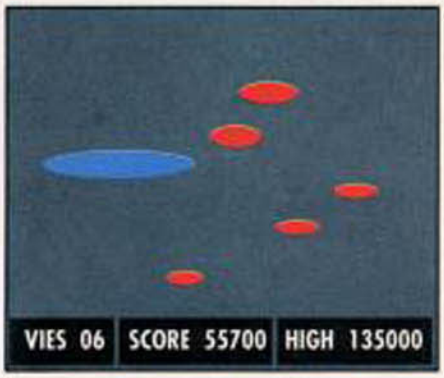
J'APPRENDS

PAR FRANÇOIS LE GRIGUER

JOYSTICK ATTACK



GENERIQUE
SCENARIO
PAGE DES SCORES
TABLEAU DE BORD
BONUS
AUGMENTATION
DU NOMBRE DES ADVERSAIRES
INVERSION FUITE/ATTAQUE
GESTION DE L'ENGIN DU JOUEUR
GESTION DES ADVERSAIRES
CONTROLE DES COLLISIONS
FIN DE LA PARTIE (GAME OVER)
SONS
CODE ASCII ET
CARACTERES REDEFINIS

TU RÊVES DE RÉALISER DES JEUX. AVEC DES IDÉES PLEIN LA TÊTE ET QUELQUES BOUQUINS TU AS ESSAYÉ. JE TE PROPOSE UNE INITIATION RÉELLE, PAS SIMPLEMENT UN TEXTE ET UN LISTING, MAIS UNE VÉRITABLE APPROCHE DE LA CRÉATION, SIMPLE ET CLAIRE, QUI SEMAINE APRÈS SEMAINE TE FERA VIVRE À FOND LA FORMIDABLE AVENTURE DE L'INFORMATIQUE EN CRÉANT, TOI-MÊME, DÈS AUJOURD'HUI, JOYSTICK ATTACK ET BIEN D'AUTRES JEUX PAR LA SUITE.

IL N'EST PAS QUESTION DE RIVALISER AVEC LES JEUX DU COMMERCE, POUR DÉBUTER IL FAUT ÊTRE MODESTE ET BIEN PROGRAMMER EN BASIC, C'EST DÉJÀ PAS MAL. MAIS PLUS TARD, ON POURRAIT ABORDER L'ASSEMBLEUR.

MAINTENANT, SI TU ES DÉJÀ TRÈS FORT, TU PEUX EN PROFITER POUR CONTRÔLER RÉELLEMENT TES CAPACITÉS EN AMÉLIORANT LES SÉQUENCES ET, POURQUOI PAS EN APPRENDRE UN PEU PLUS.

CHAQUE SEMAINE C'EST UN NOUVEAU MODULE DU JEU AVEC UN PLAN :

- L'EXPOSÉ RAPIDE DU MODULE À RÉALISER
- LA LISTE DES FICHIERS NÉCESSAIRES À LA SÉQUENCE
- LE DÉROULEMENT LOGIQUE DU TRAITEMENT AVEC UN PETIT ORDINOGRAMME

TU DISPOSES ALORS DE TOUS LES ÉLÉMENTS POUR ESSAYER TOUT SEUL, D'ÉCRIRE ET DE TESTER TON PROGRAMME AVANT LA PROCHAINE PARUTION DE JOYSTICK HEBDO. LA SEMAINE SUIVANTE, TU AURAS LE LISTING DES INSTRUCTIONS AVEC TOUTES LES EXPLICATIONS ET TU CONTRÔLES OU TU CORRIGES CE QUE TU AS PRÉPARÉ. PRESQUE UN COURS PAR CORRESPONDANCE, SUPER NON ?

JE RÉSERVE AUSSI UNE PETITE PLACE POUR UN LEXIQUE DES TERMES ANGLAIS ET LES BASES DE L'INFORMATIQUE ET, SURTOUT, POUR LE COURRIER. JE COMPTE BEAUCOUP SUR TES IDÉES, TES SUGGESTIONS, TES CRITIQUES OU TES QUESTIONS SUR UN POINT MAL COMPRIS. TU M'ÉCRIS, JE TE RÉPONDS ET TOUT LE MONDE EN PROFITE.

