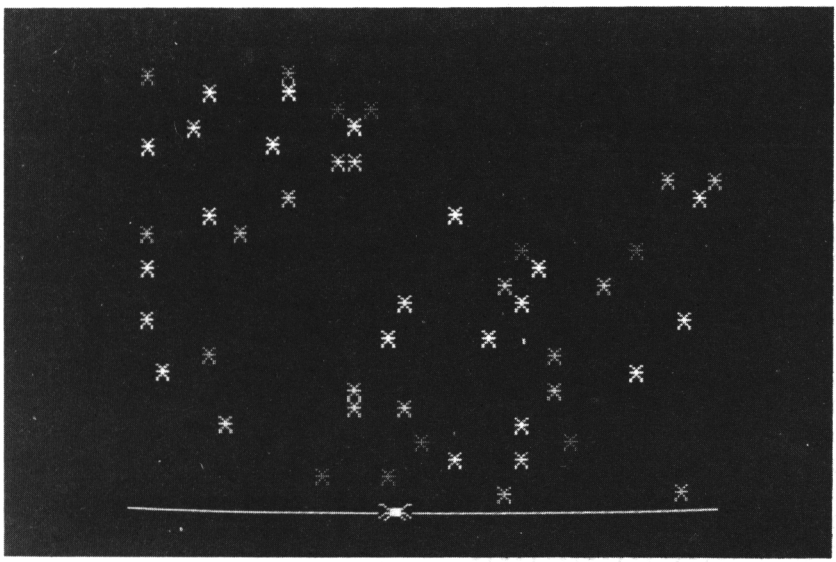
"←" et "→" vous font avancer à vitesse lente dans la direction indiquée.

"↓" et "↑" vous font avancer, dans les mêmes directions, à une vitesse triple de la précédente.

Pour avaler une mouche, il vous suffit de l'intercepter dans son vol (qui se déroule toujours verticalement). Chaque mouche rapporte un nombre de

points qui dépend de sa couleur (rouge 1 ; vert 2 ; jaune 3 ; bleu 4 ; violet 5 ; bleu ciel 6 ; blanc 7).

La partie s'achève au bout d'un temps déterminé à l'avance. Bonne chance et... bon appétit.



Vue de l'écran pendant le déroulement du programme

2. *Le Programme*

```
100 REM ********** GOBE MOUCHES ************************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
135 : FOR T=1 TO TM
140 :  X1=1+37*RND(1): X2=1+37*RND(1)
150 :  IM=1+RND(1)*7
160 :  IF SCRN(X,25)<>32 THEN SHOOT: SC=SC+SCRN(0,25)
170 :  IF SCRN(X+1,25)<>32 THEN SHOOT: SC=SC+SCRN(0,25)
172 :  C=PEEK(#208): IF C=#38 THEN 180
```

```
175 :  X=X-(C=#AC)*(X>2)+(C=#BC)*(X<37)-3*(C=#B4)*(X>4)+3*(C=#9C)*(
X<35)
180 :  PRINT CHR$(11);
190 :  PLOT 0,26,F$
195 :  PLOT 0,26,IN: PLOT X,26,G$
200 :  PLOT 0,1,IM: PLOT X1,1,MO: PLOT X2,1,MO
230 : NEXT
250 : CLS: PLOT 5,10,"SCORE": PLOT 15,10,STR$(SC): PLOT 15,10," "
260 : PLOT 5,12,"M-SCORE": PLOT 15,12,STR$(MS): PLOT 15,12," "
270 : IF SC>MS THEN MS=SC
280 : WAIT 100: PLOT 5,20,"voulez vous rejouer(O/N)"
290 : GET R$: GET R$
300 UNTIL R$<>"O"
310 END
8000 REM ----- SP INITIALISATION D'UNE PARTIE ----------
8010 CLS: SC=0
8020 X=15: PLOT X,26,C$
8030 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU ---------------
9010 PA=0: IN=3: PAPER PA: INK IN
9015 MO=94: G1=95: G2=96: F=64
9020 G$=CHR$(G1)+CHR$(G2)
9030 TM=400: MS=0
9040 POKE #26A,10
9050 DATA 17,10,4,31,4,10,17,17
9060 C=MO: GOSUB 9500
9070 DATA 48,8,7,3,15,19,36,8
9080 C=G1: GOSUB 9500
9090 DATA 3,4,56,48,60,50,9,4
9100 C=G2: GOSUB 9500
9110 DATA 0,0,0,0,63,0,0,0
9120 C=F: GOSUB 9500
9130 F$="": FOR I=1 TO 38: F$=F$+CHR$(F): NEXT I
9140 RETURN
9500 REM ----- SP CREATION CARACT GRAPHIQUES ------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier la couleur de l'araignée et de son fil

Remplacez la valeur 3 de l'instruction 9010 par un nombre de votre choix, compris entre 1 et 7.

2. Pour que toutes les mouches aient la même couleur

Remplacez l'instruction 150 par:

```
150 IM=n
```

où n désigne un nombre compris entre 1 et 7.

Par exemple:

```
150 IM=5
```

produira des mouches violettes.

REMARQUE: dans ce cas (toutes les mouches de la même couleur), chaque mouche mangée rapportera alors le même nombre de points.

3. Pour modifier la durée d'une partie

Remplacez, en 9030, la valeur 400 par la valeur de votre choix.

4. Pour que toutes les mouches rapportent le même nombre de points

Modifiez ainsi les instructions 160 et 170.

```
160 IF SCRN(X,25) <> 32 THEN SHOOT: SC=SC+1
170 IF SCRN(X+1,25) <> 32 THEN SHOOT: SC=SC+1
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Scroll.
— Configuration clavier/écran.

b) Le programme principal

110 appelle le sous-programme d'initialisation du jeu.

120 et 300 répètent les instructions 130 et 290 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer :

130 appelle le sous-programme d'initialisation d'une partie.

135 et 230 répètent 400 fois les instructions 140 à 200 :

140 et 150 calculent les positions de deux nouvelles mouches ainsi que leur couleur.

160 et 170 examinent si l'araignée rencontre une mouche et, le cas échéant, augmentent le score du nombre de points correspondants.

172-175 déterminent la nouvelle position de l'araignée lorsqu'une des touches a été pressée.

180 fait défiler vers le bas l'ensemble de l'écran (scroll).

190-200 dessinent le fil et l'araignée dans sa nouvelle position.

250-270 affichent votre score, le meilleur score et actualisent ce dernier.

280-280 vous demandent si vous désirez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9140)

9010 fixe les couleurs d'encre et de fond.

9015 fixe les codes des caractères représentant les mouches, l'araignée et le fil.

9020 fabrique la chaîne de deux caractères représentant l'araignée.

9030 fixe le temps d'une partie et initialise le meilleur score.

9040 supprime le bip du clavier et le curseur d'écran.

9050-9100 fabriquent les caractères graphiques servant à représenter les mouches, l'araignée et son fil.

d) Le sous-programme d'initialisation d'une partie (8000-8030)

8010 efface l'écran et initialise le score.

8020 dessine l'araignée dans sa position initiale.

5. Liste des variables

PA	couleur du fond (ici : noir).
IN	couleur d'encre utilisée pour l'araignée et son fil.
MO	code ASCII du caractère utilisé pour représenter une mouche.
G1	code ASCII du caractère représentant la partie gauche de l'araignée.
G2	code ASCII du caractère représentant la partie droite de l'araignée.
F	code ASCII du caractère utilisé pour représenter le fil.
G$	chaîne de deux caractères représentant l'araignée.
F$	chaîne de 38 caractères représentant le fil.
TM	durée d'une partie (exprimée en nombre de tours de boucles).
MS	meilleur score.
SC	score.
T	compteur de tours de boucle.
X	position de l'araignée sur son fil.
X1,X2	positions des deux dernières mouches (tirées au hasard).
IM	couleur d'encre des deux dernières mouches (tirée au hasard).

Variables auxiliaires

C SC R$ I (dans le sous-programme 9500).

12
KARTING

1. Le jeu

La piste est là, devant vos yeux, avec ses bottes de paille qui en délimitent le tracé. Votre bolide ne tarde pas à se présenter, moteur tournant au ralenti. Il vous suffit d'y monter et d'essayer de le conduire sur le plus grand nombre possible de kilomètres et aussi vite que vous le pouvez.

Le bolide apparaît en haut à gauche de l'écran. Le programme vous demande de signaler que vous êtes prêt en tapant sur une touche quelconque. Votre voiture se met alors en marche, à petite vitesse. A partir de cet instant, vous ne pourrez plus l'arrêter et la partie se terminera inexorablement... dans le décor.

Vous dirigez votre bolide à l'aide des quatre touches fléchées (ou, le cas échéant, avec une poignée de jeu). Vous accélérez à l'aide de la touche "return" et vous freinez avec la barre d'espace. Le bruit du moteur vous renseignera sur votre vitesse.

A la fin de la partie (c'est-à-dire quand vous avez quitté la piste), le programme vous indique la distance parcourue et votre temps de parcours. Votre score est alors calculé en fonction de ces deux éléments (il est proportionnel à la distance et à la vitesse moyenne). Vous pouvez le comparer avec le meilleur score déjà réalisé. (Voir photo 4 hors-texte : *Vue de l'écran pendant le déroulement du jeu*).

2. Le programme

```
100 REM ********** KARTING **************************
110 GOSUB 8500: GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : IF PEEK(768)=176 THEN 180
150 : DN=PEEK(735)-135
160 : IF DN=6 OR DN=25 THEN GOSUB 5000
170 : IF DN<1 OR DN>4 OR(DN-DL=1 AND DN<>3)OR(DL-DN=1 AND DL<>3) TH
EN DN=DL
180 : XN=X+(DN=1)-(DN=2): YN=Y+(DN=4)-(DN=3)
190 : IF SCRN(XN,YN)<>32 THEN GOSUB 3000: GOTO 240
200 : PLOT X,Y,32: X=XN: Y=YN: DL=DN: PLOT X,Y,CV(DL)
210 : KM=KM+1: T=T+4-V
220 : IF V<3 THEN WAIT 3: IF V<2 THEN WAIT 7
230 : GOTO 140
240 : PLOT 19,25,"rejouez vous (O/N)"
245 : GET R$: GET R$
250 UNTIL R$<>"O"
270 END
3000 REM ---- SP SORTIE DE PISTE ----------------------
3010 EXPLODE
3020 PLOT 1,24,"KM": PLOT 3,24,STR$(INT(KM)): PLOT 3,24," "
3030 PLOT 10,24,"TEMPS":PLOT 15,24,STR$(INT(T)): PLOT 15,24," "
3040 SC=INT(KM*KM/T)
3050 PLOT 1,25,"SCORE": PLOT 10,25,12
3060 PLOT 11,25,STR$(SC): PLOT 11,25," "
3065 PLOT 1,26,"M-SCORE": PLOT 11,26,STR$(MS): PLOT 11,26," "
3070 WAIT 300
3075 IF SC>MS THEN MS=SC
3080 PLOT X,Y,32
3090 RETURN
```

```
5000 REM ---- SP CHANGEMENT DE VITESSE -----------------
5010 V=V+(DN=6)*(V<3)-(DN=25)*(V>1)
5020 GOSUB 6000
5030 RETURN
6000 REM ---- SP SONORISATION ------------------------
6010 F=10+40*(3-V)
6015 PLAY 0,1,3,F
6020 SOUND 4,30,0
6030 RETURN
8000 REM ---- SP INITIALISATION D'UNE PARTIE ----------
8010 KM=0: T=0
8020 X=2: Y=2: DN=2: DL=DN: V=1
8030 PLOT X,Y,CV(2): PING: GOSUB 6000
8035 FOR I=0 TO 2: PLOT 1,24+I,BL$: NEXT I
8040 PLOT 2,25,"pour commencer tapez sur une touche"
8050 GET R$: PLOT 1,25,BL$
8070 RETURN
8500 REM ---- SP TRACE DE LA PISTE --------------------
8510 DATA "àààààààààààààààààààààààààààààààààààà"
8520 DATA "à                                    à"
8530 DATA "à                                    à"
8540 DATA "à                                    à"
8550 DATA "à      àààààààààààààààààààààààà    à"
8560 DATA "à    à          àààà              à    à"
8570 DATA "à   à            àà                à    à"
8580 DATA "à   à            à                 à    à"
8590 DATA "à   à    ààà     à     àààààà      à    à"
8600 DATA "à   à   ààààà    à    àààààààà     à    à"
8610 DATA "à   à   ààààà    à    àààààààà     à    à"
8620 DATA "à   à   ààààà    à    àààààààà     à    à"
8630 DATA "à   à    àààà    à    àààààà       à    à"
8640 DATA "à   à     àà     à      ààà        à    à"
8650 DATA "à   à     à      à       àà        à    à"
8660 DATA "à   à    à      àà         à        à    à"
8670 DATA "à   à   à      àààà        à        à    à"
8680 DATA "à   à   à     àààààà       à        à    à"
8690 DATA "à   à   à    ààààààà      à        à    à"
8700 DATA "à       à     ààààààà     à        à    à"
8710 DATA "à       à                 à             à"
8720 DATA "à       à                  à          àà"
8730 DATA "à        à                  à        ààà"
8740 DATA "àààààààààààààààààààààààààààààààààààà"
8750 CLS
8800 FOR I=0 TO 23
8810 : READ CH$: PLOT 1,I,CH$
8820 NEXT I
8830 RETURN
```

```
9000 REM ---- SP INITIALISATION DU JEU -----------------
9010 PA=3: IN=1: PAPER PA: INK IN
9020 MS=0
9040 BL$="": FOR I=1 TO 37: BL$=BL$+" ": NEXT I
9050 DIM CV(4)
9060 CP=64: CV(1)=93: CV(2)=94: CV(3)=95: CV(4)=96
9070 DATA 0,0,30,30,30,30,0,0
9080 C=CP: GOSUB 9500
9090 DATA 0,0,17,63,63,63,17,0
9100 C=CV(1): GOSUB 9500
9105 DATA 0,0,34,63,63,63,34,0
9110 C=CV(2): GOSUB 9500
9115 DATA 0,15,06,06,06,15,06,0
9120 C=CV(3): GOSUB 9500
9122 DATA 0,06,15,06,06,06,15,0
9125 C=CV(4): GOSUB 9500
9130 POKE #26A,10
9150 RETURN
9500 REM ---- SP INIT CARACTERES GRAPHIQUES -----------
9510 AD=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AD+I,XX
9540 NEXT I
9550 RETURN
```

REMARQUE: les caractères "à" de la liste correspondent en fait au caractère @ (c'est ainsi qu'il apparaît sur le clavier et à l'écran).

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier la couleur du fond (ici jaune)

Remplacez, en 9010, la valeur 3 par un nombre de votre choix (entre 0 et 7).

2. Pour modifier la couleur des "bottes de paille" et de la voiture (ici rouges)

Remplacez, en 9010, la valeur 1 par un nombre de votre choix (entre 0 et 7).

3. Pour modifier le tracé de la piste

Il vous suffit de modifier les instructions "DATA" des lignes 8510 à 8740 en tenant compte des remarques suivantes:

- Chacune de ces 24 instructions représente une ligne de la piste.
- Chaque instruction contient une chaîne d'au maximum 38 caractères et formée uniquement de caractères "espace" et " @ ". Les espaces matérialisent la piste tandis que " @ " correspond aux "bottes de paille".
- Il est absolument indispensable que la piste soit toujours bordée de " @ ". En effet, pour obtenir la vitesse maximum, le programme ne teste pas les "débordements" d'écran ; il se contente de vérifier s'il y a ou non collision avec une botte de paille. Dans ces conditions, l'absence de " @ " en bord d'écran pourrait entraîner un mauvais fonctionnement du programme.

4. Si le jeu vous paraît trop rapide

Vous pouvez réduire les vitesses 1 et 2, en modifiant la ligne 220. Il vous suffit de savoir qu'actuellement:

— la vitesse 1 correspond à une attente (instruction WAIT) de 10 (3 de WAIT 3 + 7 de WAIT 7).

— la vitesse 2 correspond à une attente de 3.

Ainsi, en remplaçant 220 par:

```
220 IF V<3 THEN WAIT5:IF V<2 THEN WAIT 10
```

vous obtiendrez:

— une attente de 15 (5 + 10) pour la vitesse 1.

— une attente de 5 pour la vitesse 2.

5. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE2" comme décrit en annexe. Branchez la poignée du côté droit. Remplacez les instructions 140 à 180 par:

```
150 : DN=PEEK(#400): IF DN=0 THEN DN=DL
160 XN=X+((DN AND 1)=1)-((DN AND 2)=2)
170 YN=Y-((DN AND 4)=4)+((DN AND 8)=8)
```

(attention, les lignes 140 et 180 disparaissent).

Enfin, remplacez les lignes 9050 à 9125 par:

```
9050 DIM CV(10)
9060 CP=64: CV(1)=93: CV(2)=94: CV(3)=0: CV(4)=95: CV(5)=96
9065 CV(6)=97: CV(7)=0: CV(8)=98: CV(9)=99: CV(10)=100
9070 DATA 0,0,30,30,30,30,0,0
9080 C=CP: GOSUB 9500
9090 DATA 0,0,17,63,63,63,17,0
9100 C=CV(1): GOSUB 9500
9105 DATA 0,0,34,63,63,63,34,0
9110 C=CV(2): GOSUB 9500
9115 DATA 0,15,6,6,6,15,6,0
9117 C=CV(4): GOSUB 9500
9118 DATA 0,4,6,4,7,28,56,20,0
9119 C=CV(5): GOSUB 9500
9120 DATA 0,8,24,61,14,7,10,0
9121 C=CV(6): GOSUB 9500
9122 DATA 0,6,15,6,6,6,15,0
9123 C=CV(8): GOSUB 9500
9124 DATA 0,20,56,28,47,6,4,0
9125 C=CV(9): GOSUB 9500
9126 DATA 0,10,7,14,61,24,8,0
9127 C=CV(10): GOSUB 9500
```

4. Description du programme

a) Techniques utilisées décrites en annexes

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.
— Poignées de jeu (facultatif).

b) Le programme principal

110 appelle les sous-programmes d'initialisation du jeu et de dessin de la piste.

120 et 250 répètent les instructions 130 à 245 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

130 appelle le sous-programme d'initialisation d'une partie.

140-230 sont répétées jusqu'à ce que la voiture quitte la piste:

140-150 examinent le clavier.

160 appelle le sous-programme de changement de vitesse lorsque l'une des touches "return" ou "espace" a été pressée.

170 détermine le nouveau déplacement DN de la voiture, en tenant compte de son sens de marche et éventuellement de la touche pressée (flèche). Notez que les demi-tours sont interdits.

180 détermine la nouvelle position de la voiture.

190 examine si la voiture sort de la piste. Dans ce cas, le sous-programme de sortie de piste est appelé et la partie est interrompue.

200 efface la voiture de sa position actuelle et la dessine dans sa nouvelle position, en tenant compte de son sens de marche.

210 actualise le temps et le kilométrage.

220 réalise une attente fonction de la vitesse de la voiture.

240-245 vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9150)

9010 fixent les couleurs du fond et de l'encre.

9020 initialise le meilleur score.

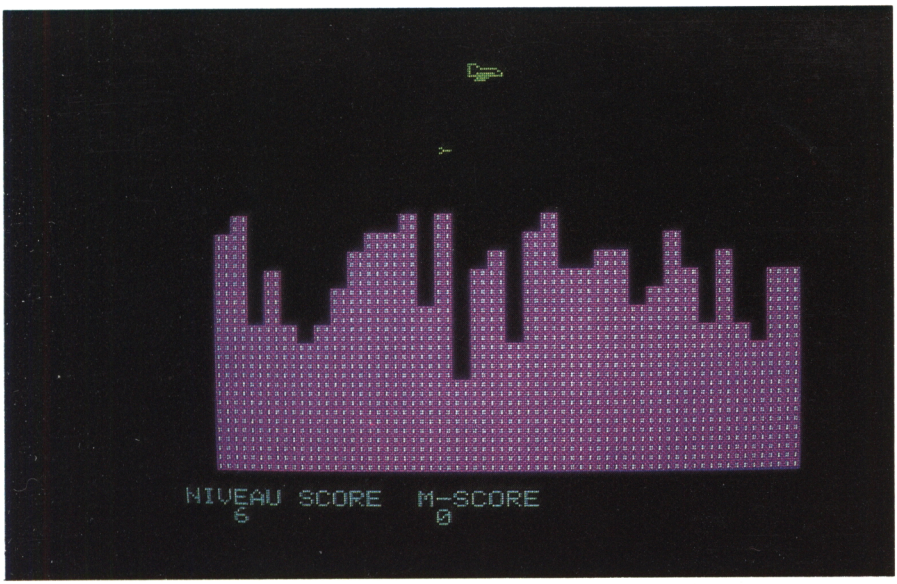
9040 prépare une chaîne de 37 caractères blancs.

9050-9125 fabriquent les caractères graphiques représentant les "bottes de paille" et la voiture dans chacune de ses quatre directions de déplacement.

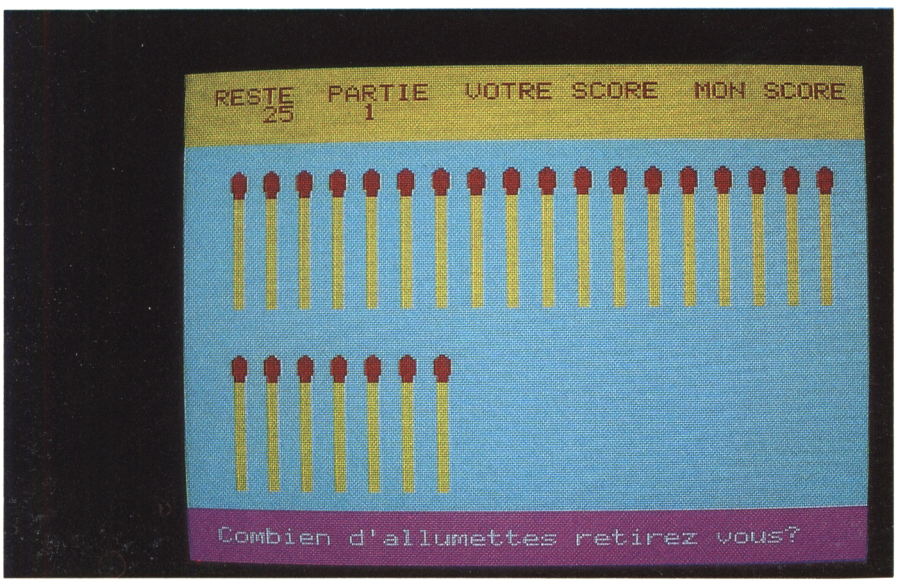
9130-9140 suppriment le bip et le curseur et effacent l'écran.

d) Le sous-programme de tracé de la piste (8500-8830)

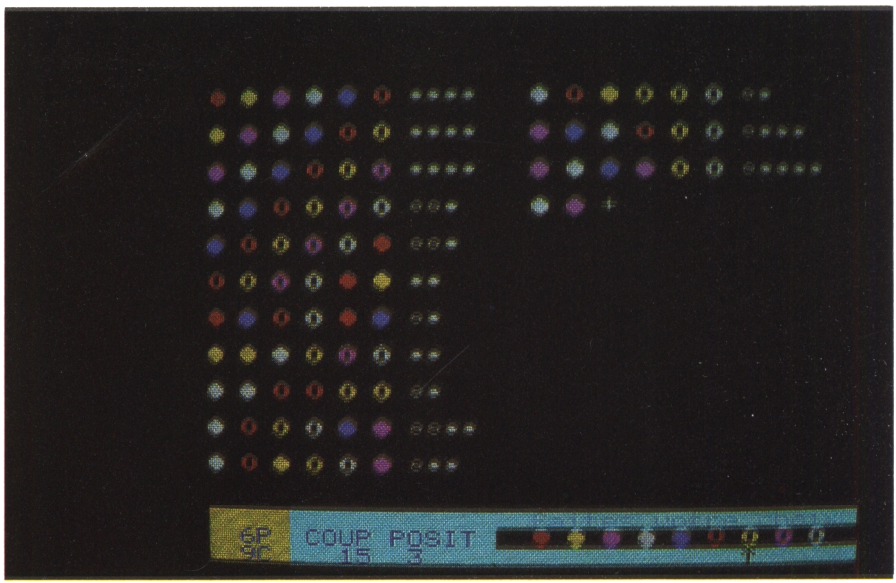
8800-8820 dessinent la piste en utilisant les instructions DATA 8510 à 8740. Notez que chaque caractère " @ " correspond, en fait, à une botte de paille.