ENCORE UNE PETITE PRÉCISION, CETTE INITIATION EST PRÉVUE POUR LES AMSTRAD CPC. SI L'ON VEUT ÊTRE PRÉCIS DANS TOUTES LES EXPLICATIONS, IL FAUT CHOISIR UN MATÉRIEL UNIQUE. PAR CONTRE TOUTE LA PARTIE IDÉE OU LOGIQUE DE PROGRAMMATION RESTE ÉVIDEMMENT GÉNÉRALE À TOUS LES MATÉRIELS.

INUTILE DE PERDRE PLUS DE TEMPS, TU ES PRÊT ? ON ENCHAINE TOUT DE SUITE AVEC LA PREMIÈRE SÉQUENCE DE TON JEU D'ARCADE : JOYSTICK ATTACK

UN GENERIQUE ORIGINAL

Ton générique doit être à l'image de ton jeu, c'est à dire animé et dynamique, tout en restant simple à réaliser. Pour le côté simple n'imaginons pas de dessins ou de caractères redéfinis. Utilisons directement les symboles existants sur l'AMSTRAD. Par contre il faut chercher des idées pour avoir quand même un générique original.
Imagine que les coordonnées X et Y de chaque caractère soient définies de manière aléatoire pour aller, ensuite, rejoindre progressivement sa place dans le titre, en changeant de couleur à chaque déplacement. Ca te donne ta première page d'écran :

JOYSTICK ATTACK

Une petite attente en fin d'affichage pour envoyer la deuxième page avec ton nom Toujours pour sortir de l'ordinaire à peu de frais, la première ligne s'affiche à gauche de l'écran, la seconde à droite et la troisième à gauche. Chaque ligne glisse ensuite vers le centre de l'écran.
Voilà pour les idées. Tu peux bien sûr en avoir d'autres.

UNE REALISATION DE TON NOM

LES FICHIERS

Pour suivre notre plan et prendre de bonnes habitudes on en arrive à la définition des fichiers nécessaires au traitement. La question à se poser c'est de quelles informations a-t-on besoin pour réaliser ce générique selon nos idées. Il s'agit d'un petit inventaire, sans rien oublier :

- les 14 caractères du titre
- la position finale de chaque caractère, c'est à dire les coordonnées X et Y dans le titre en fin d'affichage à l'écran.
- le numéro d'encre de chaque caractère pour permettre le changement individuel de couleur
- les coordonnées X et Y en cours pour assurer le déplacement depuis l'entrée jusqu'à la position finale de chaque caractère

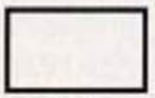
Pour la deuxième page d'écran les besoins se limitent à la valeur de X en cours et au nombre de déplacements pour chaque ligne.
On se fait une petite récapitulation de nos fichiers en leur donnant un nom :

FICHIERS GENERIQUE

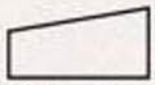
NOM	LONGEUR	TYPE	OBSERVATIONS
titre	14	Alpha	"Joystick attack"
xfin	"	Numérique	Position finale
yfin	"	"	"
coul	"	"	N° encre en cours
posx	"	"	Position en cours
posy	"	"	"

UN DESSIN NOMME ORDINOGRAMME

Après les idées et les fichiers il s'agit de tracer un schéma du traitement : l'ordinogramme. Pas de phrases mais une suite logique d'opérations avec quelques commentaires permettant de programmer directement parce que toutes les situations sont prévues, tous les tests en place avec leurs branchements. En pratique avec beaucoup d'expérience on limite l'écriture détaillée d'ordinogrammes aux cas particulièrement compliqués mais, pour toi actuellement, il s'agit d'une très bonne discipline qui te fera gagner du temps dans la programmation et la mise au point. On va utiliser six symboles (il y en a beaucoup plus)