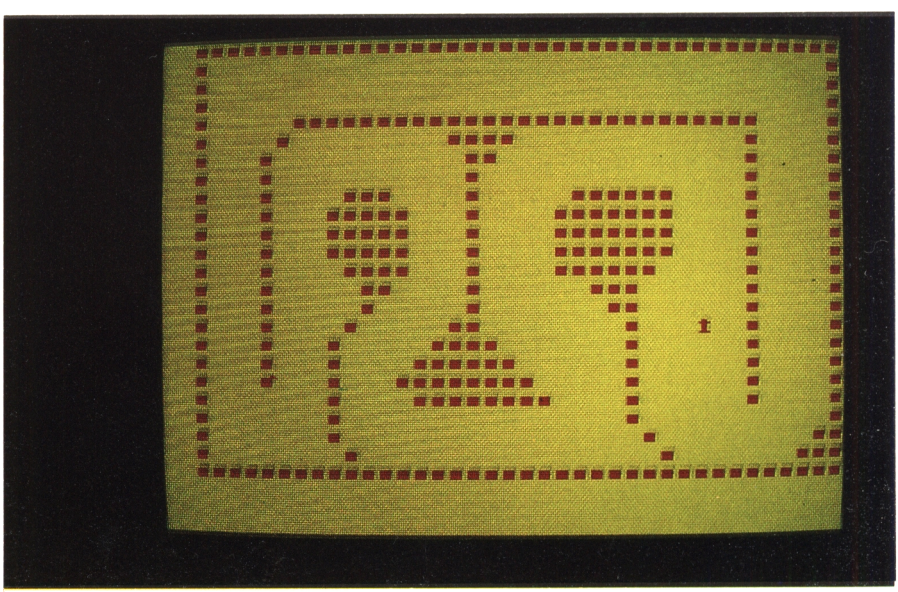
1. *Raid de nuit*



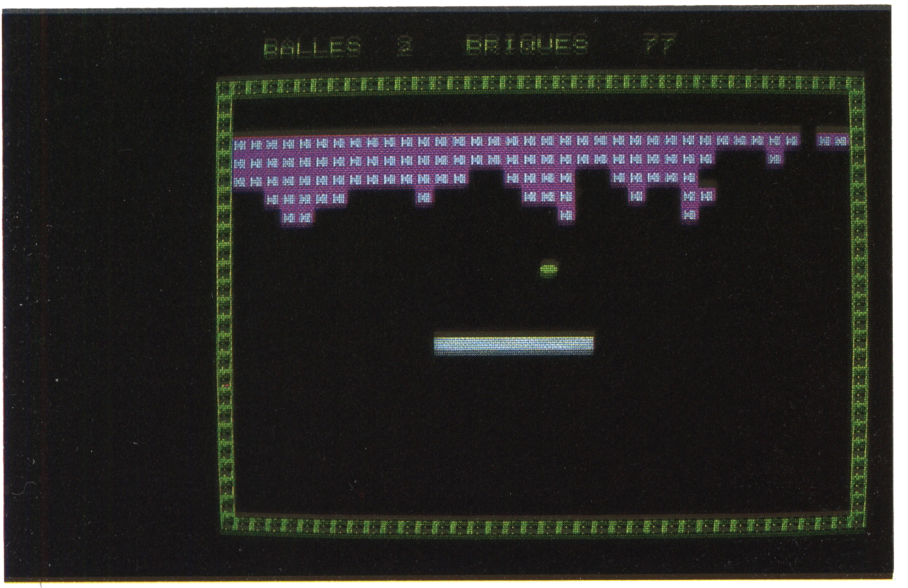
2. *Les allumettes suédoises*



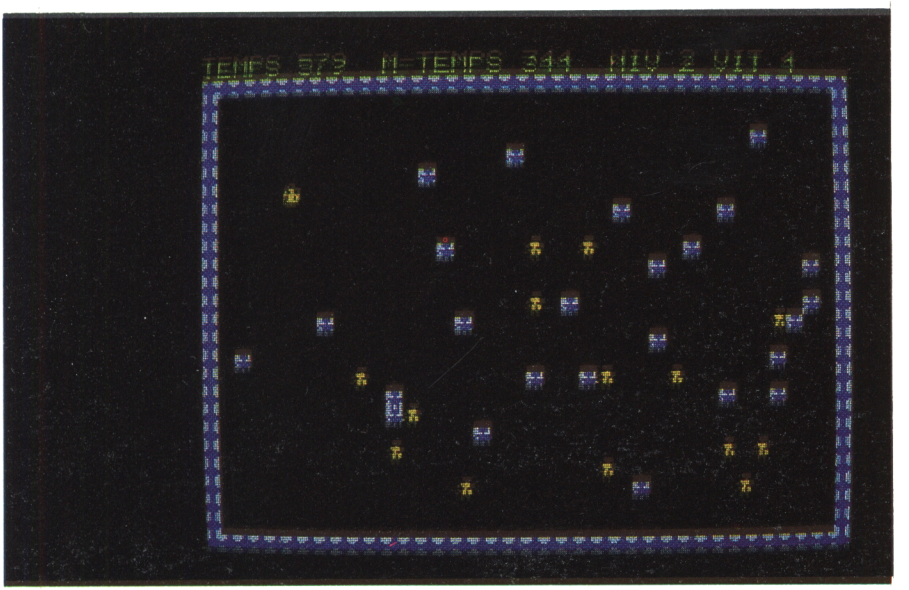
3. *Mastermind*



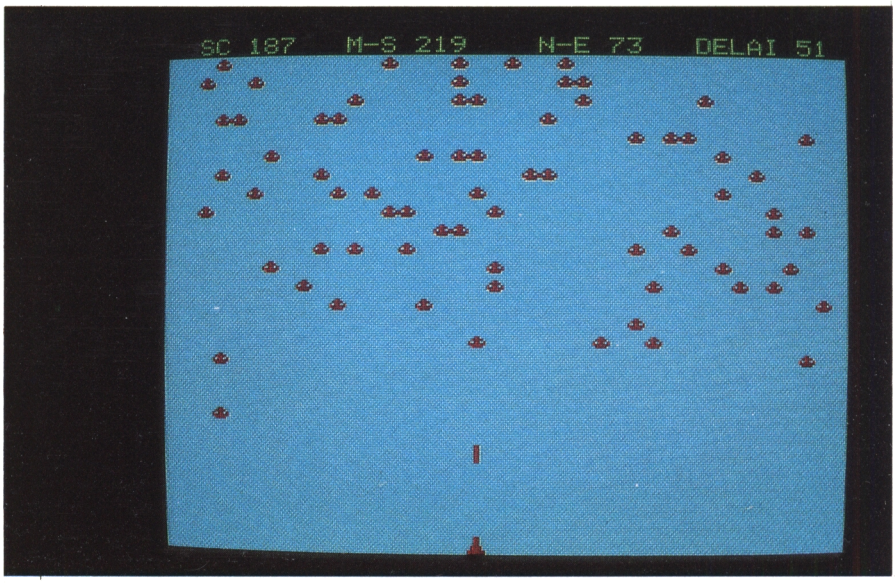
4. *Karting*



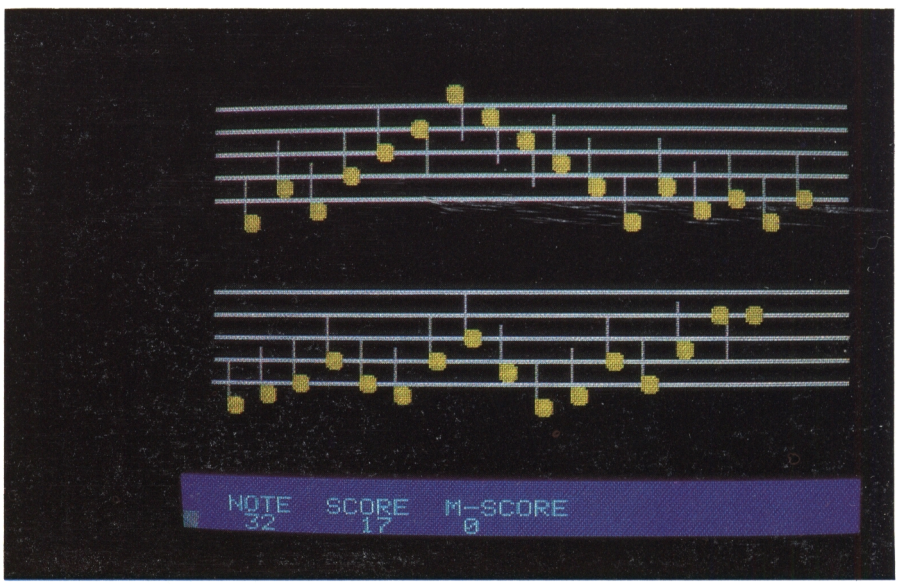
5. *Mur de briques*



6. *Embarquement immédiat*



7. *Envahisseurs*



8. *Dictée musicale*

e) Le sous-programme d'initialisation d'une partie

8010 initialise les compteurs "temps" et "kilomètres".

8020 fixe la position initiale de la voiture, son sens de déplacement et sa vitesse.

8030 dessine la voiture dans sa position initiale et fait entendre le bruit de son moteur.

8035-8050 attendent que vous tapiez sur une touche.

f) Les sous-programmes de changement de vitesse (5000-5030) et de sonorisation (6000-6030)

5010 calcule la nouvelle vitesse de la voiture.

5020 appelle le sous-programme de sonorisation.

6010 calcule la fréquence du son correspondant à la vitesse de la voiture.

6015-6020 émettent le bruit correspondant.

g) Le sous-programme de sortie de piste (3000-3090)

3010 fait retentir une explosion.

3020-3030 affichent la distance parcourue et le temps.

3040-3075 calculent le score correspondant, l'affichent et actualisent le nouveau score.

3080 efface la voiture.

5. Liste des variables

PA	couleur du fond.
IN	couleur d'encre utilisée pour les "bottes de paille" et pour la voiture.
MS	meilleur score.
BL$	chaîne de 37 "blancs" utilisée pour effacer des messages sur l'écran.

CP code ASCII du caractère représentant les bottes de paille délimitant la piste.

CV(4) tableau contenant les quatre codes ASCII des quatre caractères représentant la voiture dans chacune des quatre directions.

KM distance parcourue.

T durée du parcours.

X,Y coordonnées de la voiture.

DN code du déplacement à venir de la voiture (1 :←; 2 :→; 3 :↓; 4 :↑).

DL code de l'ancien déplacement de la voiture (voir DN).

V code de la vitesse de la voiture (1 : lent à 3 : rapide).

XN,YN coordonnées de la prochaine position de la voiture.

SC score.

F fréquence du son utilisé pour simuler le bruit du moteur.

Variables auxiliaires

C I CH$ R$

13

Tir aux ballons

1. Le jeu

Venez pour un moment vous distraire à la fête foraine en montrant vos talents de tireur. Une myriade de ballons multicolores s'élève devant vos yeux. Crevez-en le plus possible avec votre sarbacane.

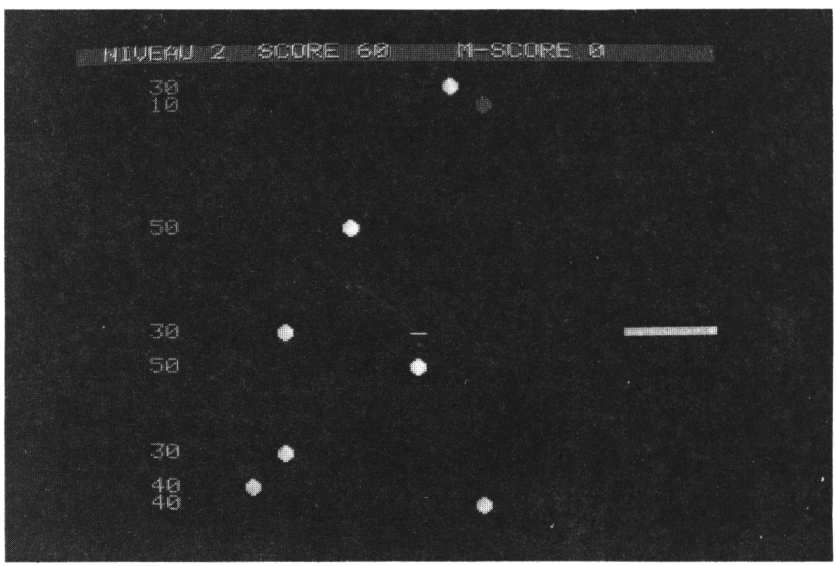
La sarbacane apparaît sur le côté droit de l'écran et se déplace verticalement à l'aide des deux touches " ↑ " et " ↓ ". Vous tirez une flèche en appuyant sur la barre d'espace. Tant qu'elle n'a pas traversé l'écran ou crevé un ballon, vous ne pouvez pas en tirer une nouvelle.

Trois niveaux de jeu vous sont proposés :

- *Niveau 1* : Tous les ballons rapportent le même nombre de points (10).

- *Niveau 2* : Le nombre de points dépend de la couleur du ballon : 10 pour un rouge, 20 pour un vert, ..., 60 pour un cyan (bleu ciel). Ce nombre apparaît sur la gauche du ballon.
- *Niveau 3* : Les ballons deviennent capricieux : lorsqu'ils sont touchés pour la première fois, ils se contentent de diminuer de taille. Il vous faut les toucher une seconde fois pour les faire disparaître et marquer les points correspondants.

La partie dure un temps déterminé à l'avance. Le programme affiche en permanence, en haut de l'écran, votre score et le meilleur score du niveau auquel vous jouez.



Vue de l'écran pendant le déroulement du jeu

2. *Le programme*

```
100 REM ********** TIR AUX BALLONS ********************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : FOR T=1 TO TM
150 :  GOSUB 3000: GOSUB 4000: GOSUB 3000: GOSUB 5000
160 :  GOSUB 3000: GOSUB 4000: GOSUB 3000
170 : NEXT T
180 : IF SC>MS(NV) THEN MS(NV)=SC
185 : WAIT 100
190 : CLS: PLOT 6,15,"voulez vous rejouer (O/N)"
200 : GET R$: GET R$
210 UNTIL R$<>"O"
220 END
2990 REM ---- SP DEPLACEMENT FLECHE --------------------
3000 IF XB=0 THEN RETURN
3010 IF SCRN(XB-2,YB)<>32 THEN 3100
3020 PLOT XB,YB,32: XB=XB-2:IF XB<X1 THEN XB=0 ELSE PLOT XB,YB,CC
3030 RETURN
3100 SHOOT: PLOT XB,YB,32
3110 IF SCRN(XB-2,YB)=C1 AND NV=3 THEN PLOT XB-2,YB,C2: XB=0:RETURN
3120 PLOT XB-2,YB,32:SC=SC+10*SCRN(XB-3,YB)
3130 CH$=STR$(SC):ID=17:GOSUB8500: XB=0: RETURN
3990 REM ---- SP DEPLACEMENT SARBACANE ET TIR ----------
4000 CL=PEEK(#208): IF CL=#38 THEN RETURN
4010 IF CL=#84 AND XB=0 THEN XB=XF-1: YB=YF: PLOT XB,YB,CB:RETURN
4020 PLOT XF,YF,CE$: YF=YF+(CL=#84)*(YF<26)-(CL=#9C)*(YF>0)
4030 PLOT XF,YF,CF$: RETURN
4040 RETURN
4990 REM ---- SP DEPLACEMENT BALLON -----------------
5000 IF RND(1)>FB THEN 5100
5010 X=H(K): IB=H(K+1): K=K+2: IF K>NXTHEN K=1
5030 IF NV=1 THEN PT=10 ELSE PT=10*IB
5040 PLOT 3,26,STR$(PT): CH$=CHR$(IB)+CHR$(C1)+CHR$(IN)
5050 PLOT X,26,CH$
5100 PLOT XF,YF,CE$: IF XB<>0 THEN PLOT XB,YB,32
5110 PRINT CHR$(10);: PLOT XF,YF,CF$: IF XB=0 THEN RETURN
5120 IF SCRN(XB,YB)<>32 THEN XB=0 ELSE PLOT XB,YB,CC
5130 RETURN
```

```
8000 REM ---- SP INITIALISATION D'UNE PARTIE ----------
8010 CLS: PLOT 10,10,"TIR AUX BALLONS"
8020 PLOT 10,15,"niveau choisi (1 a 3)?"
8030 GET R$: NV=VAL(R$): IF NV<1 OR NV>3 THEN 8030
8040 CH$="NIVEAU    SCORE       M-SCORE       ": ID=2: GOSUB 8500
8050 CH$=STR$(NV): ID=8: GOSUB 8500
8060 CH$=STR$(MS(NV)): ID=31: GOSUB 8500
8070 SC=0: XB=0: YF=15: K=1
8080 CLS: FOR I=1 TO 26: PRINT: NEXT I
8090 RETURN
8490 REM ---- SP AFFICHAGE LIGNE SUPERIEURE ------------
8500 FOR I=1 TO LEN(CH$)
8520 : POKE AD+ID+I-1,ASC(MID$(CH$,I,1))
8530 NEXT I
8540 RETURN
9000 REM ---- SP INITIALISATIONS DU JEU ---------------
9010 X1=7: X2=25
9020 XF=33: LF=6: XB=0: FB=0.2: TM=300
9030 C1=94: C2=95: CF=96: CC=64
9040 DIM MS(3)
9050 IN=3: PA=0: IS=2: PS=1: INK IN: PAPER PA
9070 AD=#BB80
9080 CF$="": FOR I=1 TO LF: CF$=CF$+CHR$(CF): NEXT I
9090 CE$="": FOR I=1 TO LF: CE$=CE$+" ": NEXT I
9100 DATA 12,30,63,63,63,63,30,12
9110 C=C1: GOSUB 9500
9120 DATA 0,0,12,30,30,12,0,0
9130 C=C2: GOSUB 9500
9140 DATA 0,0,63,63,63,63,0,0
9150 C=CF: GOSUB 9500
9160 DATA 0,0,0,0,63,0,0,0
9170 C=CC: GOSUB 9500
9180 POKE AD,PS+16: POKE AD+1,IS
9190 POKE #26A,10
9200 NX=100: DIM H(NX): K=1
9210 H(K)=X1+2*INT(RND(1)*(X2-X1)/2)
9220 IB=INT(RND(1)*6): IF IB=PA THEN 9220
9230 H(K+1)=IB: K=K+2: IF K<NX THEN 9210
9240 RETURN
9500 REM ---- SP INITIALISATION CARAC GRAPHIQUES ------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier la couleur de la sarbacane et de la flèche

Remplacez en 9050, la valeur 3 par un nombre de votre choix (entre 1 et 7).

2. Pour modifier la durée d'une partie

Remplacez en 9020, la valeur 300 par un nombre de votre choix (600 donnera des parties deux fois plus longues, 100 donnera des parties trois fois plus courtes).

3. Pour que les ballons soient plus ou moins nombreux

Remplacez, en 9020, la valeur 0.2 par un nombre compris entre 0 et 1 sachant que :

0.2 correspond, en moyenne, à un ballon toutes les cinq lignes.

0.5 à un ballon toutes les deux lignes.

1 à un ballon sur chaque ligne.

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Scroll.
— Configuration clavier/écran.
— Écriture directe en mémoire d'écran.

b) Le programme principal (100-220)

110 appelle le sous-programme d'initialisation du jeu.

120 et 210 répètent les instructions 130 à 200 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

130 appelle le sous-programme d'initialisation d'une partie.

140 et 170 répètent 300 fois les instructions 150 et 160 qui appellent les sous-programmes de déplacement de flèche (3000), de déplacement de l'ensemble des ballons (5000) et de déplacement de la sarbacane (4000).

180-200 actualisent le meilleur score et, après un court instant, vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9550)

9010 détermine le domaine dans lequel évoluent les ballons.

9020 fixe la position initiale de la sarbacane, sa dimension, la fréquence d'apparition d'un ballon et la durée d'une partie.

9030 fixe les codes des caractères représentant les ballons, la sarbacane et la flèche.

9050 fixe la couleur de fond et celle de la sarbacane et de la flèche.

9070 fixe l'adresse de début de la ligne supérieure.

9080-9170 fabriquent les caractères graphiques servant à représenter les ballons , la sarbacane et la flèche.

9180 fixe les couleurs de fond et d'encre de la ligne supérieure.

9190 supprime le bip du clavier et le curseur de l'écran.

9200-9230 préparent un tableau de valeurs tirées au hasard qui serviront à déterminer la position et la couleur de chaque ballon.

d) Le sous-programme d'initialisation d'une partie (8000-8090)

8010-8060 font choisir le niveau, affichent les textes de la ligne supérieure, le niveau et le meilleur score correspondant.

8070 initialise le score, la position de la sarbacane.

8080 initialise le défilement d'écran en plaçant le curseur en bas.

e) Le sous-programme de déplacement de la flèche (2990-3130)

3000 examine si une flèche a été tirée (dans ce cas, XB contient une valeur positive). Si c'est le cas :

3010 examine si la flèche peut être déplacée sans rencontrer un ballon.

3020-3030 déplacent la flèche lorsque cela est possible.

3100-3130 correspondent au cas où un ballon est touché :

3100 fait entendre le bruit du ballon qui crève et efface la flèche de son ancienne position.

3110 examine si l'on est en niveau 3. Dans ce cas, si le ballon est de taille normale, il est remplacé par un plus petit.

3120-3130 effacent le ballon touché, incrémentent le score et affichent sa nouvelle valeur.

f) Le sous-programme de déplacement des ballons (4990-5130)

5000 décide de l'apparition éventuelle d'un nouveau ballon. Le cas échéant :

5010-5050 dessinent le nouveau ballon et affichent le nombre de points correspondants.

5100-5120 effacent la sarbacane et la flèche (si elle existe) avant d'effectuer un défilement de l'écran vers le haut (scroll), redessinent la sarbacane et, éventuellement, la flèche.

g) Le sous-programme de déplacement de sarbacane et de tir (3990-4110)

4000 examine le clavier.

4010 place une flèche en position initiale si la barre d'espace a été pressée et si aucun tir n'est en cours.

4020-4030 déplacent la sarbacane s'il y a lieu.

h) Le sous-programme d'affichage en ligne supérieure (8490-8540)

Il affiche en colonne de numéro ID, le contenu de la chaîne CH$.

5. *Liste des variables*

X1,X2	coordonnées de l'intervalle dans lequel pourront se trouver les ballons (ces valeurs doivent impérativement être impaires)
XF,YF	coordonnées de l'extrémité gauche de la sarbacane
LF	longueur de la sarbacane
XB,YB	coordonnées de la flèche. (Si XB = 0, aucune flèche n'est présente sur l'écran)
TM	durée d'une partie (exprimée en nombre de tours de boucle)
FB	probabilité d'apparition d'un ballon sur une nouvelle ligne
C1	code ASCII du caractère représentant un ballon de taille normale
C2	code ASCII du caractère représentant un ballon de taille réduite
CF	code ASCII du caractère représentant la sarbacane
CC	code ASCII du caractère représentant la flèche
MS(3)	tableau contenant le meilleur score de chaque niveau
IN	couleur d'encre employée pour la sarbacane et la flèche
PA	couleur de fond
IS	couleur d'encre employée pour la ligne supérieure
PS	couleur de fond de la ligne supérieure
AD	adresse mémoire relative au début de la ligne supérieure
CF$	chaîne représentant la sarbacane
CE$	chaîne de caractères blancs servant à effacer la sarbacane
NX	nombre de valeurs qui seront tirées au hasard avant le début du jeu
K	pointeur dans le tableau H
NV	niveau de jeu
SC	score

Variables auxiliaires

C IB R$ CL X IB I

14

Mur de briques

1. Le jeu

C'est là un jeu très classique que vous avez certainement déjà rencontré. A l'intérieur d'un cadre (ici vert) se trouve un mur de briques (ici violettes). Une balle se promène sur l'écran rebondissant sur le cadre ou sur une "raquette mobile". Elle rebondit également sur les briques mais en les détruisant du même coup.

Par rapport aux versions habituelles de ce jeu, nous avons introduit une nouveauté : la raquette peut se déplacer à la fois horizontalement et verticalement (dans certaines limites).

L'objectif est de détruire le maximum de briques dans un minimum de temps. Pouvoir rapprocher la raquette du mur vous fera donc gagner du temps et par suite améliorer votre score.

Chaque balle ratée par la raquette est perdue. Vous disposez de six balles en tout. La raquette se déplace à l'aide des quatre touches fléchées ou, le cas échéant, de la poignée de jeu.

Le programme affiche en permanence le nombre de balles restantes et le nombre de briques détruites. La partie s'achève, soit quand vous n'avez plus de balles, soit (ce qui est plus rare) quand le mur a été entièrement détruit. Le programme vous fait alors connaître votre score qui est calculé en fonction du nombre de briques détruites et du temps mis pour y parvenir. En même temps, vous pourrez comparer votre résultat avec le meilleur score déjà réalisé. (Voir photo 5 hors texte : *Vue de l'écran pendant le déroulement du jeu*).

2. *Le programme*

REMARQUE: L'instruction 4000 occupe deux lignes de listes. Le nombre 4500 qui apparaît en dessous du numéro de ligne 4000 fait bien partie de la ligne 4000.

```
100 REM ********** CASSE BRIQUES **********
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8500: GOSUB 8000: GOSUB 7000
140 : REPEAT
150 :  GOSUB 4000: GOSUB 5000: T=T+1
160 : UNTIL ND>=NB OR NC<=0
170 : GOSUB 8700
180 UNTIL R$<>"O"
190 END
```

```
2990 REM ---- SP SONORISATION REBOND ---------------
3000 F=100: IF CX=CM THEN F=500
3010 IF CX=CR THEN F=200
3020 PLAY 1,0,0,0: SOUND 1,F,6: WAIT 10
3030 PLAY 0,0,0,0: RETURN
3990 REM ---- SP DEPLACEMENT BALLE ------------------
4000 IF SCRN(X,Y+DY)<>32ORSCRN(X+DX,Y)<>32ORSCRN(X+DX,Y+DY)<>32THEN
4050
4010 PLOT X,Y,32: X=X+DX: Y=Y+DY
4020 IF Y<YR THEN PLOT X,Y,CB: RETURN
4030 NC=NC-1: PLOT 11,0,STR$(NC): PLOT 11,0," "
4040 ZAP: WAIT 100: GOSUB 7000: RETURN
4050 C1=SCRN(X,Y+DY): C2=SCRN(X+DX,Y): C3=SCRN(X+DX,Y+DY)
4060 IF C1=32 THEN 4100
4070 IF C1=CM THEN PLOT X,Y+DY,32: GOSUB 6000
4080 CX=C1: GOSUB 3000: DY=-DY
4090 RETURN
4100 IF C2=32 THEN 4140
4110 IF C2=CM THEN PLOT X+DX,Y,32: GOSUB 6000
4120 CX=C2: GOSUB 3000: DX=-DX
4130 RETURN
4140 IF C3=CM THEN PLOT X+DX,Y+DY,32: GOSUB 6000
4150 CX=C3: GOSUB 3000: DX=-DX: DY=-DY
4160 IF RND(1)>0.75 THEN DX=-DX
4170 RETURN
4990 REM ---- SP DEPLACEMENT RAQUETTE ---------------
5000 CL=PEEK(#208)
5010 IF CL=#38 THEN RETURN
5020 PLOT XR,YR,CE$
5030 XR=XR+(CL=#BC)*(XR<XM)-(CL=#AC)*(XR>X1)
5040 YR=YR+(CL=#B4)*(YR<R2)+(CL=#9C)*(YR>R1)*(YR<>Y+1)
5050 PLOT XR,YR,CR$: RETURN
5990 REM ---- SP BRIQUE DETRUITE -----------------
6000 ND=ND+1
6010 PLOT 25,0,STR$(ND): PLOT 25,0," "
6020 RETURN
7000 REM ---- SP PLACEMENT HASARD BALLE ------------
7005 IF NC<=0 THEN RETURN
7010 X=X1+INT(RND(1)*(X2-X1+1))
7020 Y=Y4+2
7030 PLOT X,Y,CB
7040 DX=1: IF RND(1)>0.5 THEN DX=-1
7050 DY=1: IF RND(1)>0.5 THEN DY=-1
7060 PLOT XR,YR,CE$
7070 XR=19-INT(LR/2): YR=R2
7080 PLOT XR,YR,CR$
7090 RETURN
```

```
8000 REM ---- SP DESSIN CADRE ET MUR ----------------
8010 FOR X=X1-1 TO X2+1
8020 : PLOT X,Y1-1,CT: PLOT X,Y2+1,CT
8030 NEXT X
8040 FOR Y=Y1 TO Y2
8050 : PLOT X1-1,Y,CT: PLOT X2+1,Y,CT
8060 NEXT Y
8070 FOR X=X1 TO X2
8080 : FOR Y=Y3 TO Y4
8090 :  PLOT X,Y,CM
8100 : NEXT Y
8110 NEXT X
8120 RETURN
8500 REM ---- SP INITIALISATION PARTIE --------------
8510 CLS: T=0: ND=0: NC=NX
8520 PLOT 4,0,"BALLES      BRIQUES"
8530 PLOT 11,0,STR$(NC): PLOT 11,0," "
8540 RETURN
8700 REM ---- SP FIN DE PARTIE ----------------------
8720 SC=100*ND*ND/T*NX/(NX-NC)
8725 SC=INT(SC)
8730 PLOT 5,12,"SCORE": PLOT 12,12,STR$(SC): PLOT 12,12," "
8740 PLOT 5,14,"M-SCORE": PLOT 15,14,STR$(MS): PLOT 15,14," "
8750 IF SC>MS THEN MS=SC
8760 PLOT 5,17,"Voulez vous rejouer (O/N)"
8770 GET R$: GET R$
8780 RETURN
9000 REM ---- SP INITIALISATIONS DU JEU -------------
9010 MS=0: NX=6
9020 POKE #26A,10
9030 PA=0: IN=2: CLS: PAPER PA: INK IN
9040 X1=2: X2=37
9050 Y1=3: Y2=25: Y3=5: Y4=9
9060 R1=14: R2=23: LR=9
9070 XM=X2-LR+1
9080 NB=(X2-X1+1)*(Y4-Y3+1)
9085 XR=19-INT(LR/2): YR=R2
9090 CT=64: CM=95+128: CB=96: CR=160
9100 CE$=""
9110 FOR I=1 TO LR: CE$=CE$+" ": NEXT I
9120 CR$=""
9130 FOR I=1 TO LR: CR$=CR$+CHR$(CR): NEXT I
9140 DATA 63,63,41,33,33,41,63,63
9145 CX=CT: GOSUB 9500
9150 DATA 63,63,41,33,33,41,63,63
9160 CX=CM-128: GOSUB 9500
9170 DATA 0,30,63,63,63,63,30,0
9180 CX=CB: GOSUB 9500
9190 RETURN
```

```
9500 REM ---- SP INIT CARAC GRAPHIQUES ------------
9510 AM=46080+8*CX
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier le nombre de balles d'une partie

Remplacez, en 9010, le nombre 6 par la valeur de votre choix.

2. Pour modifier la taille ou la position du mur

Il vous suffit de savoir que, en 9050, les valeurs de Y3 (ici 5) et de Y4 (ici 9) correspondent aux rangs des lignes d'écran entre lesquelles s'étend le mur.

Ainsi, par exemple :

```
9050 Y1=3: Y2=25: Y3=5: Y4=11
```

vous fournira un mur commençant toujours à la 5e ligne en partant du haut, mais s'étendant jusqu'à la 11e ligne.

De même :

```
9050 Y1=3: Y2=25: Y3=7: Y4=11
```

vous fournira un mur de même étendue que le mur d'origine mais situé plus bas sur l'écran.

Attention : la valeur de Y3 doit toujours être strictement supérieure à celle de Y1 ; celle de Y4 doit toujours être strictement inférieure à celle de R1 (définie en ligne 9060).

3. Pour modifier le domaine dans lequel évolue la raquette

Il vous suffit de savoir que les valeurs 14 et 23 de la ligne 9060 en fixent les limites verticales.

Par exemple :

```
9060 R1=19: R2=23: LR=9
```

vous conduira à un domaine plus restreint.

Si vous remplacez 9060 par :

```
9060 R1=23: R2=23: LR=19
```

vous obtiendrez un jeu où la raquette ne peut plus se déplacer verticalement (vous retrouvez là la version classique du "mur de briques").

4. Pour modifier la taille de la raquette

Remplacez, en 9060, le nombre 9 par la valeur de votre choix.

5. Pour que la raquette se déplace horizontalement deux fois plus vite

Remplacez 5030 par :

```
5030 XR=XR+2*(CL=#BC)*(XR<XM)-2*(CL=#AC)*(XR>X1)
```

6. Pour que la balle se déplace plus vite

Remplacez 150 par :

```
150 : GOSUB 4000: GOSUB 5000: GOSUB 4000: T=T+1
```

7. Pour utiliser une poignée de jeu

Chargez le programme MANETTE 1 comme décrit en annexe. Branchez la poignée du côté droit et remplacez les lignes 5000 à 5040 par :

```
5000 CL=PEEK(#400)
5010 IF CL=191 THEN RETURN
5020 PLOT XR,YR,CE$
5030 XR=XR+((CL AND 2)=0)*(XR<XM)-((CL AND 1)=0)*(XR>X1)
5040 YR=YR+((CL AND 8)=0)*(YR<R2)+((CL AND 16)=0)*(YR>R1)*(YR<>Y+1)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran
— Poignées de jeu (facultatif).

b) Le programme principal (100-190)

110 appelle le sous-programme d'initialisation du jeu.

120 et 180 répètent les instructions 130 à 170 (déroulement d'une partie) jusqu'à ce que vous disiez que vous ne voulez plus rejouer :

130 prépare une partie en appelant successivement les sous-programmes :
- initialisation d'une partie,
- dessin du cadre et du mur de briques,
- placement au hasard de la balle.

140-160 répètent l'instruction 150 jusqu'à ce que le temps imparti à la partie soit écoulé ou jusqu'à ce que le mur de briques soit entièrement détruit :

150 appelle le sous-programme de déplacement de la balle et le sous-programme de déplacement de la raquette et incrémente le compteur de temps.

170 appelle le sous-programme de fin de partie.

c) Le programme d'initialisation du jeu (9000-9190)

9010 initialise le meilleur score et fixe le nombre de balles prévues.

9020 supprime le bip du clavier et le curseur d'écran.

9030 fixe les couleurs d'encre (utilisée pour le cadre et la balle) et de fond.

9040-9070 fixent les coordonnées du cadre et du domaine dans lequel évolue la raquette ainsi que sa longueur.

9080 calcule le nombre total de briques formant le mur.

9085 détermine la position initiale de la raquette.

9090-9180 fabriquent les caractères graphiques (et les chaînes de caractères) représentant la balle, les briques du mur et la raquette.

d) Le sous-programme d'initialisation d'une partie (8500-8540)

Il initialise le compteur de temps, le nombre de briques détruites, le nombre de balles restant à jouer et il en affiche la valeur.

e) Le sous-programme de placement au hasard de la balle (7000-7090)

7005 vérifie qu'il y a encore au moins une balle à jouer.

7010-7030 placent la balle, de façon aléatoire.

7040-7050 déterminent la direction de la balle, de façon aléatoire.

7060-7080 effacent l'ancienne raquette et dessinent la nouvelle en position initiale.

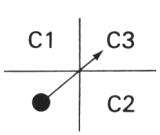
f) Le sous-programme de dessin du cadre et du mur (8000-8120)

8000-8060 dessinent le cadre.

8070-8110 dessinent le mur.

g) Le sous-programme de déplacement de la balle (3990-4170)

Pour chaque déplacement, il considère trois cases comme l'indique ce schéma :



(● désigne la balle et la flèche représente son sens de déplacement).

4000 teste le contenu des trois cases C1,C2 et C3.

4010-4040 cas où ces trois cases sont disponibles : on avance la balle en C3 en vérifiant que la raquette ne rate pas la balle. Dans ce dernier cas, le nombre de balles restantes est incrémenté et l'on dessine une nouvelle balle (sous-programme 7000).

4050-4170 examinent le contenu de ces cases, lorsqu'elles ne sont pas toutes les trois disponibles :

4070-4090 cas où C1 est occupée :

4070 appelle, lorsqu'il s'agit d'une brique, le sous-programme "brique détruite" qui incrémente le nombre de briques et qui l'affiche.

4080 assure, dans tous les cas (brique ou bord) le rebond de la balle et sa sonorisation.

4100-4130 cas où C2 est occupée (sans que C1 le soit). On procède comme ci-dessus ; seul le rebond a lieu différemment.

4140-4170 cas où seule C3 est occupée. On procède de même ; cependant 4160 donne au rebond un caractère quelque peu aléatoire (on est ici dans le cas où l'on touche une brique par un coin).

h) Le sous-programme de déplacement de la raquette (4990-5050)

5000-5010 lisent le clavier.

5020 efface la raquette de son ancienne position.

5030-5050 déterminent la nouvelle position de la raquette et la dessinent.

i) Le sous-programme de sonorisation de rebond (2990-3030)

Il émet un son dont la fréquence dépend de l'objet sur lequel la balle rebondit :

100 pour une brique,
500 pour le bord,
200 pour la raquette.

j) Le sous-programme de fin de partie (8700-8780)

8720-8740 calculent le score et l'affichent en même temps que le meilleur score.

8750 actualise le meilleur score.

8760-8770 vous demandent si vous souhaitez rejouer.

5. *Liste des variables*

MS	meilleur score
NX	nombre total de balles
PA	couleur du fond (ici noir)
IN	couleur de l'encre utilisée pour le cadre et la balle
X1,X2,Y1,Y2	coordonnées du cadre
Y3,Y4	limites (verticales) entre lesquelles s'étend le mur
R1,R2	limites (verticales) entre lesquelles peut se déplacer la raquette
LR	longueur de la raquette
XM	abscisse maximum de la raquette, compte-tenu de sa longueur
NB	nombre total de briques
CT	code ASCII du caractère représentant le cadre
CM	code ASCII du caractère représentant les briques
CB	code ASCII du caractère représentant la balle
CR	code ASCII du caractère représentant la raquette
CE$	chaîne de caractères blancs servant à l'effacement de la raquette
CR$	chaîne de caractères représentant la raquette
XR,YR	coordonnées de l'extrémité gauche de la raquette
T	compteur de temps
ND	nombre de briques détruites
NC	nombre de balles restant à jouer
X,Y	coordonnées de la balle (sauf en 8000)
DX,DY	sens de déplacement de la balle
C1,C2,C3	codes ASCII des caractères situés sur la trajectoire de la balle (voir schéma)
SC	score

Variables auxiliaires

I CX CL

X et Y (dans le sous-programme 8000)

F

15

Embarquement immédiat

1. Le jeu

Devenez pour un instant un extra-terrestre ! C'est ce que vous propose ce jeu qui vous place aux commandes d'une soucoupe volante.

Vos compatriotes ont voulu réaliser une exploration pacifique de la terre. Ils y ont déposé une colonie de robots, programmés pour recueillir le maximum d'informations. Mais les choses ont mal tourné ! Les terriens se sont emparés de vos robots et tentent d'en percer le secret.

Vous êtes chargés de les récupérer le plus vite possible à l'aide d'une soucoupe équipée d'un générateur d'ondes anti-gravitationnelles. Vous pouvez ainsi "aspirer" n'importe quel objet, simplement en le survolant. Facile direz-vous ! Certes. Encore, faut-il éviter de récupérer par maladresse des robots terriens qui voisinent avec les vôtres. En effet, ceux-ci risquent d'endommager plus ou moins gravement votre soucoupe, vous obligeant ainsi à effectuer des réparations coûteuses en temps.

Le domaine où sont parqués vos robots est entouré d'un intense champ gravitationnel. Le moindre survol vous ferait vous écraser au sol.

Enfin, votre soucoupe consomme énormément d'énergie ; vous ne disposez que d'un temps limité pour effectuer votre "embarquement". Si vous n'y parvenez pas dans le délai imparti, vous vous écraserez sur le sol terrestre.

Le programme vous propose quatre vitesses de jeu et trois niveaux :

- *Niveau 1* : Destiné à l'entraînement. Il ne comporte ni robots terriens, ni champ gravitationnel.
- *Niveau 2* : Il comporte des robots terriens mais pas de champ gravitationnel.
- *Niveau 3* : Il correspond à la description faite ci-dessus avec robots terriens et champ gravitationnel.

Vous voyez ensuite apparaître vos robots (en jaune), les robots terriens (en bleu auréolé de blanc) s'il y a lieu et votre soucoupe. Le tout est entouré d'une bordure bleue et blanche qui délimite le domaine de jeu et qui représente, au niveau 3, le champ antigravitationnel.

Vous pouvez examiner la situation à loisir avant de commencer en tapant sur une touche. Vous dirigez votre soucoupe à l'aide des quatre touches fléchées.

Le programme affiche en permanence le temps restant, le niveau de jeu, la vitesse et le meilleur score. Le survol d'un robot terrien est accompagné d'un bruit d'explosion et votre soucoupe est immobilisée un certain temps. Le survol d'un de vos robots se traduit par un "zap".

Si vous parvenez à atteindre votre objectif dans le temps imparti, vous entendrez votre soucoupe s'éloigner de plus en plus rapidement. Dans le cas contraire, vous l'entendrez tomber et s'écraser au sol. (Voir photo 6 hors-texte : *Vue de l'écran pendant le déroulement du jeu*).

2. *Le programme*

```
100 REM ********** EMBARQUEMENT IMMEDIAT ***************
110 GOSUB 9700
120 REPEAT
130 : GOSUB 9000: GOSUB 8000
140 : IF PEEK(768)<>176 THEN DL=PEEK(735)-135
150 :  XP=X+(DL=1)-(DL=2): YP=Y+(DL=4)-(DL=3)
160 :  T=T-1: PLOT 6,0,STR$(T)+"  ": PLOT 6,0," "
170 :  IF VI<4 THEN WAIT 3*(4-VI)
172 :  IF SCRN(XP,YP)<>32 THEN 250
175 :  PLOT X,Y,32: X=XP: Y=YP: PLOT X,Y,CS
180 :  IF T>0 AND NR>0 THEN 140
190 :  IF T<=0 THEN GOSUB 2000 ELSE GOSUB 3000
200 :  GOTO 300
250 :  C=SCRN(XP,YP)
260 :  IF C=CG AND NV=3 THEN GOSUB 2000: GOTO 300
265 :  IF C=CG THEN DL=0: GOTO 180
270 :  IF C=CH THEN GOSUB 1700: GOTO 180
280 :  IF C=CR THEN GOSUB 1500: GOTO 180
290 : GOTO 140
300 : IF T>MS(NV,VI) THEN MS(NV,VI)=T
310 : PLOT 3,26,12: PLOT 4,26,"voulez vous rejouer (O/N)"
320 : GET R$: GET R$
330 UNTIL R$<>"O"
340 END
1490 REM ----- SP ROBOT EMBARQUE ---------------------
1500 ZAP : NR=NR-1
1520 PLOT X,Y,32
1530 X=XP: Y=YP: PLOT X,Y,CS
1540 RETURN
1690 REM ---- SP ROBOT TERRIEN TOUCHE ----------------
1700 PING:WAIT8:SHOOT:WAIT8:SHOOT
1720 PLOT X,Y,32
1730 X=XP: Y=YP: PLOT X,Y,CS
1740 P=INT(RND(1)*60): T=T-P: IF T<1 THEN T=1
1745 WAIT 2*P
1750 RETURN
1990 REM ----- SP PERDU -----------------------------
2000 WAIT 30: T=0
2015 PLAY 1,0,0,0
2020 FOR I=40 TO 170
2030 : SOUND 1,I,7: WAIT 3
2040 NEXT  I
2050 PLAY 0,0,0,0
2060 EXPLODE: WAIT 40: EXPLODE: WAIT 25: EXPLODE
2070 RETURN
```

```
3000 REM ----- SP GAGNE ----------------------------
3008 WAIT 50
3020 PLAY 1,0,0,0
3030 FOR I=170 TO 40 STEP -1
3040 : SOUND 1,I,7: WAIT 3
3050 NEXT I
3060 PLAY 0,0,0,0: WAIT 50
3080 RETURN
8000 REM ---- SP INITIALISATION D'UNE PARTIE ----------
8010 T=800: NH=25: NR=15
8070 FOR X=X1 TO X2
8080 : PLOT X,Y1,CG: PLOT X,Y2,CG
8090 NEXT X
8100 FOR Y=Y1 TO Y2
8110 : PLOT X1,Y,CG: PLOT X2,Y,CG
8120 NEXT Y
8150 FOR I=1 TO NR
8160 : GOSUB 8500: PLOT X,Y,CR
8170 NEXT I
8175 IF NV<2 THEN 8210
8180 FOR I=1 TO NH
8190 : GOSUB 8500: PLOT X,Y,CH
8200 NEXT I
8210 GOSUB 8500: PLOT X,Y,CS
8212 DL=0
8215 PLOT 2,0,"tapez sur une touche pour commencer "
8220 GET R$: WAIT 20
8222 PLOT 0,0,2
8224 PLOT 1,0,"TEMPS      M-TEMPS      NIV   VIT   "
8226 PLOT 19,0,STR$(MS(NV,VI)): PLOT 19,0," "
8250 PLOT 28,0,STR$(NV): PLOT 34,0,STR$(VI)
8270 RETURN
8500 REM --- SP TIRAGE COORDONNEES D'UN POINT DISPONIBLE ----
8510 X=X1+2+INT(RND(1)*(X2-X1-3))
8520 Y=Y1+2+INT(RND(1)*(Y2-Y1-3))
8530 IF SCRN(X,Y)<>32 THEN 8510
8540 RETURN
9000 REM --- SP CHOIX DE L'UTILISATEUR ---------------
9010 CLS: PA=6: IN=4
9020 PAPER PA: INK IN
9030 PLOT 8,4,"EMBARQUEMENT IMMEDIAT"
9040 PLOT 5,8,"3 niveaux possibles"
9050 FOR I=1 TO 3
9060 : PLOT 3,9+I,19: PLOT 37,9+I,PA+16
9070 NEXT I
9080 FOR I=1 TO 3
9090 : PLOT 4,9+I,1: PLOT 5,9+I,CHR$(I+48): PLOT 6,9+I,IN
```

```
9100 NEXT I
9110 PLOT 7,10,"pas d'obstacles"
9120 PLOT 7,11,"obstacles (robots terriens)"
9130 PLOT 7,12,"obstacles et champ de gravite"
9170 PLOT 8,14,"niveau choisi?"
9180 R$=KEY$: IF R$="" THEN 9180
9190 NV=VAL(R$)
9200 IF NV>0 AND NV<4 THEN 9250
9210 : PLOT 22,14,"(1, 2 ou 3 svp)"
9220 : WAIT 100
9230 : PLOT 22,14,"                "
9240 : GOTO 9180
9250 PLOT 22,14,CHR$(48+NV)
9253 FOR I=1 TO 2
9255 : PLOT 3,18+I,18: PLOT 35,18+I,PA+16
9257 NEXT I
9260 PLOT 5,17,"4 vitesses possibles"
9270 PLOT 8,19,"de 1 (lent)"
9280 PLOT 8,20," a 4 (rapide)"
9290 PLOT 8,22,"vitesse choisie?"
9300 R$=KEY$: IF R$="" THEN 9300
9310 VI=VAL(R$)
9320 IF VI>0 AND VI<5 THEN 9370
9330 : PLOT 26,22,"(1 a 4 svp)"
9340 : WAIT 100
9350 : PLOT 26,22,"            "
9360 : GOTO 9300
9370 PLOT 26,22,CHR$(48+VI)
9380 WAIT 100
9390 CLS: PA=0: IN=3: PAPER PA: INK IN
9400 RETURN
9500 REM ----- SP INIT CARAC GRAPHIQUES ----------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
9700 REM ----- SP INITIALISATIONS DU JEU --------------
9720 X1=1: X2=38: Y1=1: Y2=26
9740 CG=64+128: CR=94: CS=96: CH=CR+128
9750 DATA 12,12,30,63,63,30,12,12
9760 C=CG-128: GOSUB 9500
9770 DATA 14,14,4,31,47,14,10,10
9780 C=CR: GOSUB 9500
9790 DATA 12,20,63,45,45,63,63,30
9800 C=CS: GOSUB 9500
9810 DIM MS(3,4): POKE #26A,10
9820 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour changer la couleur de fond

Remplacez, en 9390, la valeur 0 (dans PA = 0) par le nombre de votre choix. Notez que cette modification agit sur la couleur de la bordure et des robots terriens.