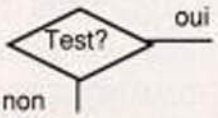


Traitement

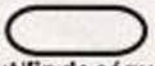


Saisie au clavier

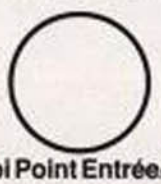
Affichage a l'écran



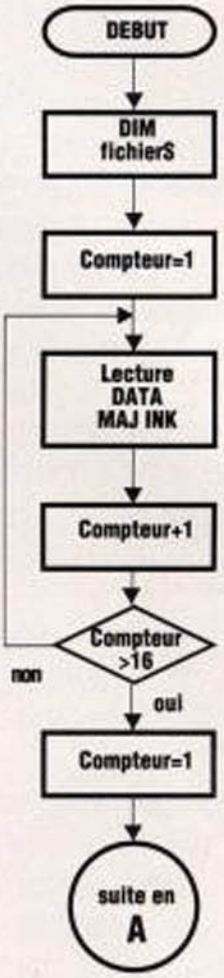
Branchement selon résultat du test: oui/ non



Début/fin de séquence



Renvoi Point Entrée/Sortie



INITIALISATION DES FICHIERS

SI TU AS DES PROBLÈMES ÉCRIS MOI

JOYSTICK HEBDO
J'APPRENDS
177, RUE St HONORE
75001 PARIS

L'initialisation concerne tous les fichiers du générique. Déclarer la DIMension, c'est-à-dire la longueur pour les 6 fichiers.

Initialisation des 16 encres utilisées - compteur positionné à 1

Boucle de programme exécutée 16 fois.

Lecture de la couleur donnée (Data) et mise à jour de l'encre (INK)

Le compteur est incrémenté de 1

Si le compteur est plus grand que 16 la boucle est terminée (16 passages)

Cette boucle est identique à la précédente à part le nombre de passages prévu cette fois pour 14 qui correspond à la longueur des fichiers (voir tableau récapitulatif)

Boucle à prévoir pour les fichiers :

titre
xfin
yfin

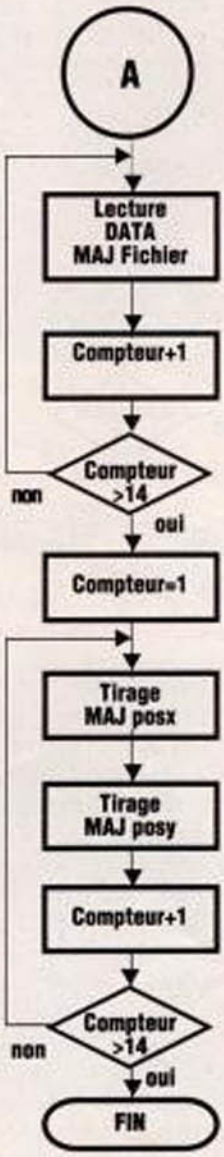
Initialiser le fichier COUL avec les encres de 1 à 14. La valeur du compteur sert directement de donnée.

Initialisation des fichiers POSX et POSY par le tirage d'un nombre aléatoire.

Tirage d'un nombre aléatoire qui doit être compris
entre 1 et 20 pour X
entre 1 et 25 pour Y.

Le générique étant en mode Ø, tu peux utiliser 16 couleurs simultanées avec 20 caractères sur 25 lignes.

Fin de l'initialisation. Les fichiers sont prêts pour l'animation du titre.



POUR QUELQUES DEFINITIONS DE PLUS...

Avant d'aborder les ordinogrammes je te propose quelques explications complémentaires.
Pour exploiter un fichier la première étape consiste à le déclarer avec son nom, en y ajoutant le symbole $ s'il s'agit de caractères et non de valeurs, et avec sa longueur. Ceci permet à l'ordinateur de reconnaitre le fichier et, de prévoir, en mémoire, la place nécessaire. L'instruction BASIC à utiliser est DIM (comme DIMension) et, voici comment s'effectue la déclaration de nos fichiers:

```
DIM titre$(14)
DIM xfin(14)
DIM yfin(14)
DIM coul(14)
DIM posx(14)
DIM posy(14)
```

Ces instructions figureront dans les listings, la semaine prochaine, avec tous les commentaires.

DONNEES A ENTRER DANS LES FICHIERS

NUMÉRO D'ORDRE DANS LE FICHIER														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
titre	J	O	Y	S	T	I	C	K	A	T	T	A	C	K
xfin	3	5	7	9	11	13	15	17	5	7	9	11	13	15
yfin	11	11	11	11	11	11	11	11	15	15	15	15	15	15
coul	1	2	3	4	5	6	7	8	9	10	11	12	13	14
posx	valeurs définies par un nombre tiré au hasard (1 20)													
posy	valeurs définies par un nombre tiré au hasard (1 25)													

Toutes les données de **titre, xfin** et **yfin** sont intégrées au programme sous la forme de lignes DATA (donnée), comme, par exemple, la ligne DATA pour le fichier titre:

```
DATA J,O,Y,S,T,I,C,K,A,T,T,A,C,K
```

L'initialisation consiste à transférer les DATA vers les Fichiers.
Une dernière définition indispensable à la compréhension des ordinogrammes:

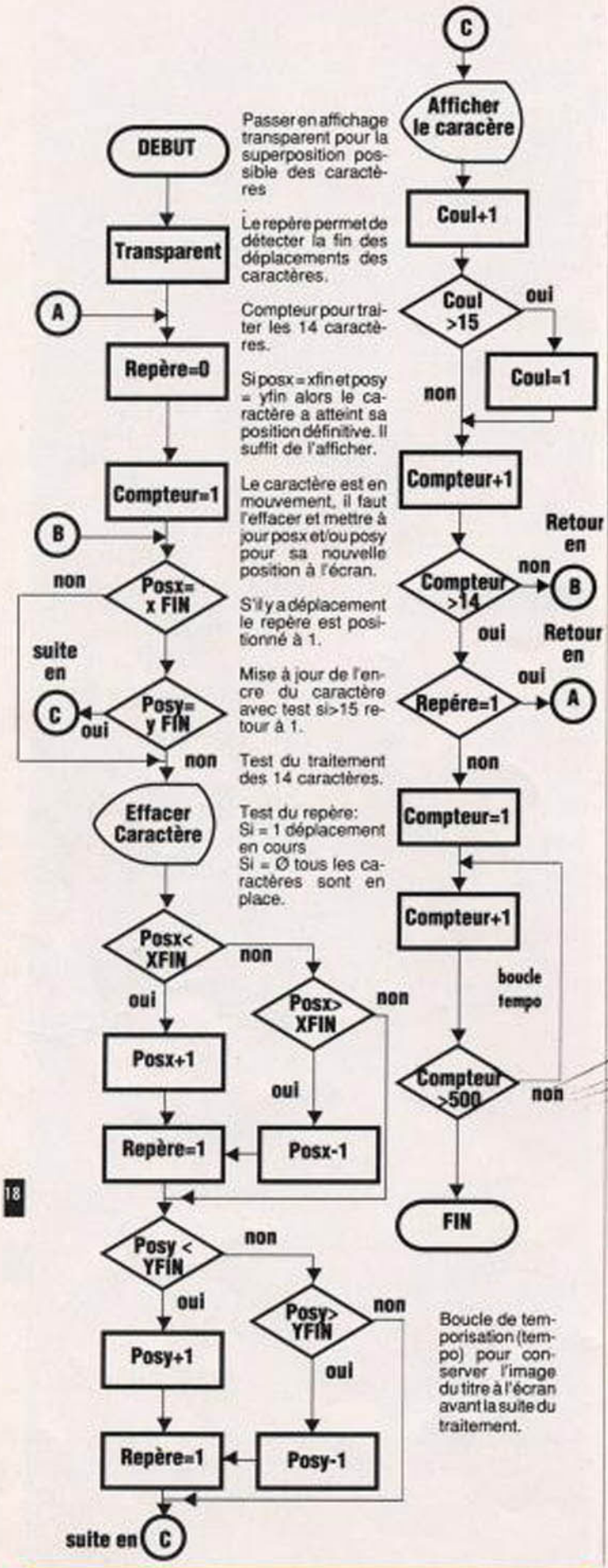
QU'EST-CE QU'UNE BOUCLE DE PROGRAMME ?