2. Pour changer la couleur de vos robots et de la soucoupe

Remplacez, en 9390, la valeur 3 (dans IN = 3) par la valeur de votre choix. Là encore, cette modification agit sur la couleur de la bordure et des robots terriens.

3. Pour changer la durée du jeu

Remplacez, en 8010, la valeur 800 par un nombre de votre choix.

4. Pour modifier le nombre de robots

Remplacez, en 8010, la valeur 15 (dans NR = 15) par le nombre de votre choix.

5. Pour modifier le nombre de robots terriens

Remplacez, en 8010, la valeur 25 (dans NH = 25) par le nombre de votre choix.

6. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1", comme décrit en annexe. Branchez la poignée droite et remplacez les lignes 140 et 150 par :

```
140 C=PEEK(#400): IF C <> 191 THEN DL=C
145 XP=X+((DL AND 1)=0)-((DL AND 2)=0)
150 YP=Y+((DL AND 16)=0)-((DL AND 8)=0)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.
— Poignées de jeu (facultatif).

b) Le programme principal (100-340)

110 appelle le sous-programme d'initialisation du jeu.

120 et 290 répètent les instructions 130 à 280 (déroulement d'une partie) jusqu'à ce que vous ne souhaitiez plus rejouer :

130 appelle le sous-programme de choix du niveau et de la vitesse de jeu ainsi que le sous-programme d'initialisation de la partie.

140-290 sont répétées jusqu'à la fin de la partie (temps dépassé ou champ anti-gravitionnel ou robots embarqués) :

140 examine le clavier et, si une touche est enfoncée, actualise la direction de déplacement de la soucoupe.

150 calcule la nouvelle position de la soucoupe.

160 actualise le temps.

170 réalise une attente dépendant de la vitesse de jeu choisie.

172 examine le prochain emplacement.

175-190 cas où il est disponible : on s'assure que la partie n'est pas terminée. Dans ce cas, on appelle l'un des sous-programmes de fin de partie (perdu ou gagné).

250-280 cas où il est occupé :

260 appelle le sous-programme de fin de partie s'il s'agit d'un champ anti-gravité et si l'on est en niveau 3.

265 immobilise la soucoupe s'il s'agit d'un champ anti-gravité et si l'on est dans un niveau différent de 3.

270 appelle, s'il y a lieu, le sous-programme "robot terrien touché".

280 appelle, s'il y a lieu, le sous-programme "robot embarqué".

300-320 actualisent le meilleur score et vous demandent si vous voulez rejouer.

c) Le sous-programme de choix du niveau de jeu (9000-9400)

9010-9020 fixent les couleurs d'encre et de fond employées pendant la durée du choix.

9030-9240 font choisir le niveau de jeu, en rejetant les réponses autres que 1, 2 ou 3.

9250-9380 font choisir la vitesse de jeu, en rejetant les réponses autres que 1, 2, 3 ou 4.

9290 fixe les couleurs d'encre et de fond employées pendant le déroulement de la partie.

d) Le sous-programme d'initialisation du jeu (9700-9820)

9720 fixe les limites du domaine de jeu.

9740-9800 fabriquent les caractères graphiques représentant le champ gravitationnel, la soucoupe et les robots.

9810 réserve le tableau des meilleurs scores et supprime le bip du clavier et le curseur d'écran.

e) Le sous-programme d'initialisation d'une partie (8000-8270)

8010 initialise le compteur de temps, le nombre de robots terriens et le nombre de robots extra-terrestres.

8070-8120 trace le contour d'un rectangle représentant la zone où règne un champ gravitationnel.

8150-8170 dessinent les robots extra-terrestres. Le sous-programme 8500 est utilisé pour tirer les coordonnées d'un emplacement disponible.

8175-8200 dessinent, s'il y a lieu (niveau de jeu au moins égal à 2), les robots terriens.

8210-8212 dessinent la soucoupe dans sa position de départ et initialisent sa direction de déplacement.

8215-8220 attendent que vous pressiez une touche.

8222-8250 affichent le meilleur score, le niveau et la vitesse de jeu.

f) Le sous-programme "robot embarqué" (1490-1540)

1500 émet un zap et actualise le nombre de robots restant à embarquer.

1520 efface le robot.

1530 déplace la soucoupe.

g) Le sous-programme "robot terrien touché" (1690-1745)

1700 émet des sons appropriés.

1720 efface le robot.

1730 déplace la soucoupe.

1740-1745 attendent pendant une durée tirée au hasard et actualisent le compteur de temps.

h) Le sous-programme "perdu" (1990-2070)

Il fait entendre le bruit de la soucoupe qui perd de l'altitude et qui explose au sol.

i) Le sous-programme "gagné" (3000-3080)

Il fait entendre le bruit de la soucoupe qui s'éloigne de plus en plus vite.

5. Liste des variables

X1,X2,Y1,Y2	coordonnées du domaine de jeu
CG	code ASCII du caractère graphique représentant la limite du domaine (champ anti-gravité)
CR	code ASCII du caractère graphique représentant les robots extra-terrestres
CS	code ASCII du caractère graphique représentant la soucoupe
CH	code ASCII du caractère graphique représentant les robots terriens

PA	couleur de fond
IN	couleur d'encre
NV	niveau de jeu
VI	vitesse de jeu
T	compteur de temps
NH	nombre de robots terriens
NR	nombre de robots extra-terrestres
DL	direction de déplacement de la soucoupe (1 à 4)
MS(3,4)	tableau des meilleurs scores. MS(i,j) représente le meilleur score relatif au niveau i et à la vitesse j
X,Y	coordonnées de la soucoupe
XP,YP	coordonnées du prochain emplacement de la soucoupe

Variables auxiliaires

C AM I XX R$

X et Y (dans les sous-programmes 8500 et 9000)

16
Aliénation

1. Le jeu

Ça y est ! Les aliens, ces êtres venus d'ailleurs, ont envahi la terre. La race humaine a presque entièrement disparu de la planète. Vous êtes l'un des rares survivants. Pour combien de temps ! De nombreux aliens ne cessent de vous poursuivre pour vous détruire. Or, le moindre contact avec l'une de ces créatures serait mortel.

Pour comble de tout, vous ne disposez d'aucune arme pour vous défendre. Toutefois, ça et là gisent des robots, à forme humaine, cloués sur place par une privation soudaine de l'énergie nécessaire à leur fonctionnement. Que puis-je attendre d'eux me direz-vous ?

En fait, chacun de ces robots va pouvoir vous rendre un ultime service en faisant disparaître un alien (et un seul). En effet, tout contact entre un alien et un objet métallique conduit à la désintégration complète, et de l'objet et de l'alien. D'autre part, les aliens sont des êtres un peu stupides et aux réflexes lents (c'est leur nombre seul qui fait leur force).

Ils se déplacent par bonds irréguliers et, de façon assez grossière, dans votre direction. Il vous suffit donc de les obliger à bondir sur un robot pour les faire disparaître.

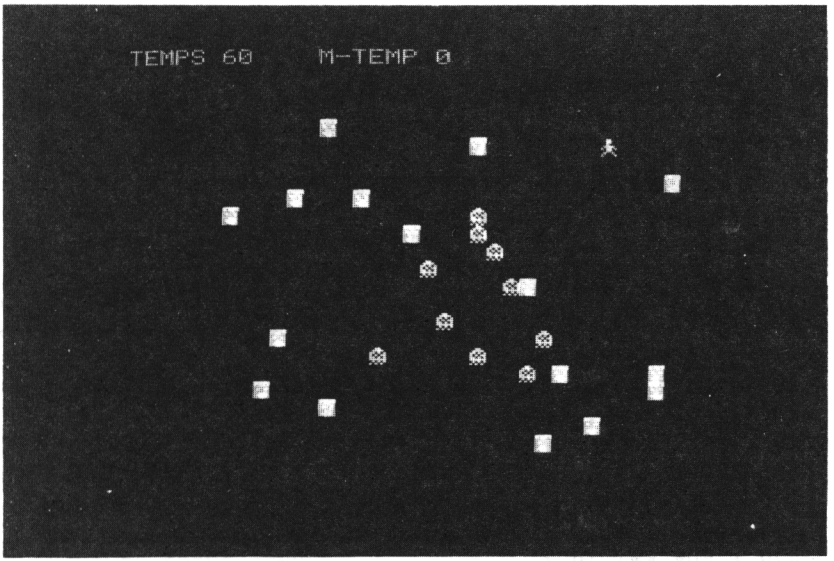
Comme vous le constaterez, les aliens font de grands bonds lorsqu'ils sont loin de vous. Par contre, une sorte de méfiance instinctive les amène à avancer plus lentement lorsqu'ils sont plus proches de vous. D'autre part, leur sens de l'orientation est peu évolué puisqu'ils peuvent être tout près de vous et faire un bond sur... "un emplacement voisin".

Enfin, sachez que vous êtes la seule proie qui intéresse les aliens. En conséquence, ils ne voient aucunement les robots et leurs déplacements ne sont nullement affectés par leur présence.

En début de partie vous voyez apparaître la disposition des aliens (rouges), des robots (bleus) et du personnage qui vous représente (en rouge). Vous pouvez prendre le temps d'étudier la situation avant d'appuyer sur une touche quelconque. La chasse infernale s'engage alors. Vous vous déplacez grâce aux quatre touches fléchées? Ne craignez pas de percuter un robot. Cela ne vous fera aucun mal.

Chaque désintégration (robot + alien) vous est signalée par un bruit sec et un éclair. Si vous êtes atteint par un alien (ou si, maladroitement, vous en percutez un) votre disparition est accompagnée de sons et d'effets lumineux appropriés.

Le programme affiche en permanence le temps qui s'écoule et le meilleur temps réalisé jusqu'ici.



Vue de l'écran pendant le déroulement d'une partie

2. *Le programme*

```
100 REM ********** ALIENATION **************************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : FOR I=1 TO NA
150 :  T=T+1: PLOT 7,0,STR$(T)
160 :  GOSUB 4000: GOSUB 5000: IF FI=1 THEN 190
170 : NEXT
180 : GOTO 140
190 : IF CT=0 AND(T<MT OR MT=0) THEN MT=T
200 : WAIT 50: PLOT 26,0,12: PLOT 27,0,"rejouez vous"
210 : GET R$: GET R$
220 UNTIL R$<>"O"
230 END
```

```
3990 REM ----- SP DEPLACEMENT HUMANOIDE ---------------
4000 CL=PEEK(#208): IF CL=#38 THEN RETURN
4010 XN=XB+(CL=#AC)-(CL=#BC): YN=YB+(CL=#9C)-(CL=#B4)
4020 IF XN>X2-2 OR XN<X1+2 OR YN>Y2-2 OR YN<Y1+2 THEN RETURN
4030 IF SCRN(XN,YN)<>32 AND SCRN(XN,YN)<>CB THEN 4050
4040 PLOT XB,YB,32: XB=XN: YB=YN: PLOT XB,YB,CB: RETURN
4050 IF SCRN(XN,YN)=CO THEN RETURN
4060 GOSUB 6000: RETURN
4990 REM ----- SP DEPLACEMENT ALIEN NUMERO I ----------
5000 X=XA(I): Y=YA(I): IF X=0 THEN RETURN
5010 XN=X+(RND(1)+0.5)*((X>XB)-(X<XB)+2*((X>XB+4)-(X<XB-4)))
5020 YN=Y+(RND(1)+0.5)*((Y>YB)-(Y<YB)+2*((Y>YB+4)-(Y<YB-4)))
5030 IF SCRN(XN,YN)<>32 THEN 5050
5040 PLOT X,Y,32: PLOT XN,YN,CA: XA(I)=XN: YA(I)=YN: RETURN
5050 IF SCRN(XN,YN)=CA THEN RETURN
5060 IF SCRN(XN,YN)=CO THEN GOSUB 7000 ELSE GOSUB 6000
5080 RETURN
6000 REM ----- SP MORT HUMANOIDE ----------------------
6010 FOR L=1 TO 7
6020 : FOR K=1 TO 7
6030 :  INK K: PAPER L
6040 : NEXT K
6050 : ZAP: WAIT 2
6060 NEXT L
6070 INK IN: PAPER PA: POKE 48040,16: PLOT 1,0,2
6075 PLOT XB,YB,32: PLOT XN,YN,CA
6080 EXPLODE: WAIT 60: EXPLODE
6090 FI=1: RETURN
6990 REM ----- SP MORT ALIEN --------------------------
7000 INK 5: PLOT X,Y,32: XA(I)=0
7010 PLOT XN,YN,32: SHOOT: WAIT 5: INK IN
7020 CT=CT-1: IF CT<=0 THEN FI=1
7030 RETURN
8000 REM ----- SP INITIALISATION D'UNE PARTIE ---------
8010 CT=NA: CLS: FI=0: T=0
8020 POKE 48040,16: PLOT 1,0,2
8030 FOR I=1 TO NO
8040 : X=X1+3+INT(RND(1)*(X2-X1-5))
8050 : Y=Y1+3+INT(RND(1)*(Y2-Y1-5))
8060 : IF SCRN(X,Y)<>32 THEN 8040
8065 : PLOT X,Y,CO
8070 NEXT I
8080 FOR I=1 TO NA
8090 : X=X1+INT(RND(1)*(X2-X1+1))
8100 : Y=Y1+INT(RND(1)*(Y2-Y1+1))
```

```
8110 : IF SCRN(X,Y)<>32 THEN 8090
8115 : XA(I)=X: YA(I)=Y: PLOT X,Y,CA
8120 NEXT I
8130 XB=X1+INT(RND(1)*(X2-X1-4))+2
8140 YB=Y1+INT(RND(1)*(Y2-Y1-4))+2
8150 IF SCRN(XB,YB)<>32 THEN 8130
8160 PLOT XB,YB,CB
8180 PLOT 2,0,"pour commencer, tapez une touche": GET R$
8190 PLOT 2,0,"TEMPS        M-TEMPS"
8200 PLOT 20,0,STR$(MT)+"              "
8210 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU ---------------
9010 CLS: IN=1: PA=3: INK IN: PAPER PA
9020 MT=0: FI=0: NO=18: NA=12
9030 DIM XA(NA),YA(NA)
9040 X1=1: X2=38: Y1=1: Y2=26
9050 CO=64+128: CA=95: CB=96
9060 DATA 12,30,53,43,53,63,21,42
9070 C=CA: GOSUB 9500
9080 DATA 12,12,8,30,45,12,18,33
9090 C=CB: GOSUB 9500
9100 DATA 0,18,63,45,63,51,33,51
9110 C=CO-128: GOSUB 9500
9140 POKE #26A,10
9150 RETURN
9500 REM ----- SP INIT CARAC GRAPHIQUES ----------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. Si vous souhaitez personnaliser ce jeu

1. Pour modifier le nombre de robots

Remplacez, en 9020, le nombre 18 par la valeur de votre choix.

2. Pour modifier le nombre d'aliens

Remplacez en 9020, le nombre 12 par la valeur de votre choix.

REMARQUE: Ne placez jamais plus d'aliens que de robots, car il serait alors impossible de gagner.

3. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1" comme indiqué en annexe. Branchez la poignée du côté droit puis remplacez les lignes 4000 et 4010 par:

```
4000 CL=PEEK(#400): IF CL=191 THEN RETURN
4010 XN=XB+((CL AND 1)=0)-((CL AND 2)=0)
4015 YN=YB+((CL AND 16)=0)-((CL AND 8)=0)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.
— Poignées de jeu (facultatif).
— Écriture directe en mémoire d'écran.

b) Le programme principal (100-230)

110 appelle le sous-programme d'initialisation du jeu.

120 et 220 répètent les instructions 130 à 220 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer.

130 appelle le sous-programme d'initialisation d'une partie.

140 à 180 sont répétées jusqu'à ce que la partie soit finie (FI = 1),

soit par disparition de tous les aliens, soit par votre propre disparition :

140 et 170 répètent, pour chaque alien, les instructions 150 et 160. Celles-ci incrémentent le temps, déplacent l'humanoïde qui vous représente et un alien.

190-200 actualisent le meilleur score et vous demandent si vous désirez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9150)

9010 fixe les couleurs de fond et d'encre.

9020 initialise le meilleur score et fixe le nombre de robots et d'aliens.

9030 réserve les deux tableaux contenant les coordonnées des aliens.

9040 fixe les limites du domaine de jeu.

9050-9110 fabriquent les caractères graphiques représentant les robots, les aliens et l'humanoïde.

9140 supprime le bip du clavier et le curseur d'écran.

d) Le sous-programme d'initialisation d'une partie (8000-8210)

8010 initialise le nombre d'aliens survivants et le compteur de temps ; efface l'écran.

8020 fixe les couleurs (fond et encre) de la ligne du haut de l'écran.

8030-8070 dessinent les robots dans leur position initiale.

8080-8120 dessinent les aliens dans leur position initiale et placent leurs coordonnées dans XA et YA.

8130-8160 déterminent les coordonnées initiales de l'humanoïde et le dessinent.

8180 attend que vous tapiez une touche.

8190-8200 affichent le meilleur temps.

e) Le sous-programme de déplacement de l'humanoïde (3990-4060)

4000 examine le clavier.

4010-4020 calculent la nouvelle position de l'humanoïde et vérifient qu'elle reste dans le domaine de jeu.

4030 examine le contenu de cette nouvelle position:

4040 déplace l'humanoïde si la position est libre.

4060 appelle le sous-programme "mort humanoïde" si la position est occupée par un alien.

f) Le sous-programme de déplacement d'un alien

Il déplace l'alien dont le numéro est contenu dans la variable I.

5000 vérifie que l'alien concerné est encore vivant.

5010-5020 calculent la nouvelle position de l'alien.

5030-5040 vérifient que cette position est disponible et, dans ce cas, déplacent l'alien.

5050 vérifie que cette position n'est pas occupée par un autre alien.

5060 appelle le sous-programme "mort humanoïde" si cette position est occupée par l'humanoïde et le sous-programme "mort alien" si elle est occupée par un robot.

g) Le sous-programme "mort d'un alien" (6990-7030)

7000 change temporairement la couleur d'encre (pour simuler un éclair) et efface l'alien.

7010 efface le robot, émet un bruit de fusil et rétablit la couleur d'encre initiale.

7020 décrémente le compteur de survivants. Met à un l'indicateur de fin de partie si tous les aliens sont morts.

h) Le sous-programme "mort de l'humanoïde" (6000-6090)

6010-6060 créent des effets sonores et lumineux en modifiant les couleurs d'encre et de fond et en faisant entendre des "zap".

6070 rétablit les couleurs d'origine.

6075-6080 effacent l'humanoïde et placent à un l'indicateur de fin de partie.

5. *Liste des variables*

PA	couleur de fond utilisée pour le domaine de jeu
IN	couleur d'encre utilisée pour l'humanoïde et les aliens; les robots sont en "vidéo inverse"
MT	meilleur temps
FI	indicateur de fin de jeu FI = 0 cas usuel FI = 1 fin de jeu détectée
NO	nombre total de robots
NA	nombre total d'aliens
XA(NA),YA(NA)	tableaux contenant les coordonnées des aliens. (Quand un alien disparaît, son abscisse passe à zéro)
X1,X2,Y1,Y2	coordonnées du domaine de jeu
CO	code ASCII du caractère représentant les robots
CA	code ASCII du caractère représentant les aliens
CB	code ASCII du caractère représentant l'humanoïde
CT	nombre d'aliens encore en vie
T	temps écoulé
XB,YB	coordonnées de l'humanoïde
XN,YN	nouvelles coordonnées de l'humanoïde ou d'un alien
I	numéro de l'alien en cours de déplacement

Variables auxiliaires

I (dans le sous-programme 8000 seulement)
L K X Y CL

17

Amanites

1. Le jeu

C'est l'automne. Dans les sous-bois et les prairies, le long des routes et des chemins, des champignons, pressés de vivre, apparaissent plus beaux et plus tentants les uns que les autres.

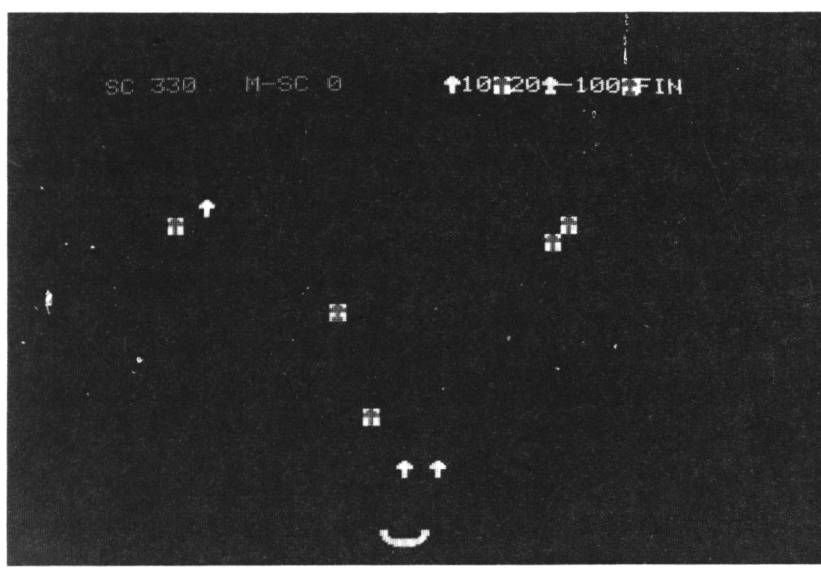
Soyez donc un fin "mycologue". Remplissez votre panier de champignons comestibles et évitez ceux qui sont dangereux et encore plus ceux qui sont mortels.

Vous déplacez votre "panier", le long du bas de l'écran à l'aide des touches "→" et "←". Les touches "↓" et "↑" le déplacent également horizontalement mais à une vitesse triple de la précédente. Les champignons viennent à vous et pour les cueillir, il vous suffit de les intercepter avec votre panier.

Vous constatez que les champignons ne se déplacent pas tous à la même vitesse ; cela rend plus difficile l'appréciation du moment où ils seront au niveau de votre panier et donc le choix de l'ordre dans lequel il faut les ramasser.

Enfin, pour comble de tout, tous n'ont pas la même valeur ! Les champignons blancs sans "volve" rapportent un nombre de points égal à dix fois leur vitesse (qui va de 1 à 4). Les mêmes champignons, rouges sur fond blanc rapportent vingt fois leur vitesse. Par contre, les champignons à volve représentent des amanites. Les blancs sont seulement dangereux et vous font perdre un nombre de points égal à 100 fois leur vitesse. Par contre, les rouges sur fond blanc sont mortels et leur "cueillette" interrompt immédiatement la partie. Vous verrez un récapitulatif de ces règles apparaître en haut à droite de l'écran.

Vous disposez d'un temps limité pour obtenir un maximum de points. Lorsqu'ils tombent dans votre panier, les champignons comestibles se reconnaissent à un bruit de clochette. Les autres se signalent par un coup de fusil pour les dangereux et par une explosion pour les mortels (nous devrions dire le mortel car vous n'aurez guère l'occasion d'en goûter plusieurs au cours d'une même partie).



Vue de l'écran pendant le déroulement d'une partie

2. Le programme

```
100 REM ********** AMANITES ****************************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 8000
140 : REPEAT
150 : FOR I=1 TO NC
160 : GOSUB 4000: GOSUB 3000: T=T-1: IF FI=1 THEN 190
170 : NEXT
190 : UNTIL FI=1 OR T<=0
200 : PLOT 20,0,12: PLOT 21,0,"rejouez vous      "
210 : GET R$: GET R$
220 : IF SC>MS THEN MS=SC
230 UNTIL R$<>"O"
240 END
2990 REM ----- SP DEPLACEMENT DU CHAMPIGNON I ---------
3000 YN=Y(I)+V(I): PLOT X(I),Y(I),32
3030 IF YN<Y2 THEN Y(I)=YN: PLOT X(I),Y(I),C(I): RETURN
3040 IF SCRN(X(I),Y(I)+1)<>32 THEN GOSUB 5000
3050 GOSUB 6000: RETURN
3990 REM ----- SP DEPLACEMENT PANIER -----------------
4000 C=PEEK(#208): IF C=#38 THEN RETURN
4020 PLOT X,Y,"   "
4030 X=X+(C=#BC)*(X<36)-(C=#AC)*(X>1)+3*(C=#9C)*(X<34)-3*(C=#B4)*(X
>3)
4040 PLOT X,Y,CR$: RETURN
4990 REM ----- SP CHAMPIGNON CUEILLI -----------------
5000 IF C(I)=223 THEN 5100
5010 K=10: PING: IF C(I)=222 THEN K=20
5015 IF C(I)=95 THEN K=-100: SHOOT
5020 SC=SC+K*V(I): IF SC<0 THEN SC=0
5040 PLOT 4,0,"       ":PLOT 4,0,STR$(SC)
5050 RETURN
5100 EXPLODE: FI=1
5110 RETURN
5990 REM ----- TIRAGE NOUVEAU CHAMPIGNON ------------
6000 Y(I)=1: X(I)=1+INT(RND(1)*38)
6010 C(I)=CC(INT(RND(1)*8+1)): V(I)=INT(RND(1)*VM+1)
6020 RETURN
```

```
8000 REM ----- SP INITIALISATION D'UNE PARTIE -----------
8010 CLS: PLOT 1,0,2: PLOT 0,26,IR: SC=0
8020 PLOT 2,0,"SC       M-SC       "
8030 PLOT 15,0,STR$(MS): PLOT 22,0,IN
8040 PLOT 23,0,CHR$(CC(1))+"10"+CHR$(CC(2))+"20"
8050 PLOT 29,0,CHR$(CC(3))+"-100"+CHR$(CC(4))+"FIN"
8060 FOR I=1 TO NC: GOSUB 6000: NEXT I
8070 X=15: Y=26: FI=0: T=1000
8080 PLOT X,Y,CR$
8090 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU ---------------
9010 IN=6: PA=0: IR=3: CLS: INK IN: PAPER PA
9020 X1=1: X2=38: Y1=1: Y2=26
9030 NC=9: VM=4: MS=0
9040 DIM X(NC),Y(NC),C(NC),V(NC),CC(8)
9045 POKE #26A,10
9050 DATA 12,30,63,63,12,12,12,12
9060 DATA 12,30,63,63,12,12,30,30
9070 CC(1)=94: CC(2)=94+128: CC(3)=95: CC(4)=95+128
9075 CC(5)=94: CC(6)=94: CC(7)=94+128: CC(8)=94+128
9077 C1=64: C2=93: C3=96
9080 C=CC(1): GOSUB 9500: C=CC(3): GOSUB 9500
9090 DATA 56,56,56,28,31,15,7,1
9100 C=C1: GOSUB 9500
9110 DATA 0,0,0,0,63,63,63,63
9120 C=C2: GOSUB 9500
9130 DATA 7,7,7,14,62,60,56,32
9140 C=C3: GOSUB 9500
9150 CR$=CHR$(C1)+CHR$(C2)+CHR$(C3)
9160 RETURN
9500 REM ----- SP INIT CARAC GRAPHIQUES ---------------
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier la couleur du panier

Remplacez, en 9010, le nombre 3 par une valeur de votre choix (1 à 7).