Il s'agit de pouvoir répéter plusieurs fois un ensemble d'instructions. Le procédé est simple:
- créer un compteur avec une valeur d'origine au dé but de la séquence à répéter
- en fin de séquence prévoir l'incrémentation (augmentation) du compteur
- tester le compteur pour définir si le nombre prévu de répétitions est réalisé

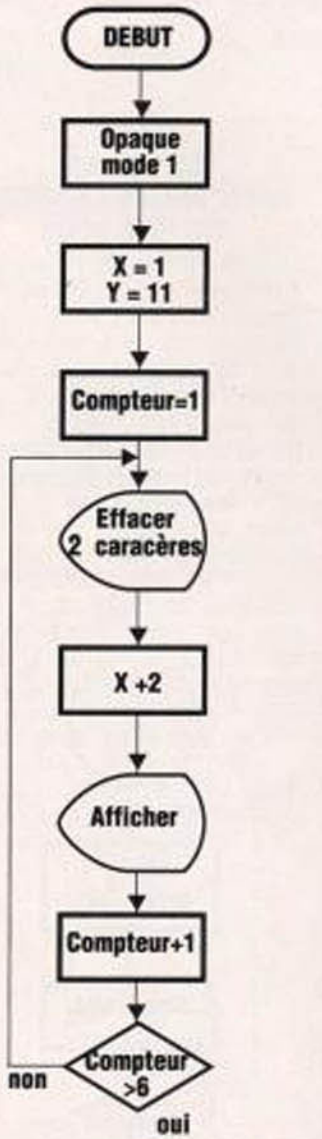
si NON nouvelle exécution de la séquence
si OUI passage à la suite du programme

Tu verras, par la suite, d'autres possibilités concernant le compteur de boucle.

ANIMATION DU TITRE



2éme PAGE GENERIQUE



Affichage normal. Mode 1 (4 couleurs, 40 caractères sur 25 lignes)

1ère boucle pour l'affichage de «une réalisation».

Le déplacement étant de 2, il suffit d'effacer les 2 derniers caractères.

Mise à jour de X (déplacement de gauche à droite)

«une réalisation»

Deux autres boucles sont à prévoir pour :
- affichage «de» X=39 Y=13 8 déplacements de la droite vers la gauche (X=X-2)
- affichage «ton nom» X=2 Y=16 5 déplacements de la gauche vers la droite (X=X+2)

En fin d'affichage, comme pour le titre, tu prévois une boucle de tempo, valeur du compteur 1 à 500.

Si tu connais un peu la programmation en BASIC tu as tous les éléments pour écrire ton générique. Pour toi qui débute, je crois que le plus important c'est de bien comprendre comment passer des idées à la réalisation concrète d'un programme, savoir définir les fichiers nécessaires et leur structure, pouvoir construire un ordinogramme.
En comprenant bien tout celà, tu verras la semaine prochaine qu'écrire les instructions ce n'est pas compliqué.

Bien Joystiquement votre

François LE GRIGUER

LA SEMAINE PROCHAINE

- Le listing BASIC du générique commenté et expliqué
- Le scénario
- Les nouveaux modules à réaliser :
  - la page de score
  - les initialisations de début de partie

PAR FRANÇOIS LE GRIGUER

J'APPRENDS

CETTE SEMAINE APRÈS UN COUP D'OEIL SUR LE SCENARIO ET, QUELQUES DÉFINITIONS, TU PLONGES DANS LE LISTING DU GÉNÉRIQUE AVEC TOUTES LES EXPLICATIONS, ENSUITE TU PRÉPARES LA PAGE DE SCORE. UN SCÉNARIO D'ARCADE CETTE DÉFINITION DU JEU, AVEC TOUTES SES PARTICULARITÉS, INTERVIENT ÉVIDEMMENT DÈS LE DÉBUT MAIS, LA SEMAINE DERNIÈRE NOUS AVIONS DÉJÀ UN PROGRAMME TRÈS CHARGÉ ET, IL VALAIT MIEUX PASSER DIRECTEMENT À LA RÉALISATION. UN ENGIN QUE TU DÉPLACES DANS HUIT DIRECTIONS DOIT ÉVITER OU, AU CONTRAIRE, DÉTRUIRE DES ADVERSAIRES EN LES REJOIGNANT. LES DEUX PÉRIODES, POURSUIVRE OU ÊTRE POURSUIVI, MATÉRIALISÉES PAR UNE COULEUR DIFFÉRENTE DU FOND, SONT ALÉATOIRES ET LE NOMBRE D'ADVERSAIRES AUGMENTE AVEC LE TEMPS. LE BONUS, AVEC UNE VALEUR BASÉE SUR LE NOMBRE D'ADVERSAIRES, INTERVIENT SUIVANT UNE PÉRIODE DE TEMPS SANS PERTE. ENFIN LES POINTS MARQUÉS À CHAQUE DESTRUCTION SONT MULTIPLIÉS APRÈS UN BONUS ET, JUSQU'À LA PROCHAINE ERREUR. LA PARTIE S'ACHÈVE AVEC L'ÉPUISEMENT DES VIES ET L'AFFICHAGE DU TRADITIONNEL «GAME OVER» ! UN PETIT TABLEAU DE BORD EN BAS D'ÉCRAN DONNE LE NOMBRE DE VIES, LE SCORE EN COURS ET LE HIGH SCORE. DES SONS VIENDRONT SOUTENIR L'ACTION. EN PLUS DU GÉNÉRIQUE QUE TU CONNAIS, UNE PAGE DE SCORE AVEC L'ENTRÉE DE TON NOM COMPLÈTE LE SCÉNARIO SIMPLE À METTRE EN OEUVRE, INTÉRESSANT À JOUER, AVEC UN BON NOMBRE DE PROBLÈMES À RÉSOUDRE POUR ATTEINDRE NOTRE OBJECTIF : TE PERFECTIONNER EN PROGRAMMATION.

JOYSTICK ATTACK

UN SCENARIO D'ARCADE

A propos du listing BASIC
Seuls les mots-clé (termes) BASIC sont en majuscules. Les minuscules ou les valeurs numériques correspondent aux constantes, variables et fichiers définis par le programmeur (il s'agit, là, d'une convention entre nous).

• J'ai, volontairement, multiplié les lignes de certaines parties du programme pour plus de clarté dans les explications mais ceci coûte plus de place mémoire et, davantage de temps à l'exécution.

• Le listing représente la rédaction, dans un langage interprété par l'ordinateur, de nos ordinogrammes de la semaine dernière, alors reprends les pour mieux suivre.

• Ne confondons pas le O de l'alphabet avec le chiffre Ø et respecte les espaces.

• Certains listings comporteront des ruptures dans la numérotation des lignes pour l'insertion ultérieure de nouvelles lignes (ne pas renuméroter).

• Toutes les lignes qui nécessitent des explications particulières sont reprises à part avec le n° de la ligne et tous les commentaires.

UN PETIT RAPPEL DES MANIPULATIONS

• Pour entrer (saisir) un listing :
Mets-toi en minuscules (les mots clé seront reconnus et automatiquement édités en majuscules lors d'un LIST)
Compose :
AUTO n, 10
en remplaçant n par le 1er numéro de ligne à entrer. Cette fonction te donne la numérotation des lignes avec une progression de 10 à chaque fois que tu appuies sur ENTER (à la fin de chaque ligne BASIC).

• Pour interrompre la saisie :
appuie sur ESC
• Pour sauvegarder le programme sur K7 ou disquette
compose : SAVE "ATTACK" ENTER
à faire impérativement à la fin de la saisie de ton listing.
• Pou r modifier une ligne
compose : EDIT n ENTER
n = le numéro de ligne. Tu déplaces le curseur dans les 4 directions pour atteindre la zone à modifier. Tu supprimes avec CLEAR ou tu ajoutes en composant. Les modifications terminées appuie sur ENTER.
• Pour charger le programme en mémoire : LOAD " ATTACK