2. Pour modifier la durée d'une partie

Remplacez, en 8070, le nombre 1000 par une valeur de votre choix.

3. Pour modifier le nombre de champignons présents en même temps sur l'écran

Remplacez, en 9030, la valeur 9 par un nombre de votre choix. Notez que :

- une valeur trop petite conduit à une perte d'intérêt du jeu,
- une valeur trop grande entraîne un déplacement trop "saccadé" des champignons. (Seule, une programmation en langage machine permettrait de déplacer convenablement un nombre important de champignons).

4. Pour modifier la vitesse maximum de déplacement d'un champignon

Remplacez, en 9030, le nombre 4 par une valeur de votre choix comprise entre 1 et 4 (une valeur supérieure à 4 conduirait à un fonctionnement défectueux).

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— SCROLL.

— Écriture directe en mémoire d'écran.
— Configuration clavier/écran.

b) Le programme principal (100-240)

110 appelle le sous-programme d'initialisation du jeu.

120 et 230 répètent les instructions 130 à 220 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer :

130 appelle le sous-programme d'initialisation d'une partie.

140 et 190 répètent les instructions 150 à 170 jusqu'à ce que le temps imparti soit écoulé ou que vous ayez "cueilli" un champignon mortel (FI = 1) :

150 à 170 déplacent tour à tour chacun des champignons (sous-programme 3000) ainsi que le panier (sous-programme 4000).

200-210 vous demandent si vous souhaitez rejouer.

220 actualise le meilleur score.

c) Le sous-programme d'initialisation du jeu (9000-9160)

9010 fixe les couleurs du fond, de l'encre et celle du panier.

9020 fixe les limites du domaine dans lequel évoluent les champignons.

9030 fixe le nombre de champignons apparaissant simultanément sur l'écran, la vitesse maximum et initialise le meilleur score.

9040 donne les dimensions des tableaux.

9045 supprime le bip du clavier et le curseur de l'écran.

9050-9150 fabriquent les caractères graphiques servant à représenter les diverses sortes de champignons ainsi que le panier.

d) Le sous-programme d'initialisation d'une partie (8000-8080)

8010-8050 initialisent le score, affichent le meilleur score, les différents types de champignons et les points correspondants.

8060 appelle huit fois le sous-programme 6000 qui tire au hasard la position, le type et la vitesse d'un champignon.

8070-8080 dessinent le panier dans sa position initiale.

e) Le sous-programme de déplacement du champignon numéro I (2990-3050)

3000 calcule la nouvelle position du champignon et l'efface de sa position actuelle.

3030 dessine le champignon dans sa nouvelle position s'il n'a pas atteint le bas de l'écran.

3040 détermine, dans le cas contraire, s'il rencontre le panier et, dans ce cas, appelle le sous-programme "champignon cueilli".

3050 appelle le sous-programme "tirage d'un nouveau champignon", à chaque fois qu'un champignon disparaît en bas de l'écran.

f) Le sous-programme de déplacement du panier (3990-4040)

4000 examine si une touche du clavier a été enfoncée.

4020 efface le panier de sa position actuelle.

4030 détermine la nouvelle position du panier en fonction de la touche enfoncée.

4040 dessine le panier dans sa nouvelle position.

g) Le sous-programme "champignon cueilli" (4990-5110)

5000 examine s'il s'agit ou non d'un champignon mortel.

5010-5050 émettent le son approprié et actualisent le score dans le cas d'un champignon non mortel.

5100-5110 font entendre une explosion et signalent que la partie est finie, dans le cas d'un champignon mortel.

5. Liste des variables

PA	couleur du fond (ici noir)
IN	couleur de l'encre utilisée pour représenter certains champignons (les autres étant représentés en "vidéo inverse")

IR	couleur du panier
X1,X2,Y1,Y2	coordonnées du domaine dans lequel évoluent les champignons
NC	nombre de champignons présents simultanément sur l'écran
VM	vitesse maximum de déplacement d'un champignon
MS	meilleur score
X(NC),Y(NC)	tableaux contenant les coordonnées des champignons présents sur l'écran
C(NC)	code ASCII du caractère représentant chacun des champignons présents sur l'écran
V(NC)	tableau contenant la vitesse de chaque champignon
CC(8)	code ASCII des huit caractères représentant les huit types de champignons (ici, plusieurs de ces caractères sont, en fait, identiques)
C1,C2,C3	codes ASCII des trois caractères représentant le panier
CR$	chaîne de trois caractères représentant le panier
SC	score
X,Y	coordonnées de l'extrémité gauche du panier
FI	indicateur de fin de partie : 0 : cas normal 1 : fin de partie
T	durée d'une partie
I	numéro du champignon en cours de déplacement (sauf dans les sous-programmes 8000 et 9500)
K	nombre de points attribués au champignon cueilli
YN	nouvelle ordonnée du champignon en cours de déplacement

Variables auxiliaires

C I R$

18

Les envahisseurs

1. Le jeu

Les extra-terrestres envahissent notre planète. Bien que non armés, leurs intentions ne sont nullement pacifiques. Vous êtes chargés d'en détruire le maximum à l'aide de missiles classiques.

Les envahisseurs se présentent par vagues de dix soucoupes. Leur entrée brutale dans l'atmosphère terrestre entraîne des perturbations de tout genre dont les deux plus importantes sont : violente illumination du ciel, neutralisation de votre base de tir (elle ne peut ni se déplacer, ni tirer de missiles).

Les envahisseurs ont prévu que les différentes vagues se succèdent de plus en plus rapidement. Cependant, ils tiennent compte de votre habileté à les détruire. Plus vous serez rapides, plus le temps séparant deux vagues sera élevé. Votre ordinateur central, couplé à vos radars vous renseignera sur le délai prévu entre une vague et la suivante. Vous pourrez ainsi mesurer l'effet de votre effort destructeur.

Au début du jeu, trois vagues apparaissent successivement en haut de l'écran. Les soucoupes s'immobilisent alors dans l'attente de la vague suivante.

Vous voyez votre base de tir en bas de l'écran. Elle se déplace à l'aide des touches "←" et "→" et vous tirez un missile en pressant la barre d'espace. Vous ne pouvez tirer de nouveau missile tant que le précédent n'a pas atteint son but ou disparu en haut de l'écran. Par contre, vous pouvez déplacer votre base pendant qu'un missile est en l'air. Très souvent même, vous serez obligés de le faire pour gagner du temps.

Quand une nouvelle vague de dix soucoupes se présente, le ciel devient blanc et vous ne pouvez plus agir sur votre base de tir. Les soucoupes déjà présentes descendent légèrement pour faire place aux nouveaux arrivants puis tout redevient normal (si l'on peut dire).

Le jeu se poursuit jusqu'à ce que l'une des soucoupes atteigne le niveau du sol. Vous avez donc tout intérêt à éliminer les soucoupes les plus basses.

Le programme affiche en permanence votre score (nombre de soucoupes détruites), le meilleur score et le nombre d'envahisseurs présents dans le ciel (N — E). (Voir photo 7 hors texte : *Vue de l'écran pendant le déroulement du jeu*).

2. Le programme

```
100 REM ************ ENVAHISSEURS **********************
110 GOSUB 5000
120 REPEAT
125 GOSUB 4000: FI=0
130 FOR I=1 TO 4: GOSUB 2000: NEXT I
135 DN=DV: DC=DN
140 REPEAT
145 FOR I=1 TO LV: GOSUB 2000: NEXT I
147 CH$="     ":ID=36: GOSUB 7000: CH$=STR$(INT(DC)): GOSUB 7000
150 FOR I=1 TO DC
160 GOSUB 1500: GOSUB 1000: GOSUB 1500
170 NEXT I
180 DN=DN*TE: DC=DN*(EV*30-NT)/(EV*30)
```

```
190 UNTIL FI=1
200 IF SC<MS THEN 230
210 MS=SC: CH$=STR$(SC): ID=14: GOSUB 7000
230 WAIT 200: CLS
240 PLOT 5,10,"voulez vous rejouer(O/N)?"
250 GET R$: GET R$
260 UNTIL R$<>"O"
270 POKE #26A,3
280 END
1000 REM ---- SP DEPLACEMENT BASE DE TIR ---------------
1010 DL=PEEK(#208)
1020 IF DL=#84 AND YO=YT THEN XO=XT: YO=YT-1: PLOT XO,YO,CO: RETURN
1025 IF DL<>#AC AND DL<>#BC THEN RETURN
1030 XN=XT+(DL=#AC)-(DL=#BC)
1040 IF XN>X2 OR XN<X1 THEN RETURN
1050 PLOT XT,YT,32: PLOT XN,YT,CT: XT=XN
1060 RETURN
1500 REM ---- SP DEPLACEMENT MISSILE -------------------
1510 IF YO=YT THEN RETURN
1520 YN=YO-1: PLOT XO,YO,32
1530 IF YN<Y1 THEN YO=YT: RETURN
1540 IF SCRN(XO,YN)<>CE THEN PLOT XO,YN,CO: YO=YN: RETURN
1550 SHOOT: PLOT XO,YN,32
1560 NE(YN+1)=NE(YN+1)-1: SC=SC+1: NT=NT-1
1570 CH$=STR$(SC): ID=4: GOSUB 7000
1580 CH$=STR$(NT): ID=25: GOSUB 7000
1590 YO=YT
1600 RETURN
2000 REM ---- SP NOUVELLE VAGUE ENVAHISSEURS ----------
2005 PAPER 7
2010 IF YO<>YT THEN PLOT XO,YO,32: YO=YT
2020 IF NE(YT-1)<>0 THEN GOSUB 3000: RETURN
2030 FOR K=YT TO 2 STEP -1: NE(K)=NE(K-1): NEXT K
2040 NE(1)=EV: NT=NT+EV
2045 CH$=STR$(NT): ID=25: GOSUB 7000
2050 PRINT CHR$(11);: PLOT XT,YT,CT
2060 FOR K=1 TO EV
2070 : X=INT(RND(1)*(X2-X1)+1)
2080 : IF SCRN(X,Y1)<>32 THEN 2070
2090 : PLOT X,Y1,CE
2100 NEXT K
2105 PAPER PA
2110 RETURN
3000 REM ---- SP FIN DE PARTIE --------------------
3010 FOR K=7 TO 0 STEP -1
3020 : PAPER K: ZAP
```

```
3030 : FOR L=7 TO 0 STEP -1
3040 :  INK L: WAIT 3
3050 : NEXT L
3060 NEXT K
3070 WAIT 100: FI=1: PAPER PA: INK IN
3080 RETURN
4000 REM ---- SP INITIALISATION D'UNE PARTIE ---------
4010 CLS
4015 CH$="SC        M-SC        N-EN      DELAI     "
4017 ID=2: GOSUB 7000
4018 CH$=STR$(MS): ID=14: GOSUB 7000
4020 XT=X1+INT(RND(1)*(X2-X1)+1)
4030 YT=26: XO=XT: PLOT XT,YT,CT
4040 DC=DV: SC=0: FI=0: NT=0
4050 FOR I=1 TO YT: NE(I)=0: NEXT I
4060 RETURN
5000 REM ---- SP INITIALISATIONS DU JEU -----------
5005 POKE #26A,10
5010 X1=1: X2=38: Y1=0: Y2=26
5020 DV=160: EV=5: LV=2: TE=.975
5040 DIM NE(26)
5045 MS=0
5050 PA=6: IN=1
5060 PAPER PA: INK IN
5065 CT=94: CO=95: CE=96
5070 DATA 12,12,12,12,30,30,63,63
5080 C=CT: GOSUB 8000
5090 DATA 12,12,12,12,12,12,12,12
5100 C=CO: GOSUB 8000
5110 DATA 0,12,30,45,63,30,0,0
5120 C=CE: GOSUB 8000
5130 AD=#BB80: POKE AD,2
5160 RETURN
7000 REM ---- SP AFFICHAGE EN LIGNE SUPERIEURE -------
7010 FOR K=1 TO LEN(CH$)
7020 POKE AD+ID+K-1,ASC(MID$(CH$,K,1))
7030 NEXT K
7040 RETURN
8000 REM ---- INIT CARAC GRAPHIQUES ----------------
8010 AM=46080+8*C
8020 FOR I=0 TO 7
8030 : READ XX: POKE AM+I,XX
8040 NEXT I
8050 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour modifier la couleur du ciel

Remplacez, en 5050, la valeur 6 (dans PA = 6) par un nombre de votre choix, entre 0 et 6. (Si vous utilisez 7 qui correspond à blanc, l'arrivée d'une nouvelle vague ne se signalera plus par un changement de couleur).

2. Pour modifier la couleur des soucoupes et de la base

Remplacez, en 5050, la valeur 1 (dans IN = 1) par un nombre de votre choix (entre 0 et 7).

3. Pour modifier le nombre de soucoupes par vague

Vous pouvez agir à la fois sur le nombre de soucoupes par ligne (actuellement 5) et sur le nombre de lignes par vague (actuellement 2).

Pour modifier le nombre de soucoupes par ligne, remplacez, en 5020, la valeur 5 (dans EV = 5) par le nombre de votre choix.

Pour modifier le nombre de lignes par vague, remplacez, en 5020, la valeur 2 (dans LV = 2) par un nombre de votre choix.

4. **Pour modifier le rythme de jeu** et le rendre ainsi plus facile ou plus difficile

Vous pouvez agir à la fois sur le délai initial séparant deux vagues (actuellement 160) et sur le coefficient réducteur de délai (actuellement 0.975).

Pour modifier le délai initial séparant deux vagues, remplacez, en 5020, la valeur 160 (dans DV = 160) par un nombre de votre choix.

Pour modifier le coefficient réducteur, remplacez, en 5020, la valeur .975 par un nombre de votre choix ne dépassant pas 1. Un nombre supérieur à 0.975 correspondra à une partie dont le rythme s'accélérera moins rapidement. Un nombre plus petit correspondra à une accélération plus rapide du rythme.

Notez que si vous souhaitez que le délai séparant deux vagues soit constant, il vous suffit de choisir un comme coefficient.

5. Pour utiliser une poignée de jeu

Chargez le programme "MANETTE 1" comme décrit en annexe. Branchez la poignée du côté droit et remplacez les lignes 1010 à 1030 par:

```
1010 DL=PEEK(#400)
1020 IF DL=191 THEN RETURN
1025 IF (DL AND 32)=0 THEN XO=XT: YO=YT-1: PLOT XO,YO,CO: RETURN
1030 XN=XT+((DL AND 1)=0)-((DL AND 2)=0)
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Écriture directe en mémoire d'écran.
— SCROLL.
— Poignées de jeu (facultatif).

b) Le programme principal

110 appelle le sous-programme d'initialisation du jeu.

120 et 260 répètent les instructions 125 à 250 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne désirez plus rejouer:

125 appelle le sous-programme d'initialisation d'une partie.

130 fait apparaître quatre rangées d'envahisseurs.

140 et 190 répètent les instructions 145 à 180 jusqu'à ce que les envahisseurs aient atteint le sol (FI = 1):

145 fait apparaître une nouvelle vague (deux rangées de soucoupes).

147 affiche le délai prévu.

150-170 répètent un nombre de fois correspondant au délai prévu l'appel du sous-programme de déplacement de la base de tir et l'appel du sous-programme de déplacement de missile.

200-230 actualisent le meilleur score, l'affichent et appellent le sous-programme de fin de partie.

240-250 vous demandent si vous désirez rejouer.

c) Le sous-programme d'initialisation du jeu (5000-5160)

5005 supprime le bip du clavier et le curseur d'écran.

5010 fixe les limites du domaine de jeu.

5020 initialise le délai séparant deux vagues, fixe le nombre d'envahisseurs par ligne, le nombre de lignes par vague et le coefficient de réduction de délai.

5045 initialise le meilleur score.

5050-5060 fixe les couleurs d'encre et de fond.

5065-5120 fabriquent les caractères graphiques représentant la base de tir, les missiles et les soucoupes.

5130 fixe la couleur d'encre utilisée sur la ligne supérieure.

d) Le sous-programme d'initialisation d'une partie (4000-4060)

4010 efface l'écran.

4015-4017 affichent les "titres" en ligne supérieure.

4018 affiche le meilleur score.

4020-4030 dessinent la base de tir.

4040 initialise le délai séparant deux vagues, le score, l'indicateur de fin de partie et le nombre de soucoupes détruites.

4050 initialise le tableau: nombre de soucoupes par ligne.

e) Le sous-programme de déplacement de la base de tir (1000-1060)

1010 examine le clavier.

1020 initialise un tir de missile si la barre d'espace a été pressée et qu'aucun tir n'est en cours.

1030 calcule la nouvelle position de la base.

1040 vérifie que la base ne sort pas des limites.

1050 dessine la base dans sa nouvelle position.

f) Le sous-programme de déplacement de missile (1500-1590)

1510 examine si un tir est en cours.

1520 définit la nouvelle position du missile et efface l'ancien.

1530 examine si le missile sort de l'écran et si oui remet à zéro l'indicateur "tir en cours".

1540 examine si une soucoupe est atteinte et, si ce n'est pas le cas, dessine le missile dans sa nouvelle position.

1550-1600 cas où une soucoupe est atteinte :

1550 émet un bruit et efface la soucoupe.

1560-1580 actualisent le nombre de soucoupes par ligne, le score et le nombre de soucoupes présentes.

1590 remet à zéro l'indicateur "tir en cours".

g) Le sous-programme "nouvelle vague d'envahisseurs" (2000-2110)

2005 rend l'écran blanc.

2010 supprime le tir en cours s'il y a lieu.

2020 vérifie que la dernière ligne ne contient pas d'envahisseurs. Si ce n'est pas le cas, le sous-programme de fin de partie est appelé.

2030-2045 met à jour le tableau nombre d'envahisseurs par ligne et le nombre d'envahisseurs présents sur l'écran.

2050 fait défiler le contenu de l'écran vers le bas (scroll) et redessine la base de tir.

2060-2100 place au hasard cinq envahisseurs en haut de l'écran.

2105 redonne à l'écran sa couleur d'origine.

5. *Liste des variables*

X1,X2,Y1,Y2	limites du domaine de jeu
DV	délai maximum séparant deux vagues
EV	nombre d'envahisseurs par ligne
LV	nombre de lignes formant une vague
TE	coefficient réducteur du délai séparant deux vagues
NE(26)	tableau contenant le nombre de soucoupes de chaque ligne d'écran
MS	meilleur score
PA	couleur de fond
IN	couleur d'encre utilisée pour les soucoupes et la base de tir
CT	code ASCII du caractère représentant la base de tir
CO	code ASCII du caractère représentant le missile
CE	code ASCII du caractère représentant les soucoupes
AD	adresse mémoire correspondant au début de la ligne supérieure
ID	désignation d'un emplacement courant en ligne supérieure
XT,YT	coordonnées de la base de tir
XO,YO	coordonnées du missile. Si YO = YT aucun tir n'est en cours
DC	délai courant prévu entre deux vagues, compte tenu du nombre d'envahisseurs présents sur l'écran
SC	score
FI	indicateur de fin de partie 0 : cas normal 1 : fin de partie
NT	nombre d'envahisseurs présents sur l'écran
DN	délai courant prévu entre deux vagues, compte non tenu du nombre d'envahisseurs présents sur l'écran
XN	nouvelle abscisse de la base de tir
YN	nouvelle ordonnée du missile

Variables auxiliaires

AM I XX K L CH$ X DL

19

Alunissage

1. Le jeu

Serez-vous capable de poser une fusée en douceur sur la lune. C'est ce que va vous révéler ce jeu.

Vous êtes aux commandes d'une fusée. Au départ, elle se trouve, immobile, à 100 mètres d'altitude. Sous l'effet de la pesanteur, votre fusée descend de plus en plus rapidement vers la surface. Heureusement vous disposez de "rétro-fusées" que vous pouvez faire fonctionner quand vous le souhaitez pour réduire votre vitesse de chute. Vous pouvez même, si vous n'y prenez garde, changer le sens de votre vitesse et vous éloigner de la surface de la lune. Toutefois, ce n'est pas dans votre intérêt car votre carburant est limité et vous risquez fort dans ce cas de ne plus pouvoir amortir la chute finale.

Les seules informations qui vous sont fournies par les instruments de bord sont :

— altitude,

— vitesse,
— niveau de carburant.

Au début du jeu, vous voyez se dessiner les trois instruments sur l'écran :

- *à gauche* : le compteur de vitesse gradué de 0 à 5. La vitesse y est représentée par une bande de couleur verte quand vous descendez et bleue quand vous montez,
- *au centre* : l'altimètre gradué de 0 à 10. 0 représente le niveau de la surface lunaire et 10 l'altitude dont vous partez (100 m si vous ne modifiez pas le programme). L'altitude y est représentée par une bande jaune. Si, par hasard, vous dépassez votre altitude de départ, le programme fonctionnera normalement. Seul votre altimètre restera bloqué au maximum tant que vous ne serez pas redescendu au-dessous de 100 m. Quand votre altitude tombera au-dessous de 10 m, elle sera représentée par une bande violette, avec une précision dix fois plus grande. Autrement dit, le maximum correspondra alors à 10 m. Cette particularité doit vous permettre de mieux contrôler la phase finale de l'alunissage.
- *à droite* : votre jauge de carburant. Le niveau y est matérialisé par une bande rouge.

  Dès que le jeu est commencé, votre fusée se met à descendre. Vous êtes renseigné sur sa vitesse de façon visuelle grâce au compteur et également de façon sonore grâce à un son dont la hauteur (fréquence) augmente avec la vitesse. Vous pouvez ainsi quitter le compteur des yeux tout en restant renseigné de ce qui se passe.

Vous déclenchez les rétro-fusées grâce à la barre d'espace. Celles-ci fonctionnent tant que vous maintenez la pression.

Quand votre altitude atteint zéro, le programme considère que vous avez aluni. Il vous communique la vitesse avec laquelle vous avez pris contact avec le sol et le niveau de carburant restant. Il vous fournit un score qui dépend de ces deux valeurs. Toutefois, si votre vitesse de contact est supérieure à 2, le programme vous fera entendre l'explosion de votre fusée et votre score sera nul. Dans les deux cas, vous verrez apparaître le meilleur score;

REMARQUE: Par souci de simplification des mesures des instruments de bord, les valeurs fournies pour la vitesse ne correspondent pas à la réalité lunaire. Pour les puristes, disons que la vitesse indiquée serait exacte si l'on divisait l'altitude indiquée par environ 5.

2. Le programme

```
100 REM ********** ALUNISSAGE **************************
110 GOSUB 9000
120 REPEAT
130 : GOSUB 5000
135 : PLAY 1,0,0,0
140 : REPEAT
150 :  CL=PEEK(#208)
160 :  IF CL=#84 AND CA>0 THEN V=V+D2: CA =CA-1 ELSE V=V-D1
170 :  H=H+V: GOSUB 4000
180 : UNTIL H<0
185 : V=INT(V*1000)/1000
190 : IF V>=-2 THEN GOTO 200
195 :  PRINT"vous vous ecrasez a ";-V;" m/s":EXPLODE:SC=0:GOTO 250
200 : PRINT "vous alunissez a ";-V;" m/s"
210 : IF V<-1 THEN K=-1/(2*V)
220 : IF V>=-1 AND V<=0.1 THEN K=1+V/2
230 : IF V>-0.1 THEN K=1
235 : PLAY 0,0,0,0
240 : SC=INT(K*200+CA)
250 : PRINT "SCORE ";SC;" M-SCORE ";MS;" CARBURANT ";CA
260 : IF SC>MS THEN MS=SC
270 : WAIT 100: PRINT "voulez vous rejouer";
280 : GET R$: GET R$
290 UNTIL R$<>"O"
300 END
4000 REM ----- SP VISUALISATION DE B,H,CA AVEC SONO ------
4010 HX=INT(-H1*V/VM): CX=C1
4020 IF V>0 THEN CX=C2: HX=-HX
4030 IF HX>H1 THEN HX=H1
4040 CURSET X1+6,Y1,0: IF HX>=1 THEN FILL HX,L1/6-2,CX
4050 IF H1-HX>=1 AND HX>0 THEN FILL H1-HX,L1/6-2,16
4060 HX=H1*H/HM: CX=C3: IF HX>H1 THEN HX=H1
4070 IF H<HM/10 THEN HX=10*HX: CX=C4
4075 HX=INT(HX)
4080 CURSET X2+6,Y1,0
4085 IF H1-HX>=1 AND HX>=0 THEN FILL H1-HX,L2/6-2,16
4090 IF HX>=1 THEN FILL HX,L2/6-2,CX
4100 HX=INT(H1*CA/CM)
4110 CURSET X3+6,Y1,0: IF H1-HX>=1 THEN FILL H1-HX,L3/6-2,16
4120 IF HX>=1 THEN FILL HX,L3/6-2,C5
4130 IF V<0.01 AND V>-0.01 THEN 4160
4140 F=200/V: IF F<0 THEN F=-F
4150 SOUND 1,F,5
4160 RETURN
```

```
5000 REM ----- SP INITIALISATION D'UNE PARTIE ---------
5005 V=0: H=HM: CA=CM
5010 GOSUB 4000
5020 PRINT: PRINT: PRINT
5030 RETURN
8000 REM ----- SP TRACE CADRE EN X,Y DE DIMENSION H,L --------
8010 CURSET X-1,Y-1,1
8020 DRAW L+2,0,1: DRAW 0,H+1,1
8030 DRAW -(L+2),0,1: DRAW 0,-(H+1),1
8040 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU --------------
9010 PA=0: IN=3: CLS: PAPER PA : INK IN
9020 MS=0: D1=0.1: D2=0.05
9030 C1=16+2: C2=16+6: C3=16+3: C4=16+5: C5=16+1
9040 PG=0: IG=7: HIRES: PAPER PG: INK IG
9050 X1=24: X2=108: X3=180
9060 Y1=1: H1=190: L1=41: L2=41: L3=35
9070 VM=5: HM=100: CM=80
9080 X=X1: Y=Y1: L=L1: H=H1: GOSUB 8000
9090 X=X2: Y=Y1: L=L2: H=H1: GOSUB 8000
9100 X=X3: Y=Y1: L=L3: H=H1: GOSUB 8000
9105 X=X1-12
9110 FOR V=1 TO VM
9120 : Y=Y1+V*H1/VM
9130 : CURSET X,Y-8,0: CHAR ASC(MID$(STR$(V),2,1)),0,1
9140 : CURSET X+6,Y,1: DRAW 6,0,1
9150 NEXT V
9160 FOR I=0 TO 9
9170 : Y=Y1+H1*(10-I)/10
9180 : CURSET X2-6,Y,0: DRAW 6,0,1
9190 : CURSET X2-12,Y-8,0: CHAR ASC(MID$(STR$(I),2,1)),0,1
9200 NEXT I
9210 CURSET X1+L1-5,Y1,0: FILL H1,1,16
9220 CURSET X2+L2-5,Y1,0: FILL H1,1,16
9230 CURSET X3+L3-5,Y1,0: FILL H1,1,16
9240 POKE #26A,10
9250 RETURN
```

3. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Lecture rapide du clavier.
— Configuration clavier/écran

b) Le programme principal (100-300)

110 appelle le sous-programme d'initialisation du jeu.

120 et 290 répètent les instructions 130 à 280 (déroulement d'une partie) jusqu'à ce que vous disiez que vous ne souhaitez plus rejouer:

130 appelle le sous-programme d'initialisation d'une partie.

140 et 180 répètent les instructions 150 à 180 qui "déplacent la fusée" jusqu'à ce que vous ayez aluni:

150 examine le clavier.

160 actualise la vitesse et éventuellement le niveau de carburant.

170 actualise l'altitude et appelle le sous-programme qui présente les différentes informations sur vos instruments de bord.

185 arrondit la vitesse d'alunissage à 3 décimales (avant de la présenter sur l'écran).

190 examine votre vitesse d'alunissage.

195 vous dit à quelle vitesse vous vous êtes écrasé (si celle-ci est supérieure à 2 m/s).

200-240 calculent votre score, lorsque votre vitesse d'alunissage ne dépasse pas 2 m/s.

250-260 affichent votre score, le meilleur score, votre vitesse et votre niveau de carburant et actualisent le meilleur score.

270-280 vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9250)

9010 fixe les couleurs de fond et d'encre utilisées pour le bas de l'écran.

9020 initialise le meilleur score et fixe les variations élémentaires de vitesse.

9030-9040 fixent les différentes couleurs employées en graphique haute résolution (HIRES).

9050-9060 fixent les emplacements et les dimensions des instruments de bord.

9070 fixe la vitesse maximum indiquée par le compteur, l'altitude initiale et le niveau initial de carburant.

9080-9100 dessinent les trois cadres des instruments de bord (en utilisant le sous-programme 8000).

9105-9150 trace la graduation du compteur de vitesse.

9160-9200 trace la graduation de l'altimètre.

9210-9230 place l'attribut "fond noir" dans les parties droites des cadres.

9240 supprime le bip du clavier et le curseur de l'écran.

d) Le sous-programme d'initialisation d'une partie (5000-5030)

5005 initialise la vitesse, l'altitude et le niveau de carburant.

5010 appelle le sous-programme présentant les différentes informations sur les instruments de bord.

5020 efface le bas de l'écran.

e) Le sous-programme de présentation des informations (4000-4160)

4010-4050 visualisent la vitesse :

4010-4020 déterminent la hauteur de la bande et sa couleur.

4030 examine si la vitesse dépasse la vitesse maximum affichable.

4040-4050 effectuent le coloriage de la bande.

4060-4090 visualisent l'altitude :

4060-4075 déterminent la hauteur de la bande et sa couleur.

4080-4090 effectuent le coloriage de la bande.

4100-4120 visualisent le niveau de carburant.

4130-4150 adaptent le son émis à la vitesse.

4. *Liste des variables*

PA	couleur de fond du bas de l'écran
IN	couleur d'encre employée pour le bas de l'écran
MS	meilleur score
D1	variation élémentaire de la vitesse, en cas de chute libre
D2	variation élémentaire de la vitesse quand les retro-fusées sont en action
C1,C2	attributs servant à colorier la bande du compteur de vitesse. C1 correspond à une vitesse dirigée vers le bas et C2 à une vitesse dirigée vers le haut.
C3,C4	attributs servant à colorier la bande de l'altimètre. C3 correspond à la présicion normale et C4 à une précision multipliée par dix.
C5	attribut servant à colorier la bande de la jauge de carburant
PG	couleur de fond de la partie graphique
IG	couleur d'encre utilisée pour les cadres des instruments de bord
X1,X2,X3	abscisse du côté gauche de chacun des instruments de bord
Y1	ordonnée commune du haut des instruments de bord
H1	hauteur des trois instruments de bord
L1,L2,L3	largeur de chacun des trois instruments de bord
VM	vitesse maximum indiquée par le compteur
HM	altitude de départ
CM	niveau initial du carburant
V	vitesse de la fusée (négative quand la fusée descend, positive quand elle monte)
H	altitude
CA	carburant restant
SC	score

Variables auxiliaires

X Y L I CL K R$

H V (dans sous-programme 9000)

20

Billard

1. Le jeu

Vous allez pouvoir vous adonner aux joies de ce jeu, sans risquer de déchirer le tapis d'un coup de canne malencontreux.

Le but consiste à lancer votre boule, grâce à une canne, de manière à ce qu'elle vienne toucher une autre boule après deux rebonds sur les bords du billard. Si vous touchez sans rebond (ce qui est très facile) ou après un seul rebond, vous ne marquez aucun point.

Le programme commence par vous dessiner un tapis de billard (vert) entouré d'une bordure violette. Vous voyez apparaître deux boules, en des emplacements quelconques. L'une d'entre elles est accompagnée d'une "canne de billard" représentée par un trait. Vous allez tout d'abord devoir mettre en place la canne dans la position qui vous semble la meilleure pour l'objectif visé (toucher l'autre boule en deux rebonds). Vous utilisez pour cela les touches fléchées qui vous donnent deux vitesses de déplacement de la canne (autour de la boule):

←: déplacement lent dans le sens inverse des aiguilles d'une montre.

↓: déplacement rapide dans le sens inverse des aiguilles d'une montre.

→: déplacement lent dans le sens des aiguilles d'une montre.

↑: déplacement rapide dans le sens des aiguilles d'une montre.

Quand la position vous paraît correcte, vous "tirez" en pressant la barre d'espace. Vous entendez alors le son de la canne heurtant votre boule puis celle-ci se déplace sur le tapis. Chaque rebond se signale par un son approprié.

Si vous touchez la boule dans les conditions requises vous serez récompensés par le jeu d'un accord parfait (notez, au passage, les possibilités sonores de l'ORIC).

Douze coups vous seront ainsi proposés. A la fin vous obtiendrez un score qui dépendra du nombre de coups réussis et du temps passé à mettre votre canne en place.

Le programme affiche en permanence le temps qui s'écoule pendant la mise en place de la canne, le numéro du coup et le nombre de coups réussis.

2. *Le programme*

```
100 REM ********** BILLARD **********
120 GOSUB 9000: GOSUB 8000
125 GOSUB 9500
130 FOR NB=1 TO NC
140 : GOSUB 7000
150 : REPEAT
160 :   XN=X+CT: YN=Y+ST
170 :   IF ABS(XC-XN)<6 AND ABS(YC-YN)<8 THEN TC=1
180 :   IF XN<X1+4 OR XN>X2-4 OR YN<Y1+4 OR YN>Y2-4 THEN GOSUB 2000
190 :   CURSET X-3,Y-4,0: CHAR CB,0,2
200 :   X=XN: Y=YN: CURSET X-3,Y-4,0: CHAR CB,0,1
210 : UNTIL TC=1 OR NR=RX+1
220 : GOSUB 4000
230 NEXT NB
240 WAIT 100: SC=INT(NT*1000-5*T)
250 IF SC<0 THEN SC=0
260 PRINT CHR$(13);CHR$(13)
290 PRINT " VOTRE SCORE EST :";SC
300 IF SC<MS THEN PRINT "LE MEILLEUR SCORE RESTE:";MS
```

```
310 IF SC>=MS THEN PRINT "LE MEILLEUR SCORE ETAIT";MS : MS=SC
320 WAIT 100
330 PRINT "voulez vous rejouer(O/N)?";
340 GET R$: GET R$
350 IF R$="O" THEN GOTO 125
360 PRINT CHR$(6);CHR$(17);
370 END
2000 REM ----- SP DEPLACEMENT AVEC REBOND --------------
2030 IF XN<X1+4 THEN XN=X1+4: NR=NR+1: CT=-CT
2040 IF XN>X2-4 THEN XN=X2-4: NR=NR+1: CT=-CT
2050 IF YN<Y1+4 THEN YN=Y1+4: NR=NR+1: ST=-ST
2060 IF YN>Y2-4 THEN YN=Y2-4: NR=NR+1: ST=-ST
2070 CURSET X-3,Y-4,0: CHAR CB,0,2
2080 X=XN: Y=YN: CURSET X-3,Y-4,0: CHAR CB,0,1
2090 SOUND 1,10000,0: PLAY 1,0,1,200
2100 RETURN
4000 REM ----- SP FIN D'UN COUP ------------------------
4010 IF TC=1 THEN SOUND 1,4000,0: PLAY 1,0,1,100
4020 IF TC<>1 OR NR<>RX THEN 4100
4030 MUSIC 1,3,1,0: PLAY 1,0,1,500: WAIT 35
4040 MUSIC 2,3,5,0: PLAY 2,0,1,500: WAIT 35
4050 MUSIC 3,3,8,0: PLAY 4,0,1,500: WAIT 30
4060 PLAY 7,0,1,1500: WAIT 60
4070 NT=NT+1
4080 FOR I=1 TO 25: PRINT CHR$(9);: NEXT I
4090 PRINT NT;CHR$(13);
4100 CURSET XC-3,YC-4,0: CHAR CB,0,0
4110 CURSET X-3,Y-4,0: CHAR CB,0,0
4200 RETURN
6000 REM ---- SP MISE EN PLACE DE LA CANE -------------
6010 CURSET X-3,Y-4,0: CHAR CB,0,1
6020 T=T+1: GOSUB 6500
6030 XE=X+LC*COS(TH): YE=Y+LC*SIN(TH)
6040 CURSET X,Y,0: DRAW XE-X,YE-Y,1
6060 CL=PEEK(#208)
6070 IF CL=#38 THEN T=T+0.3: GOSUB 6500: GOTO 6060
6080 IF CL=#84 THEN 6150
6090 TH=TH+((CL=#AC)-(CL=#BC)+7*((CL=#B4)-(CL=#9C)))*DT
6100 CURSET X,Y,0: DRAW XE-X,YE-Y,0
6120 GOTO 6010
6150 CURSET X,Y,0: DRAW XE-X,YE-Y,0
6170 CURSET X-3,Y-4,0: CHAR CB,0,1
6190 SOUND 1,28,0: PLAY 0,1,1,400
6200 RETURN
6500 REM ---- SP AFFICHAGE DU TEMPS -------------------
6510 PRINT " ";INT(T);
6520 PRINT CHR$(13);
6530 RETURN
```

```
7000 REM ----- SP DEBUT D'UN COUP ----------------------
7010 FOR I=1 TO 12: PRINT CHR$(9);: NEXT I
7015 PRINT NB;CHR$(13);
7020 XC=X1+5+INT(RND(1)*(X2-X1-10))
7030 YC=Y1+5+INT(RND(1)*(Y2-Y1-10))
7040 CURSET XC-3,YC-4,0
7050 CHAR CB,0,1
7060 X=X1+5+LC+INT(RND(1)*(X2-X1-10-2*LC))
7070 Y=Y1+5+LC+INT(RND(1)*(Y2-Y1-10-2*LC))
7080 IF SQR ((X-XC)^2+(Y-YC)^2)<=LC+5 THEN 7060
7090 CURSET X-3,Y-4,0
7100 CHAR CB,0,1
7110 TC=0: NR=0
7120 GOSUB 6000
7130 CT=-DE*COS(TH): ST=-DE*SIN(TH)
7140 RETURN
8000 REM ----- SP DESSIN DU BILLARD --------------------
8010 X1=12: X2=228: Y1=16: Y2=184
8020 IN=4: PA=2: B1=128: B2=165
8030 HIRES
8040 CURSET 0,0,0
8050 FILL 200,1,B1+PA+16
8060 CURSET 6,0,0
8070 FILL 200,1,B1+IN
8080 CURSET 12,0,0
8090 IF X1>12 THEN FILL 200,(X1-X2)/6,B1
8100 CURSET X1,0,0
8110 FILL Y1,(240-Y1)/6,B2
8120 CURSET X1,Y2,0
8130 FILL 200-Y2,(X2-X1)/6,B2
8140 CURSET X2,0,0
8150 FILL 200,(240-X2)/6,B1
8160 POKE #26A,10
8200 RETURN
9000 REM ----- SP INITIALISATIONS DU JEU --------------
9005 PAPER 6: INK 4: NC=12: MS=0: RX=2
9020 LC=40: DT=0.02
9030 DE=8: TH=6.28*RND(1)
9040 CB=64
9050 DATA 12,30,63,63,63,63,30,12
9060 AM=46080+8*CB
9070 FOR I=0 TO 7
9080 : READ XX: POKE AM+I,XX
9090 NEXT I
9200 RETURN
9500 REM ----- SP INITIALISATION D'UNE PARTIE ----------
9505 T=0: NT=0: PRINT CHR$(13);
```

```
9510 FOR I=1 TO 36: PRINT " ";:NEXT I
9540 PRINT "  TEMPS   BOULE   COUPS REUSSIS"
9550 FOR I=1 TO 36: PRINT " ";:NEXT I
9560 PRINT CHR$(13);
9570 RETURN
```

3. Si vous souhaitez personnaliser ce programme

1. Pour modifier la couleur des boules

Remplacez, en 8020, la valeur 4 (dans IN = 4) par un nombre de votre choix (0 à 7).

2. Pour modifier la couleur du "tapis"

Remplacez, en 8020, la valeur 2 (dans PA = 2) par un nombre de votre choix (0 à 7). Évitez toutefois d'employer la même couleur pour le tapis et les boules !

3. Pour modifier le nombre de coups par partie

Remplacez, en 9005, la valeur 12 (dans NC = 12) par un nombre de votre choix.

4. Pour rendre le jeu plus facile ou plus difficile

Vous pouvez agir sur le nombre obligatoire de rebonds. Pour cela, il vous suffit de remplacer, en 9005, la valeur 2 (dans RX = 2) par celle de votre choix. Ainsi :

- 0 conduira à un jeu extrêmement facile puisqu'aucun rebond ne sera nécessaire. Cela peut vous servir, au début, à vous entraîner.
- 1 conduira à un jeu assez facile ne demandant qu'un seul rebond.
- 2 correspond au programme actuel.
- 3 conduira à un jeu très difficile puisque trois rebonds seront nécessaires.
etc...

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran

b) Le programme principal (100-370)

120 appelle les sous-programmes d'initialisation du jeu et de dessin du billard.

125-350 sont répétées jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

125 appelle le sous-programme d'initialisation d'une partie.

130 et 230 répètent douze fois les instructions 140 à 220 qui constituent le jeu d'un coup:

140 appelle le sous-programme de début d'un coup (dessin des boules et mise en place de la canne).

150 et 210 répètent le déplacement de votre boule jusqu'à ce qu'elle touche l'autre boule ou que le nombre de rebonds dépasse deux:

160 détermine les nouvelles coordonnées de votre boule.

170 examine s'il y a contact entre les deux boules.

180 examine s'il y a rebond, et dans ce cas appelle le sous-programme "rebond".

190-200 déplacent votre boule.

220 appelle le sous-programme de fin d'un coup.

240-310 calculent votre score, l'affichent en même temps que le meilleur score qui est actualisé.

320-340 vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9200)

9005 fixe les couleurs de la bande inférieure de l'écran, fixe le nombre de

coups d'une partie, le nombre de rebonds requis et initialise le meilleur score.

9020-9030 fixent la longueur de la canne, sa position initiale et les déplacements élémentaires.

9035 supprime le bip du clavier et le curseur d'écran.

9040-9090 fabriquent le caractère graphique représentant les boules.

d) Le sous-programme de dessin du billard (8000-8200)

8010 fixe les coordonnées du tapis de jeu.

8020 fixe les couleurs du billard et les codes des caractères utilisés pour représenter les bords.

8030 place l'ORIC en mode graphique haute résolution.

8040-8150 dessinent les bords du billard.

e) Le sous-programme d'initialisation d'une partie (9500-9570)

9505 initialise le compteur de temps et le nombre de coups réussis.

9510-9560 affichent les "titres" de la bande inférieure (qui n'est pas en haute résolution, et dans laquelle on peut écrire par PRINT).

f) Le sous-programme de début d'un coup (7000-7140)

7010-7015 affichent le numéro du coup.

7020-7050 dessinent la boule "cible", après avoir tiré sa position au hasard.

7060-7100 tirent au hasard la position de votre boule, en tenant compte de la longueur de la canne, et la dessinent.

7110 initialisent l'indicateur "touché" et le nombre de rebonds.

7120 appelle le sous-programme de mise en place de la canne.

7130 détermine les déplacements élémentaires de la boule.

g) Le sous-programme de mise en place de la canne (6000-6210)

6010-6120 sont répétées jusqu'à ce que vous pressiez la barre d'espace :

6010 redessine la boule.

6020 actualise le temps.

6030-6040 dessinent la canne.

6060-6070 attendent qu'une touche du clavier soit pressée.

6080 examine s'il s'agit de la barre d'espace.

6090 détermine la nouvelle position de la canne en fonction de la touche pressée.

6100 efface la canne.

6150 efface la canne.

6170 redessine la boule.

6190 fait entendre le bruit de la canne qui frappe la boule.

h) Le sous-programme de déplacement avec rebond (2000-2100)

2030-2060 déterminent la nouvelle position de la boule (en évitant qu'elle ne franchisse le bord) et actualisent les déplacements élémentaires, en fonction du bord sur lequel rebondit la boule.

2070 efface la boule de sa position actuelle.

2080 dessine la boule dans sa nouvelle position.

2090 émet le bruit du rebond.

i) Le sous-programme de fin d'un coup (4000-4200)

4010 fait entendre un bruit de choc, s'il y a lieu.

4020 examine si le coup est réussi ou non.

4030-4090 font retentir, en cas de coup réussi, les notes do, mi, sol suivies de l'accord formé de ces trois notes puis actualisent le nombre de coups réussis.

4100-4110 effacent les deux boules.

5. *Liste des variables*

NC	nombre de coups d'une partie
MS	meilleur score
RX	nombre de rebonds requis
LC	longueur de la canne
DT	déplacement angulaire élémentaire de la canne
TH	angle formé par la canne avec l'axe horizontal
DE	déplacement élémentaire de la boule
CB	code ASCII du caractère représentant les boules
T	temps écoulé pendant la mise en place de la canne
NT	nombre de coups réussis
X1,X2,Y1,Y2	coordonnées du "tapis" du billard
IN	couleur des boules et de la canne
PA	couleur du tapis
B1	caractère représentant les bords verticaux du billard
B2	caractère représentant les bords horizontaux du billard
XC,YC	coordonnées de la boule cible
X,Y	coordonnées de votre boule
TC	indicateur "boule cible touchée" 0 : pas de boule touchée 1 : boule cible touchée
NR	compteur du nombre de rebonds, au cours d'un même coup
CT	déplacement horizontal élémentaire de votre boule
ST	déplacement vertical élémentaire de votre boule
XE,YE	coordonnées de l'extrémité de la canne
XN,YN	coordonnées du prochain emplacement de votre boule.
SC	score

Variables auxiliaires

AM I XX CL

21

Dictée musicale

1. Le jeu

Voici un jeu qui va vous permettre de révéler ou de développer vos talents de musicien.

Il s'agit bel et bien d'une dictée puisque le programme va vous faire entendre des notes qu'il vous faudra reconnaître en les plaçant sur une portée musicale. La pratique répétée d'un tel jeu peut vous permettre d'affiner considérablement votre "oreille".