• Pour exécuter un programme en mémoire RUN
• Pour charger et exécuter le programme RUN " ATTACK
• Pour lister le prgramme LIST interruption par ESC
ou LIST 100 - (liste à partir de la ligne 100)
ou LIST 100 - 150 (liste de la ligne 100 à la ligne 150)

```
1Ø REM**** JOYSTICK ATTACK ***********
2Ø REM
3Ø REM ::::::::::::::::::::::::::::::::
4Ø REM ::::::::::::::::::::::::::::::::
5Ø REM     PROGRAMME PRINCIPAL
6Ø REM ::::::::::::::::::::::::::::::::
7Ø REM ::::::::::::::::::::::::::::::::
8Ø CLS
9Ø ' INITIALISATION DES FICHIERS
1ØØ GOSUB 36Ø
11Ø ' 1 ere PARTIE DU GENERIQUE
12Ø GOSUB 83Ø
13Ø ' 2eme PARTIE DU GENERIQUE
14Ø GOSUB 1Ø7Ø
15Ø RUN
29Ø '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
3ØØ '      S E Q U E N C E S
31Ø '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
32Ø '*********************************
33Ø ' INITIALISATION DES FICHIERS GENERIQUE
34Ø '*********************************
35Ø '
36Ø nb=14
37Ø DIM xfin(nb)
38Ø DIM yfin(nb)
39Ø DIM coul(nb)
4ØØ DIM posx(nb)
41Ø DIM posy(nb)
42Ø DIM titre$(nb)
43Ø ' ********** DATA GENERIQUE ****************
44Ø '  couleurs
45Ø DATA 13,4,8,11,14,15,16,17,18,19,2Ø,21,22,23,24,25
46Ø '  titre
47Ø DATA J,O,Y,S,T,I,C,K,A,T,T,A,C,K
48Ø '  xfin
49Ø DATA 3,5,7,9,11,13,15,17,5,7,9,11,13,15
5ØØ '  yfin
51Ø DATA 11,11,11,11,11,11,11,11,15,15,15,15,15,15
52Ø '*********************************
53Ø FOR j=Ø TO 15
54Ø READ a
55Ø INK j,a
56Ø NEXT j
57Ø '
58Ø FOR j=1 TO nb
59Ø READ titre$(j)
6ØØ NEXT j
61Ø '
62Ø FOR j=1 TO nb:READ xfin(j):NEXT
63Ø FOR j=1 TO nb:READ yfin(j):NEXT
64Ø mini=1:maxi=2Ø
65Ø FOR j=1 TO nb
66Ø GOSUB 77Ø
67Ø posx(j)=nombre
68Ø NEXT
69Ø maxi=25
7ØØ FOR j=1 TO nb:GOSUB 77Ø:posy(j)=nombre:NEXT
71Ø coul=1
72Ø FOR j=1 TO nb:coul(j)=coul:coul=coul+1
73Ø NEXT
74Ø RETURN
75Ø '*********************************
76Ø ' SOUS-PROGR ALEATOIRE
77Ø RANDOMIZE TIME
78Ø nombre=INT(RND*(maxi-mini+1))+mini
79Ø RETURN
```