Au départ, vous voyez apparaître une portée avec, en jaune la note do. Le programme vous fait alors entendre le son correspondant, en transformant la note en rouge, pendant tout le temps où elle est jouée. Une deuxième note vous est alors jouée. Il vous faut la reconnaître en plaçant correctement une note sur la portée. Pour cela, vous déplacez la dernière note jaune dessinée par le programme (à un endroit toujours inexact) à l'aide des touches " ↑ " et " ↓ ". Quand vous pensez l'avoir bien située, vous

appuyez sur la barre d'espace. Si votre réponse est correcte, vous marquez un point. Dans le cas contraire, votre note est effacée tandis qu'un "zap" retentit et la note correcte est dessinée.

Le programme va alors vous faire deviner une troisième note, en vous faisant entendre à nouveau les deux notes qui figurent déjà sur la portée. A chaque fois, la note jouée se signale en "rougissant".

Le jeu se poursuit ainsi, de note en note, de devinette en devinette. Toutefois, pour que tout cela ne devienne pas trop lent, à partir de la sixième note, le programme se contente de vous faire entendre les cinq dernières notes seulement. Cela devrait vous fournir une préparation sonore suffisante pour trouver la dernière note entendue. Si ce n'était pas le cas, vous verrez au paragraphe 3, comment modifier cela.

Le programme vous propose 27 niveaux de jeu, grâce à deux paramètres que vous pouvez choisir:

a) Le domaine dans lequel vont évoluer les notes. Trois options possibles:

1 — de "do" à "sol": cela fait donc 5 notes en tout: do, ré, mi, fa et sol.

2 — de "do" à "do": cela correspond donc à 8 notes d'un octave.

3 — de "do" à "la": il faut comprendre du do d'un octave au la de l'octave supérieur. Cela correspond donc à 13 notes.

b) L'intervalle séparant deux notes successives. Il peut varier de un à neuf.

A titre d'exemple, si vous choisissez la valeur 1, la note émise après un mi ne pourra être que ré ou fa.

Par contre, si vous choisissez la valeur 3, la note émise après un mi pourra être: ré, do, fa ou sol.

Le programme affiche en permanence, le numéro de la note à deviner (nombre de notes déjà jouées), votre score (nombre de notes reconnues) et le meilleur score du niveau. (Voir photo 8 hors texte: *Vue de l'écran pendant le déroulement du jeu*).

2. *Le programme*

```
100 REM ********** DICTEE MUSICALE **********************
110 REM
120 GOSUB 9000
130 REPEAT
140 : GOSUB 8000
150 : NT=1: NN=1: CO=I2: GOSUB 3000: CD=1: GOSUB 5000
170 : NJ(1)=1
180 : FOR N=2 TO NX
190 :  GOSUB 2000
195 :  ND=N-5: IF ND<1 THEN ND=1
200 :  FOR NN=ND TO N-1
210 :   GOSUB 3000
220 :   NT=NJ(NN)
230 :   CD=3: CO=I3: GOSUB 5000: GOSUB 4000
240 :   CD=3: CO=I2: GOSUB 5000
250 :  NEXT NN
260 :  CX=1: IF RND(1)>0.5 THEN CX=-1
270 :  NT=NJ(N-1)+CX*INT(RND(1)*IM+1)
280 :  IF NT<1 OR NT>NM THEN 260
290 :  NJ(NN)=NT: NN=N
295 :  GOSUB 4000
300 :  GOSUB 6000
310 :  IF NT=NJ(N) THEN SC=SC+1: GOSUB 2000
315 :  IF NT<>NJ(N) THEN ZAP
320 :  CO=I2: NT=NJ(N): CD=1: GOSUB 5000
330 : NEXT N
340 : IF SC>MS(DM,IM) THEN MS(DM,IM)=SC
350 : FOR K=1 TO 22: PRINT CHR$(9);: NEXT K
360 : PRINT"rejouez vous (O/N)";
370 : GET R$: GET R$
380 UNTIL R$<>"O"
390 END
2000 REM ---- SP MISE A JOUR ET AFFICHAGE SCORE --------
2010 PRINT "  ";STR$(N-1);: IF N-1<10 THEN PRINT " ";
2020 PRINT "     ";STR$(SC);: IF SC<10 THEN PRINT " ";
2030 PRINT "    ";STR$(MS(DM,IM));CHR$(13);
2040 RETURN
3000 REM ---- SP CALCUL X ET YB POUR LA NOTE NUMERO NN ------
3010 YB=Y1: X=X1+(NN-1)*DX
3020 IF X>=X2-DX THEN YB=Y2: X=X-(X2-X1)
3030 RETURN
4000 REM ---- SP SON NOTE NT ----------------------
4020 MUSIC 1,OC(NT),NO(NT),0
```

```
4030 PLAY 1,0,1,1500: WAIT 40
4040 RETURN
5000 REM ---- SP AFFICHAGE, EFFACEMENT DE NOTE ---------
5010 Y=YB-(NT-1)*(DT+ET)/2+ET+1
5020 IF CD=1 THEN GOSUB5100: GOSUB5200: GOSUB5300: GOSUB5400:RETURN
5030 IF CD=2 THEN GOSUB 5100: GOSUB 5200: GOSUB 5300: RETURN
5040 IF CD=3 THEN GOSUB 5200: RETURN
5050 IF CD=4 THEN GOSUB 5500: RETURN
5060 RETURN
5100 REM
5110 REM CURSET X+6,Y-7,0: FILL 8,1,IG
5120 IF NT<>3 AND NT<>5 AND NT<>7 AND NT<>9 AND NT<>11 THEN RETURN
5130 CURSET X+6,Y-4,0: FILL 2,1,IG+128
5140 RETURN
5200 REM
5210 CURSET X-6,Y-7,0: FILL 8,1,CO
5220 IF NT<>3 AND NT<>5 AND NT<>7 AND NT<>9 AND NT<>11 THEN RETURN
5230 CURSET X-6,Y-4,0: FILL 2,1,CO+128
5240 RETURN
5300 REM
5310 CURSET X,Y-7,0: CHAR CN,0,1
5320 IF NT<>3 AND NT<>5 AND NT<>7 AND NT<>9 AND NT<>11 THEN RETURN
5340 RETURN
5400 REM
5410 IF NT<=7 THEN CURSET X,Y-4,1: DRAW 0,-20,1
5420 IF NT>7 THEN CURSET X+5,Y-4,1: DRAW 0,20,1
5430 RETURN
5500 REM
5510 CURSET X,Y-7,0: CHAR CN,0,0
5515 CURSET X-6,Y-7,0: FILL 8,1,IG
5520 IF NT<>3 AND NT<>5 AND NT<>7 AND NT<>9 AND NT<>11 THEN RETURN
5530 CURSET X-6,Y-4,0: FILL 2,1,IG+128
5540 CURSET X,Y-4,1: DRAW 6,0,1
5550 CURSET X,Y-3,1: DRAW 6,0,1
5560 RETURN
6000 REM ---- SP LECTURE NOTE PROPOSEE --------------
6010 GOSUB 3000
6020 CO=I2: NT=NJ(N-1)
6030 CD=2: GOSUB 5000
6040 DP=PEEK(#208): IF DP=#84 THEN 6110
6050 NE=NT+(DP=#9C)*(NT<NM)-(DP=#B4)*(NT>1)
6060 IF NE=NT THEN 6040
6070 CD=4: GOSUB 5000
6080 NT=NE: CO=I2: CD=2: GOSUB 5000
6090 GOTO 6040
6100 REM
6110 CD=4: GOSUB 5000
6120 RETURN
```

```
8000 REM ---- SP INITIALISATION D'UNE PARTIE ---------
8010 TEXT: CLS: PLOT 8,2," DICTEE MUSICALE"
8020 PLOT 3,7,"3 domaines possibles"
8028 PLOT 5,8," 1 : DO a SOL"
8040 PLOT 5,9," 2 : DO a DO"
8050 PLOT 5,10," 3 : DO a LA"
8060 PLOT 3,12,"quel domaine choisissez vous?"
8070 GET R$: IF R$<"1" OR R$>"3" THEN 8070
8080 PLOT 33,12,R$: DM=VAL(R$)
8090 PLOT 3,16,"intervalle maximum entre"
8100 PLOT 3,17,"deux notes (1 a 9)?"
8110 GET R$: IF R$<"1" OR R$>"9" THEN 8110
8120 PLOT 24,17,R$: IM=VAL(R$)
8130 WAIT 200
8140 CLS: PG=0: IG=7-PG: I2=3: I3=1
8150 HIRES: PAPER PG: INK IG
8160 YD=Y1: SC=0
8170 FOR I=1 TO 2
8180 : FOR Y=YD-DT-ET TO YD-5*(DT+ET) STEP -(DT+ET)
8185 :  FOR K=0 TO ET-1
8190 :  CURSET X0,Y-K,0: DRAW X2-X0,0,1
8195 :  NEXT K
8200 : NEXT Y
8210 : YD=Y2
8220 NEXT I
8230 PRINT: PRINT" NOTE  SCORE  M-SCORE";CHR$(13)
8240 NM=5: IF DM=2 THEN NM=8
8250 IF DM=3 THEN NM=13
8260 RETURN
9000 REM ---- INITIALISATIONS DU JEU --------------
9010 POKE #26A,10
9020 CN=64: AM=46080+8*CN
9030 DATA 30,63,63,63,63,63,63,30
9040 FOR I=0 TO 7
9050 : READ XX: POKE AM+I,XX
9060 NEXT I
9070 DT=8: ET=2: DX=12
9080 PA=4: IN=6: PAPER PA: INK IN
9090 X0=12: X1=24: X2=234
9100 Y1=88: Y2=168
9110 NX=2*INT((X2-X1)/DX)
9120 DIM NJ(NX),NO(13),OC(13)
9125 DIM MS(3,9)
9130 DATA 1,3,5,6,8,10,12,1,3,5,6,8,10
9140 FOR I=1 TO 13: READ NO(I): NEXT I
9150 DATA 3,3,3,3,3,3,3,4,4,4,4,4,4
9160 FOR I=1 TO 13: READ OC(I): NEXT I
9170 RETURN
```

3. *Si vous souhaitez personnaliser ce programme*

1. Pour que les notes soient jouées sur un autre octave

Modifiez les valeurs de l'instruction DATA de la ligne 9150, en y plaçant sept fois le numéro de l'octave voulu (de 0 à 5), suivi de six fois le numéro de l'octave suivant.

Exemples :

```
9150 DATA 2,2,2,2,2,2,2,3,3,3,3,3,3
```

jouera les notes un octave plus bas que le programme actuel.

```
9150 DATA 4,4,4,4,4,4,4,5,5,5,5,5,5
```

les jouera un octave plus haut.

2. Si vous trouvez que le jeu est trop lent ou trop rapide

Vous pouvez modifier la durée de chaque note, en remplaçant, en 4030, la valeur 40 par un nombre de votre choix. Par exemple :

```
4030 PLAY 1,0,1,1500: WAIT 10
```

jouera environ deux fois plus vite.

3. Pour modifier le nombre de notes jouées avant la nouvelle note à deviner

Remplacez, en 195, dans ND = N − 5, le nombre 5 par une valeur de votre choix.

Ainsi :

```
195 ND=N-1: IF ND<1 THEN ND=1
```

vous jouera une seule note avant la note à deviner.

```
195 ND=N-10: IF ND<1 THEN ND=1
```

vous en jouera dix (du moins à partir du moment où il y aura déjà dix notes de représentées).

Si vous souhaitez que toutes les notes soient rejouées avant la note à deviner, remplacez 195 par:

```
195 ND=1
```

4. *Description du programme*

a) Techniques utilisées, décrites en annexe

— Hasard.
— Création de caractères graphiques.
— Lecture rapide du clavier.
— Configuration clavier/écran.

b) Le programme principal (100-390)

120 appelle le sous-programme d'initialisation du jeu.

130 et 380 répètent les instructions 140 à 370 (déroulement d'une partie) jusqu'à ce que vous précisiez que vous ne souhaitez plus rejouer:

140 appelle le sous-programme d'initialisation d'une partie.

150-170 dessinent la première note et mettent son numéro (1) dans le tableau des notes déjà jouées.

180 et 330 répètent 33 fois les instructions 190 à 320 qui proposent, à chaque fois, une nouvelle note à découvrir:

190-250 font entendre les cinq dernières notes (ou toutes les notes lorsqu'il y en a moins de cinq) en les faisant "rougir".

260-290 choisissent au hasard une nouvelle note (en tenant compte des limitations imposées par votre choix de domaine et d'intervalle) et rangent sa valeur dans NJ.

295 fait entendre la note choisie.

300 appelle le sous-programme "lecture note proposée" qui place votre réponse dans NT.

310-315 analysent votre réponse ; actualisent le score si elle est exacte, font entendre un "zap" dans le cas contraire.

320 affiche définitivement la note correcte (celle qui vous avez proposée, correcte ou non, a été effacée par le sous-programme "lecture note").

340-370 actualisent le meilleur score et vous demandent si vous souhaitez rejouer.

c) Le sous-programme d'initialisation du jeu (9000-9170)

9010 supprime le bip du clavier et le curseur d'écran.

9020-9060 fabriquent le caractère graphique représentant une note.

9070 fixe la distance entre deux traits de portée, l'épaisseur du trait et la distance entre deux notes successives.

9080 fixent les couleurs de l'écran en mode TEXT.

9090-9100 déterminent les emplacements des deux portées.

9110 calcule le nombre total de notes qui seront présentées.

9120-9160 réservent les tableaux et fixent les numéros des notes et le numéro d'octave correspondant.

d) Le sous-programme d'initialisation d'une partie (8000-8260)

8010-8130 passent en mode "TEXT" et vous font choisir votre niveau de jeu (domaine, intervalle).

8140-8150 effacent l'écran et passent en mode "HIRES" (graphique haute résolution), en fixant les couleurs de ce mode.

8155 initialise le score.

8160 initialise le score.

8170-8220 tracent les deux portées.

8240-8250 déterminent la note la plus haute, compte-tenu de l'intervalle choisi.

e) Le sous-programme 3000-3030

Il calcule pour la note de rang NN :

— l'abscisse correspondante,
— l'ordonnée de la base de la portée correspondante.

f) Le sous-programme d'affichage et d'effacement de note (5000-5560)

Il réalise plusieurs fonctions déterminées par la valeur de la variable CD.

CD=1 : affichage d'une nouvelle note avec queue,

CD=2 : affichage d'une nouvelle note sans queue,

CD=3 : changement de couleur d'une note,

CD=4 : effacement d'une note (sans queue).

La hauteur de la note figure dans NT, son abscisse dans X. Le cas échéant, la couleur concernée figure dans CO.

5010 calcule l'ordonnée de la note.

5020-5050 réalisent chacune des fonctions par enchaînement de sous-programmes élémentaires :

5100-5140 rétablit la couleur de fond, à droite de la note.

5200-5240 fixe la couleur d'une note.

5300-5340 dessine une note.

5400-5430 dessine la queue.

5500-5560 efface une note.

g) Le sous-programme de lecture de la note proposée (6000-6120)

6010-6030 affichent une note provisoire (sans queue).

6040-6090 examinent le clavier et déplacent la note provisoire en fonction des touches enfoncées jusqu'à ce que vous appuyiez sur la barre d'espace.

6110 efface la note provisoire.

h) Le sous-programme son (4000-4040)

Il émet un son correspondant à la note de hauteur NT.

5. *Liste des variables*

CN	code ASCII du caractère représentant une note
DT	nombre de "points élémentaires" séparant deux lignes d'une portée
ET	épaisseur des traits des portées
DX	écart entre les abscisses de deux notes successives
PA	couleur de fond, en mode TEXT (choix des niveaux et bande inférieure de l'écran)
IN	couleur d'encre en mode TEXT
X0,X2	abscisses de début et de fin d'une portée
X1	abscisse de la première note
Y1,Y2	ordonnées de base des deux portées
NX	nombre total de notes
NJ(NX)	tableau contenant les hauteurs des notes déjà jouées
NO(13),OC(13)	tableaux contenant les codes et octaves des treize hauteurs de note possibles.
MS(3,9)	tableau contenant le meilleur score de chaque niveau
DM	domaine choisi (1 à 3)
IM	intervalle maximum choisi (1 à 9)
PG	couleur de fond utilisée en HIRES
IG	couleur d'encre utilisée pour le tracé de la portée
I2	couleur des notes
I3	couleur d'une note pendant qu'elle est "jouée"
NM	hauteur de la note la plus haute, compte tenu de l'intervalle choisi
NT	hauteur de note courante
NN	numéro de note courante
CO	couleur de note, utilisée par le sous-programme "affichage et effacement"
CD	précise au sous-programme "affichage et effacement" la fonction choisie (de 1 à 4) (voir description de ce sous-programme)

N	numéro de la dernière note proposée
YB	ordonnée de base de la portée correspondant à une note courante
X	abscisse d'une note courante
SC	score

Variables auxiliaires

AM XX I R$ Y YD
K CX

Annexe 1
Hasard

La fonction RND permet, lorsqu'on l'utilise avec un argument supérieur ou égal à un, d'obtenir un nombre tiré au hasard entre 0 (compris) et 1 (exclu). Bien qu'aucun de nos jeux ne soit ce que l'on a coutume d'appeler des "jeux de hasard", il se trouve quand même que le hasard y intervient souvent. C'est notamment le cas lorsqu'il faut choisir au hasard la position d'un objet sur l'écran, une lettre de l'alphabet, etc...

Dans tous les cas, le problème à résoudre est le suivant : choisir au hasard, une valeur entière comprise entre deux valeurs données p et n.

Commençons par le cas où p = 1, autrement dit, lorsque l'on désire une valeur entière comprise entre 1 et n (1 et n compris) :

Expressions	*Valeur correspondante*
RND(1)	entre 0 (compris) et 1 (exclu)
n * RND(1)	entre 0 (compris) et n (exclu)
n * RND(1) + 1	entre 1 (compris) et n+1 (exclu)
INT(n * RND(1) + 1)	entière entre 1 (compris) et n (compris)

Par exemple, l'expression :

```
INT(6*RND(1)+1)
```

fournira un nombre entier compris entre 1 et 6.

Voyons maintenant le cas plus général évoqué précédemment : obtenir un nombre entier compris entre p (compris) et n (compris) :

Expressions	*Valeur correspondante*
RND(1)	entre 0 (compris) et 1 (exclu)
(n−p+1) * RND(1)	entre 0 (compris) et n−p+1 (exclu)
p + (n−p+1) * RND(1)	entre p (compris) et n+1 (exclu)
INT(p+ (n−p+1) * RND(1)	entier entre p (compris) et n (compris)

La dernière expression peut également s'écrire :

```
p+INT((n−p+1)*RND(1))
```

Celle-ci est fréquemment utilisée dans nos programmes. Ainsi pour tirer dans X une coordonnée d'écran comprise entre deux limites X1 et X2, nous écrivons :

```
X=X1+INT(RND(1)*(X2−X1+1)
```

De même pour tirer dans C, le code ASCII d'une lettre comprise entre A (code 65) et Z (code 90) nous écrivons :

```
C=65+INT(RND(1)*26)
```

(puisque 90 − 65 + 1 = 26)

Annexe 2

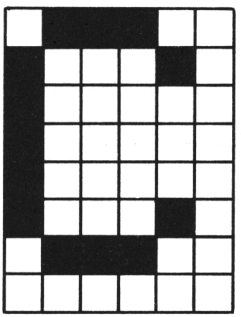
Création de caractères graphiques

Pour bien comprendre la manière de fabriquer des caractères graphiques, il est nécessaire de savoir comment l'ORIC réalise les caractères que vous voyez apparaître sur votre écran.

1. Comment l'ORIC fabrique-t-il les caractères?

Chaque caractère est affiché à l'écran sur un quadrillage de 8 × 6 cellules élémentaires. Chaque cellule peut être soit de la couleur de fond (qu'ici nous supposerons blanc) soit de la couleur d'encre (supposée ici noire). Voici, par exemple comment est représentée la lettre C:

Pour vous en convaincre, il vous suffit d'examiner votre écran de très près (ou de regarder notre dessin de loin !).



Comment l'ORIC conserve-t-il ce dessin en mémoire ? Il le code sur huit octets consécutifs représentant chacun une ligne du quadrillage. Dans chaque octet, on trouve le nombre binaire obtenu en considérant qu'une cellule blanche correspond à 0 et qu'une cellule noire correspond à 1. Ainsi, pour notre lettre C, nous obtenons :

Ligne	*Nombre binaire*	*Valeur décimale correspondante*
1	0 1 1 1 0 0	28
2	1 0 0 0 1 0	34
3	1 0 0 0 0 0	32
4	1 0 0 0 0 0	32
5	1 0 0 0 0 0	32
6	1 0 0 0 1 0	34
7	0 1 1 1 0 0	28
8	0 0 0 0 0 0	0

A quel emplacement de la mémoire se trouvent ces huit octets ? Les descriptions de chacun des 128 caractères (codes ASCII 0 à 127) sont rangées suivant l'ordre des codes ASCII, à partir de l'adresse 46080 (ou 13312 sur l'ORIC 16K). Autrement dit le caractère de code ASCII 0 est décrit par les octets d'adresse 46080, 46081, ..., 46087. Le caractère de code ASCII 1 est décrit par les octets d'adresse 46088, 46089, ..., 46095 etc...

D'une manière générale, la description du caractère de code x occupe huit octets consécutifs à partir de l'adresse:

```
46080+8*x
```

A titre d'exemple, vous pouvez retrouver les huit valeurs permettant de fabriquer la lettre C à l'aide de ce petit programme:

```
100 X=67
110 FOR I=0 TO 7
120       PRINT PEEK(46080+8*X+I)
130 NEXT I
```

2. *Comment créer vous-même de nouveaux caractères?*

Il vous suffit tout simplement de modifier, avec l'instruction POKE, le contenu des octets contenant la description de caractères. C'est cette méthode qui nous a permis de créer, dans la plupart des jeux, des graphismes tels que: balle, boule, champignon, missile, robot, soucoupe, etc... Voici comment procéder:

a) Dessiner votre motif dans un quadrillage 8 x 6 comme nous l'avons fait pour la lettre C.

b) Écrire la valeur binaire correspondant à chaque ligne, puis sa valeur décimale.

c) Choisir un caractère non utilisé dans votre programme (par exemple @ ou © ...) et regarder son code dans la table ASCII.

d) Entrer les huit valeurs définies en b, par POKE, aux adresses obtenues par la formule précédente.

Voici un sous-programme qui "redéfinit" le caractère dont le code ASCII est contenu dans la variable C, à l'aide de huit valeurs supposées placées en "DATA".

```
9500 REM ---- SP CREATION CAR GRAPHIQUES ----
9510 AM=46080+8*C
9520 FOR I=0 TO 7
9530 : READ XX: POKE AM+I,XX
9540 NEXT I
9550 RETURN
```

C'est là le sous-programme que vous retrouverez dans la plupart des jeux où l'on utilise des caractères graphiques.

Annexe 3
Lecture rapide du clavier

Les instructions usuelles de Basic ne sont pas adaptées à la conduite de mobiles par l'intermédiaire du clavier (déplacement d'une raquette, d'une voiture, d'une base de tir, ...). Dans nos programmes, nous avons employé, Lorsque cela le nécessitait, d'autres méthodes que nous allons décrire ici.

1. Introduction

En Basic, il existe trois manières de lire le clavier:

* **INPUT** lit une chaîne de caractères en l'affichant à l'écran en même temps que vous la composez. Son action se termine lorsque vous frappez la touche "return". Cette instruction qui "bloque" le programme tant que

vous n'avez pas fini de frapper votre message, est inutilisable pour commander des déplacements.

* **GET** attend qu'un caractère soit frappé et le range dans la variable suivant le mot GET. Là encore le programme est bloqué tant que vous ne tapez pas un caractère. C'est d'ailleurs cette instruction que nous avons employée, de préférence à INPUT lorsque le programme attendait une réponse brève (un chiffre, O/N, ...).

* **KEY$** est une fonction qui se contente d'examiner si une touche du clavier est enfoncée. Si oui, elle restitue le caractère correspondant, sinon la chaîne vide. En apparence elle devrait permettre de résoudre notre problème de commande de mobile puisque, contrairement aux deux instructions précédentes, elle ne bloque pas le programme. Malheureusement, une imperfection de taille apparaît. Pour vous en convaincre, tapez ce petit programme :

```
100 FOR I=1 TO 25
110 A$=KEY$: IF A$="  " THEN 110
120 PRINT A$
130 NEXT I
```

Lancez le (RUN). Tant que vous n'appuyez sur aucune touche, il ne se passe rien ; l'instruction 110 boucle sur elle-même. Si vous pressez, par exemple la touche Z, le programme vous écrit immédiatement Z. Mais si vous laissez cette touche enfoncée en permanence, vous découvrirez qu'il s'écoule un temps assez long avant l'apparition du second Z, puis les lettres suivantes arrivent ensuite à un rythme régulier et plus rapide.

En fait, vous obtenez le même phénomène que lorsque vous tapez plusieurs fois la même lettre sous l'effet de la "répétition automatique".

Nous n'entrerons pas dans le détail de fonctionnement de KEY$; nous retiendrons seulement qu'elle est tributaire du rythme de la répétition automatique. Elle est donc inutilisable pour ce qui nous préoccupe. Notez bien que l'on rencontre de nombreux jeux Basic (y compris sur cassette) qui se contentent de cette fonction. Le déplacement d'un objet à l'aide des touches fléchées est très désagréable puisqu'irrégulier. Si vous disposez déjà de tels jeux, il vous sera facile d'y adapter l'une des techniques que nous vous proposons.

2. *Une première méthode: Utilisation des adresses 768 et 735*

On constate que:

l'octet 768: prend la valeur 176 lorsqu'aucune touche n'est enfoncée et une valeur différente dans le cas contraire;

l'octet 735: prend la valeur:

136 quand ← est enfoncée
137 quand → est enfoncée
138 quand ↓ est enfoncée
139 quand ↑ est enfoncée
141 quand return est enfoncée
etc...

Cette valeur subsiste, même quand la touche correspondante a été relâchée, jusqu'à ce que vous pressiez une nouvelle touche. C'est pour cette raison que l'examen de l'octet 768 est, en général, nécessaire.

Voici, à titre d'exemple, un programme qui permet de déplacer horizontalement la lettre "A" à l'aide des touches "→" et "←":

```
100 X=10: Y=10
110 PLOT X,Y,"A"
120 IF PEEK(768)=176 THEN 120
130 CL=PEEK(735)-135
140 PLOT X,Y,32
150 X=X+(CL=1)-(CL=2)
160 GOTO 110
```

L'instruction 150 mérite quelques explications. Sur ORIC une expression telle que CL = 1 prend la valeur − 1 quand l'égalité mentionnée est vraie, la valeur 0 dans le cas contraire. Autrement dit X diminue de 1 quand CL vaut 1 (c'est-à-dire quand l'octet 735 vaut 136, ce qui correspond à "←") et X augmente de 1 quand CL vaut 2 ("→").

Si vous exécutez, ce petit programme, il est probable que vous amènerez votre lettre A sur l'un des bords de l'écran et que vous obtiendrez le message d'erreur:

```
ILLEGAL QUANTITY ERROR IN 110
```

En effet, nous n'imposons pas de limite à la valeur de X. Nous pouvons le faire, soit en ajoutant des instructions de test, soit simplement, en utilisant la technique déjà décrite en remplaçant la ligne 150 par:

```
150 X=X-(CL=1)*(X>1)+(CL=2)*(X<38)
```

Le principal mérite de cette première méthode est que les valeurs correspondant aux quatre touches fléchées sont consécutives (736 à 739). Cela peut être fort utile lorsque l'on souhaite, par exemple, sélectionner un élément d'un tableau en fonction de cette valeur (voir: Karting).

Une seconde méthode: Utilisation de l'adresse # 208 (soit 520 en décimal)

On constate que cet octet prend la valeur:

#38 lorsqu'aucune touche n'est enfoncée
#AC lorsque la touche ← est enfoncée
#BC lorsque la touche → est enfoncée
#B4 lorsque la touche ↓ est enfoncée
#9C lorsque la touche ↑ est enfoncée

Ici, dès que la touche est relâchée, l'octet #208 reprend la valeur #38. Par rapport à la méthode précédente, cette dernière présente l'avantage de ne nécessiter la consultation que d'un seul octet. Par contre, les valeurs obtenues ne sont plus consécutives.

Voici, à titre d'exemple, comment adapter cette méthode à notre programme de déplacement de lettre:

```
100 X=10: Y=10
110 PLOT X,Y,"A"
120 CL=PEEK(#208): IF CL=#38 THEN 120
140 PLOT X,Y,32
150 X=X-(CL=#AC)*(X>1)+(CL=#BC)*(X<38)
160 GOTO 110
```

D'une manière générale, c'est cette méthode que nous avons utilisée dans nos programmes à chaque fois que la précédente ne s'imposait pas.

Annexe 4
SCROLL

Certains micro-ordinateurs possèdent une instruction Basic réalisant le défilement d'écran (en anglais SCROLL). Sur votre ORIC, vous obtenez automatiquement un défilement (vers le haut), dès que l'écran est rempli. Si vous faites une liste d'un programme, vous avez un défilement quasi-continu.

Ce défilement vers le haut peut s'obtenir également manuellement en déplaçant le curseur jusqu'en bas de l'écran et en conservant la touche " ↓ " enfoncée. Vous pouvez, à l'inverse, obtenir un défilement vers le bas, en déplaçant le curseur jusqu'en haut de l'écran et en conservant la touche " ↑ " enfoncée.

Pour réaliser ce défilement par programme, il suffit de savoir que :

PRINT CHR$(10) ; réalise la même chose que l'appui sur la touche ↓
PRINT CHR$(11) ; réalise la même chose que l'appui sur la touche ↑

Bien entendu, au début du programme, il faut faire en sorte que le curseur soit bien positionné, suivant le cas :

— en haut de l'écran (ce qui est le cas après l'instruction CLS),

— en bas de l'écran. Il suffit "d'imprimer" suffisamment de caractères CHR$(11) ou même simplement d'imprimer 26 lignes blanches :

```
FOR I=1 TO 26: PRINT: NEXT I
```

L'avantage considérable de cette méthode réside dans la simulation d'un mouvement d'ensemble, sans qu'il soit nécessaire de programmer le déplacement de chacun des éléments. On l'emploie généralement lorsqu'un petit nombre d'objets doivent rester immobiles par rapport à l'ensemble. Dans ce cas, les objets fixes sont effacés avant le défilement pour être redessinés immédiatement après.

Annexe 5

Écriture directe en mémoire d'écran

1. *Introduction*

L'introduction PLOT ne peut être employée qu'avec des abscisses variant entre 0 et 38 et des ordonnées variant entre 0 et 26. Cela signifie qu'elle permet d'utiliser 39 colonnes et 27 lignes. Or, la totalité de l'écran comprend 40 colonnes et 28 lignes.

Effectivement, la première ligne de l'écran n'est pas accessible par PLOT. C'est d'ailleurs là que Basic place certains messages: CAPS, searching, loading, etc... Notez, au passage, que cette ligne n'est pas soumise au "défilement" d'écran (scroll); cela se remarque, notamment quand vous listez un programme.

Quand à la première colonne, elle est utilisée par Basic pour définir la couleur de chaque ligne de l'écran (PAPER et CLS agissent sur cette colonne). Par contre, la couleur d'encre est placée par Basic (INK) en seconde colonne, accessible à PLOT puisque correspondant à l'abscisse zéro.

Or, il est intéressant de pouvoir accéder à la première colonne lorsque, par exemple l'on désire obtenir plusieurs zones horizontales de couleurs différentes. Certes, l'attribut de couleur peut toujours être écrit par PLOT en seconde colonne (abscisse 0 pour PLOT) mais, dans ce cas, la première colonne a une couleur différente du reste de l'écran ; cela n'est pas toujours esthétique.

De même l'accès à la première ligne présente un grand intérêt lorsque l'on utilise une technique de "scroll" (défilement d'écran). En effet, c'est la seule ligne où l'on peut faire figurer des informations qui ne risquent pas de disparaître après défilement.

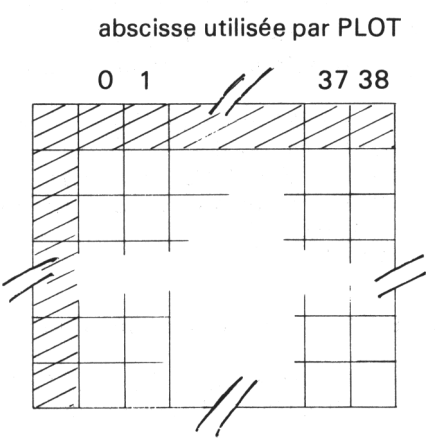
2. *La mémoire d'écran*

En fait la solution à ces problèmes devient extrêmement simple dès que l'on sait que l'image apparaissant à l'écran est stockée dans la mémoire de l'ORIC. Elle occupe les adresses 48000(#BB80 en hexa) à 49119 (#BFDF en hexa) sur l'ORIC 48K et les adresses 15232(#3B80) à 16351 (#38DF) sur l'ORIC 16K. Chaque octet correspond à une position écran et les informations sont stockées ligne par ligne.

Ainsi 48000 correspond à la première position de la première ligne de l'écran.

Voici un petit schéma récapitulatif (les hachures correspondent aux zones d'écran non accessibles par PLOT).

adresse mémoire 48 K	adresse mémoire 16 K	ordonnée utilisée par PLOT
48000	15232	
48040	15272	0
48080	15312	1
49040	16272	25
49080	16312	26



Pour écrire dans les zones hachurées, il vous suffit donc d'utiliser l'instruction POKE. Bien sûr, rien ne vous empêche d'employer POKE pour écrire dans n'importe quelle autre partie de l'écran.

3. Exemple : plusieurs couleurs de fond

Pour obtenir un fond jaune avec les trois premières lignes en rouge, nous pouvons procéder ainsi :

```
100 PAPER 3
110 FOR I=1 TO 3
120 : POKE 48000+40*I,17
130 NEXT I
```

4. Exemple : affichage en ligne supérieure

- Pour effacer cette ligne (ce qui fera du même coup disparaître le mot "CAPS"), vous pouvez utiliser ces instructions :

```
100 AM=48000
110 FOR I=0 TO 39
120 : POKE AM+I,32
130 NEXT I
```

- Pour écrire une chaîne dans cette ligne, il faudra l'écrire lettre par lettre. Voici un petit sous-programme qui affiche, à partir de la colonne IC de la ligne supérieure, la chaîne contenue dans CH$:

```
6000 REM affichage ligne supérieure
6010 AM=48000
6020 FOR I=1 TO LEN(CH$)
6030 : POKE AM+I+IC-1,ASC(MID$(CH$,I,1))
6040 NEXT I
6050 RETURN
```

Essayez-le en exécutant, en mode direct :

```
IC=1:CH$="BONJOUR":GOSUB 6000
IC=10:CH$="MADAME":GOSUB 6000
```

Annexe 6
Configuration clavier/écran

Vous savez que l'on peut supprimer le "bip" du clavier en tapant simplement "Controle F" (c'est-à-dire en maintenant la touche "CTRL" enfoncée tandis que l'on presse la touche F). Cette même action rétablit le "bip". On a un fonctionnement dit "à bascule" puisqu'une même commande fait basculer d'un état à l'autre.

De la même façon "Contrôle Q" fait apparaître ou disparaître le curseur de l'écran (carré clignotant).

Comment réaliser ces actions par programme ? Nous vous proposons deux solutions :

Solution 1 : Utiliser l'instruction PRINT

```
PRINT CHR$(6); a le même effet que "controle F"
PRINT CHR$(17); a le même effet que "controle Q"
```

Ainsi vous pouvez placer ces deux instructions au début d'un programme pour supprimer le bip et le curseur. Il faudra toutefois penser à les exécuter en fin de programme pour retrouver l'état initial.

Cette méthode présente un inconvénient dû à ce que les effets sont "à bascule". Par exemple, si votre programme est interrompu, vous serez dans l'état "bip et curseur supprimés". Si vous relancez alors le programme, les instructions PRINT... vous redonneront le bip et le curseur.

La solution suivante permet d'éviter cet inconvénient:

Solution 2:

L'état du clavier et de l'écran est conservé par ORIC dans l'octet d'adresse #26A. Voici sa signification, bit par bit:

Numero du bit	*Poids*	*Valeur standard*	*État correspondant à*	
			0	*1*
0	1	1	curseur supprimé	curseur visible
1	2	1	vidéo supprimée	vidéo active
2	4	0	—	—
3	8	0	clavier sonore (bip)	clavier muet (bip supprimé)
4	16	0	ESC actif	pas d'ESC
5	32	0	38 colonnes	40 colonnes
6	64	0	simple hauteur	double hauteur
7	128	0	—	—

Il faut noter que lorsque vous mettez votre ORIC sous tension, cet octet contient la valeur 3. Pour supprimer le bip du clavier, il faut mettre à 1 le bit 3 et pour supprimer le curseur, il faut mettre à 0 le bit 1. Il faut, bien entendu, laisser les autres bits inchangés. La valeur à placer (par POKE) est donc 10 (2+8). Pour cela, il suffira d'écrire:

```
POKE #26A,10
```

Si, à la fin du programme, vous souhaitez revenir à l'état initial, vous écrirez:

```
POKE #26A,3
```

L'avantage de cette méthode est que, quel que soit l'état initial, le résultat de POKE #26A,10 conduira à la suppression du bip et du curseur. Ce n'était pas le cas de la première méthode comme nous l'avons vu.

REMARQUE: Nous vous avons donné la signification de chacun des bits de l'octet #26A bien que seuls les bits 0 et 3 nous aient intéressés ici. Ce sont les seuls que nous avons été amenés à utiliser dans les programmes de jeux de ce manuel.

Annexe 7
Poignées de jeu

1. Introduction

Beaucoup des jeux proposés dans ce manuel peuvent utiliser une poignée de jeu (aucun ne nécessite deux poignées). Pour cela, il est nécessaire d'entrer en mémoire l'un des deux programmes proposés par la note accompagnant les poignées. Il s'agit de programmes Basic dont l'exécution provoque l'écriture d'instructions en langage machine dans un emplacement disponible de la mémoire. Ceci peut, éventuellement, n'être réalisé qu'une seule fois pour différents jeux. En effet, le chargement d'un programme de jeu (par CLOAD) ne détruira pas les instructions en langage machine. Seul le programme Basic ayant servi à les fabriquer sera effacé, ce qui n'a pas d'importance. Par contre, ces instructions disparaîtront (comme tout le reste de la mémoire) lorsque vous mettrez votre ORIC hors tension.

Nous vous conseillons de conserver ces deux programmes sur cassette afin d'éviter d'avoir à les taper plusieurs fois.

Les deux paragraphes suivants expliquent le fonctionnement des programmes proposés. Leur lecture n'est pas indispensable.

2. *Premier programme*

Nous le nommerons "MANETTE 1"

```
99 DATA #48
100 DATA #AD,#01,#03,#48
105 DATA #AD,#03,#03,#48,#A9,#C0
110 DATA #8D,#03,#03,#A9,#80,#8D,#0F,#03,#AD,#0F,#03,#8D,#00,#04
115 DATA #A9,#40,#8D,#0F,#03,#AD,#01,#03
120 DATA #8D,#01,#04,#68,#8D,#03,#03
125 DATA #68,#8D,#0F,#03
130 DATA #68,#4C,#03,#EC
200 MEM=#0402
210 READ DTA: POKE MEM,DTA
220 IF DTA<>#EC THEN MEM=MEM+1: GOTO 210
225 DOKE #0229,#402
230 PRINT PEEK(#400),PEEK(#401):GOTO230
```

Les instructions 200 à 220 placent, à partir de l'adresse mémoire #402, les instructions machine fournies par les "DATA" des lignes 100 à 130. L'instruction 225 permet à ce programme d'être exécuté en même temps que la routine de gestion d'interruptions.

Le rôle des instructions machines ainsi créées consiste à placer dans les octets #400 et #401 l'état de chacune des deux poignées (#400 correspondant à la poignée droite). La valeur de chaque bit est définie ainsi:

Numéro du bit	*Poids du bit*	*Signification*
0	1	gauche
1	2	droite
2	4	(inutilisé)
3	8	bas
4	16	haut
5	32	bouton de tir (l'un des deux)
6	64	poignée gauche
7	128	poignée droite

Le "poids du bit" correspond à la valeur obtenue lorsque ce bit est à un. Contrairement à ce que l'on pourrait croire, chaque bit prend la valeur 0 (et non 1) quand l'action correspondante est effective et la valeur 1 dans le cas contraire. Par exemple, si le bouton de tir de la poignée droite est enfoncé, en même temps qu'elle est en position "haut", l'octet #400 contiendra le nombre binaire :

N° du bit :	7	6	5	4	3	2	1	0
Valeur :	0	1	0	0	1	1	1	1

ce qui correspond à la valeur 64 + 8 + 4 + 2 + 1 = 79.

Lorsque la poignée droite est en position de repos, l'octet #400 contient : 64 + 32 + 16 + 8 + 4 + 2 + 1 = 127. De même, quand la poignée gauche est en position de repos, l'octet #401 contient : 128 + 32 + 16 + 8 + 4 + 2 + 1 = 191.

Notez bien que deux bits de position (haut, bas, gauche, droite) peuvent être simultanément à zéro. C'est ce qui nous permettra le déplacement "en diagonale", chose que les touches du clavier ne permettent pas.

Comment utilisons-nous ce programme

- Pour savoir si la poignée droite est au repos, nous comparons à 191, la valeur de l'octet d'adresse #400.
- Pour savoir si un tir est demandé, nous examinons le bit 5 de cet octet en employant la fonction AND. Ainsi, ces instructions :

```
CL=PEEK( #400)
IF((CL AND 32)=0) THEN...
```

exécutent l'action figurée par ... lorsque l'un des boutons de tir est enfoncé, quelle que soit la position de la poignée. Si nous avions simplement comparé la valeur de l'octet #400 à 95 (64+16+8+4+2+1), nous n'aurions trouvé l'égalité que si l'un des boutons était enfoncé alors que la manette était en position centrale. Cette façon de faire empêcherait de tirer pendant un déplacement.

- Pour réaliser un déplacement, nous procédons ainsi : si X et Y désignent les coordonnées d'un mobile à un instant donné, ses nouvelles coordonnées XN,YN sont définies par :

```
CL=PEEK(#400)
XN=X+((CL AND 1)=0)-((CL AND 2)=0)
YN=Y+((CL AND 16)=0)-((CL AND 8)=0)
```

Là encore, nous avons considéré individuellement chacun des bits 0,1,3 et 4 de l'octet #400. En procédant ainsi, le déplacement en diagonale devient possible.

REMARQUE: L'instruction 210 écrit, en permanence, le contenu des octets #400 et #401. Lorsque vous exécutez le programme MANETTE 1, vous pouvez ainsi examiner les différentes valeurs obtenues en agissant sur vos poignées.

3. *Deuxième programme*

Nous le nommerons " MANETTE 2 "

```
100 DATA #48,#AD,1,3,#48,#AD,3,3,#48,#A9,#C0,#8D,#3,3,#A9,#80
110 DATA #8D,#F,3,#20,#32,#04,#AD,0,4,#8D,1,4,#A9,#40,#8D,#F,3
120 DATA #20,#32,4,#68,#8D,3,3,#68,#8D,#F,3,#68,#4C,3,#EC
130 DATA #A9,0,#8D,0,4,#AD,#F,3,#49,#3B,#0A,#0A,#0A,#2E,0,4,#0A
140 DATA #2E,0,4,#0A,#2E,0,4,#0A,#0A,#2E,0,4,#0A,#2E,0,4,#60
150 FOR AD=#402 TO #453: READ DTA: POKE AD,DTA: NEXT
160 IF DTA<>#60 THEN STOP
200 DOKE #229,#402
210 PRINT PEEK(#400),PEEK(#401)
220 GOTO 210
```

Le mécanisme est le même que celui employé dans le programme précédent. La différence réside simplement dans les valeurs placées dans les octets #400 et #401. Seuls, les bits 0 à 4 sont utilisés, comme ceci:

Numéro du bit	*Poids*	*Signification*
0	1	gauche
1	2	droite
2	4	bas
3	8	haut
4	16	tir

Cette fois, un bit donné prend la valeur 1 quand l'action correspondante est effective.

Ce résultat est obtenu grâce à quelques instructions machine supplémentaires. L'examen de la valeur des octets #400 et #401 s'en trouve ainsi simplifié.

Comment utilisons-nous ce programme

- Pour savoir si la poignée droite est au repos, il suffit de comparer à zéro la valeur de l'octet #400.
- Pour savoir si un tir est demandé, nous examinons le bit 4. Ainsi ces instructions :

```
CL=PEEK(#400)
IF(CL AND 16) THEN...
```

exécutent l'action représentée par... lorsque l'un des boutons de tir est enfoncé.

- Pour réaliser un déplacement, on peut procéder ainsi :

```
CL=PEEK(#400)
XN=X-(CL AND 1)+(CL AND 2)
YN=Y-(CL AND 8)+(CL AND 4)
```

mais on peut aussi profiter du fait que les valeurs sont "groupées", et écrire :

```
ON  (CL AND 16) GOTO X,Y,Z
ON  (CL AND 16) GOSUB X, Y, Z
```

x,y,z,... représentant des numéros de ligne.

Notez bien que l'expression (CL AND 16) prend une valeur comprise entre 0 et 10 ; cependant elle ne prend jamais la valeur 3 ni la valeur 7.

Voici un schéma récapitulant les valeurs possibles:

```
                HAUT
              9  8  10
GAUCHE        1  0   2   DROITE
              5  4   6
                BAS
```

- Pour dessiner un mobile qui change d'aspect suivant le sens de déplacement, on peut créer un tableau C de dimension 10, contenant les codes des caractères utilisés pour chacune des directions:

  C(1) correspond à un déplacement vers la gauche,
  C(2) correspond à un déplacement vers la droite,
  C(3) n'est jamais utilisé,
  ...
  C(5) correspond à un déplacement vers le bas et vers la gauche (en diagonale), etc...

Pour afficher le bon caractère en X,Y il suffit d'écrire:

```
CL=PEEK(#400)
IF CL <> 0 THEN I=CL AND 16:PLOT X,Y,C(I)
```

C'est cette méthode que nous employons dans "karting".

REMARQUE: Là encore, l'instruction 230 écrit en permanence les valeurs des octets #400 et #401.

4. *Comment employer une poignée dans les jeux proposés*

a) Examinez la rubrique "pour utiliser une poignée de jeu" du paragraphe 3 du chapitre relatif au jeu concerné. Celle-ci vous précise:

— le programme à employer (MANETTE1 ou MANETTE2),
— les modifications à apporter au programme de jeu.

b) Chargez le programme demandé (MANETTE1 ou MANETTE2) et exécutez-le. Vous devez voir défiler:

197 et 127 pour MANETTE 1
0 et 0 pour MANETTE 2

Vérifiez que la poignée droite est correctement branchée: en agissant sur elle, la première valeur doit évoluer. (Si c'est la seconde valeur qui change, il vous suffit de modifier le branchement).

Interrompez l'exécution, par "Controle C".

c) Chargez maintenant votre programme de jeu. Effectuez les modifications proposées, à moins que vous n'ayez déjà sauvegardé une version "poignée de jeu".

d) Lancez votre programme par RUN.

REMARQUE: La façon dont les poignées sont branchées sur l'ORIC fait que leur utilisation modifie ou interrompt les sons générés par le haut-parleur.

Achevé d'imprimer
par **Cid** *éditions*

Dépôt légal : mars 1984
N° d'Éditeur : 4067

Découvrez les excellentes possibilités graphiques et sonores de l'ORIC avec ces vingt jeux plus passionnants les uns que les autres : venez piloter un bombardier, une voiture de course, une soucoupe volante ; essayez de détruire les envahisseurs ou d'alunir en douceur ; distrayez-vous au tir aux ballons ou à la chasse aux champignons ; détendez-vous avec le « mur de briques » ; entraînez votre mémoire avec « phosphore » ; exercez vos facultés de raisonnement avec le master-mind géant...

Affirmez votre personnalité en adaptant chaque jeu en fonction de vos désirs : choisissez pour cela parmi les nombreuses modifications qui vous sont proposées (y compris l'utilisation d'une poignée de jeu) et qui ne nécessitent aucune pratique de la programmation.

Etendez vos connaissances du BASIC et de l'ORIC, tout en vous amusant, grâce aux explications détaillées qui accompagnent chaque programme de jeu.

Donnez libre cours à votre imagination en appliquant les techniques décrites ici à la programmation des scénarios de jeux qui vous tiennent à cœur.

EYROLLES

8210

FAITES VOS JEUX AVEC ORIC

C. DELANNOY