LE GENERIQUE EN INSTRUCTIONS

10 Les lignes 10 à 70 sont des commentaires mot clé REM abréviation de REMARK. Des commentaires qui servent au programmeur mais restent ignorés lors du traitement.
50 Pour avoir un programme bien structuré, dans lequel tu peux revenir facilement des mois plus tard, il faut beaucoup d'ordre : un programme principal, des séquences et des sous-programmes bien séparés.
80 Effacement de l'écran
90 l'apostrophe remplace le REM (plus clair et plus simple)
100 GO = aller. SUB = sous-programme. GOSUB 360 = se brancher dans un sous-programme à la ligne 360
150 Relancer le traitement
300 Tu vas découvrir ici chaque séquence. Actuellement il n'y en a que trois mais le nombre augmentera avec l'avancement du programme.
330 Initialisation des fichiers telle que nous l'avons prévue dans notre ordinogramme de la semaine dernière
360 déclaration d'une constante portant le nom de «n b» avec une valeur de 14.
370 DIM : Dimensionner un fichier (voir tableau récapitulafit des fichiers). DIM x fin (nb) dimensionne le fichier x fin pour 14 (nb = 14). Il s'agit de la valeur x des 14 caractères dans le titre en fin d'affichage. Une petite particularité avec le symbole $ après «titre» qui indique qu'il s'agit d'un fichier prévu pour des informations alphanumériques (texte) et non plus de valeurs numériques.
450 DATA = Donnée. Ce sont toutes les données nécessaires à l'initialisation des fichiers avec la virgule comme séparateur.
520 Initialisation des fichiers (mise en place des données dans les fichiers)
530 Boucle de programme pour lire les 16 numéros de couleur en DATA et les charger dans les 16 numéros d'encre (INK 0 à INK 15). Déclaration de la variable «j» (valeur de 0 à 15).
540 Lecture en «a» du numéro de couleur
550 Mise à jour de INK j, valeur de a
560 fin de boucle j=j +1. Si j>15 suite du traitement à la ligne 570 sinon retour en 540
580 Boucle pour charger dans le fichier «titre $» les 14 caractères en DATA. nb = 14, donc la variable «j» prendra les valeurs de 1 à 14.
620 Cette boucle pour initialiser «x fin» est écrite en une seule ligne avec le séparateur «:» entre chaque instruction. Les précédentes boucles tenaient sur plusieurs lignes pour plus de clarté dans les explications.
640 Pour le fichier «pos x» il s'agit de valeurs aléatoires comprises entre 1 et 20
650 Début de boucle
660 Branchement dans un sous-programme à la ligne 770 pour la définition du nombre tiré au hasard.
670 Maj (mise à jour) de «pos x» selon le nombre aléatoire
690 Pour le fichier «posy» la valeur aléatoire doit être comprise entre 1 et 25.
710 Boucle de programme pour initialiser le fichier «coul», c-a-d le numéro d'INK de chaque caractère
740 Fin de la séquence. L'accès s'est fait depuis la ligne 100 (GOSUB 360). RETURN provoque le retour à la ligne 110 (instruction suivante)
760 Ce sous-programme va être appelé depuis différents points du traitement pour la définition d'un nombre au hasard (aléatoire) dont on donne les limites : «mini» et «maxi».
770 Génération de nombres au hasard à partir de la valeur de l'horloge interne
780 INT arrondit au plus petit nombre entier RND donne un nombre au hasard à l'intérieur d'une série.

```
8ØØ '*********************************
81Ø 'GENERIQUE 1ere partie deplacement des
      caracteres du titre
82Ø '*********************************
83Ø PRINT CHR$(22)+CHR$(1)
84Ø MODE Ø
85Ø flag=Ø
86Ø FOR j=1 TO nb
87Ø IF posx(j)=xfin(j) AND posy(j)=yfin(j) THEN 95Ø
88Ø PEN Ø
89Ø LOCATE posx(j),posy(j)
9ØØ PRINT CHR$(143)
91Ø IF posx(j)<xfin(j) THEN posx(j)=posx(j)+1:flag=1
     :GOTO 93Ø
92Ø IF posx(j)>xfin(j) THEN posx(j)=posx(j)-1:flag=1
93Ø IF posy(j)<yfin(j) THEN posy(j)=posy(j)+1:flag=1
     .GOTO 95Ø
94Ø IF posy(j)>yfin(j) THEN posy(j)=posy(j)-1:flag=1
95Ø PEN coul(j)
96Ø LOCATE posx(j),posy(j)
97Ø PRINT titre$(j)
98Ø coul(j)=coul(j)+1
99Ø IF coul(j)>15 THEN coul(j)=1
1ØØØ NEXT J
1Ø1Ø IF flag=1 THEN 85Ø
1Ø2Ø FOR j=1 TO 5ØØ:NEXT
1Ø3Ø RETURN
1Ø4Ø '*********************************
1Ø5Ø 'GENERIQUE 2eme partie "une realisation de.....
1Ø6Ø '*********************************
1Ø7Ø MODE 1
1Ø8Ø PRINT CHR$(22)+CHR$(Ø)
1Ø9Ø y=11:x=1
11ØØ FOR j=Ø TO 5
111Ø LOCATE x,y
112Ø PEN Ø
113Ø PRINT" "
114Ø x=x+2:LOCATE x,y
115Ø PEN 1
116Ø PRINT"une  realisation"
117Ø NEXT
118Ø x1=39:FOR j=Ø TO 8:LOCATE x1,y+2:PEN Ø
     :PRINT "  ":LOCATE x1-2,y+2:PEN 2
     :PRINT"de":x1=x1-2:NEXT
119Ø x=2:FOR j=Ø TO 4:LOCATE x,y+5:PEN Ø
     :PRINT "  ":LOCATE x+2,y+5:PEN 3
     :PRINT".....":x=x+2:NEXT
12ØØ FOR J=1 TO 5ØØ:NEXT
121Ø RETURN
122Ø '*********************************
123Ø '  FIN DU GENERIQUE
124Ø '*********************************
```

Observation : Compte tenu de notre mise en page, nous avons été amené à mettre les instructions des lignes (91Ø, 93Ø, 118Ø, 119Ø) sur plusieurs lignes. Lorsque vous les taperez sur votre écran, ecrivez les sur une seule ligne, sans laisser d'espace à notre coupure.

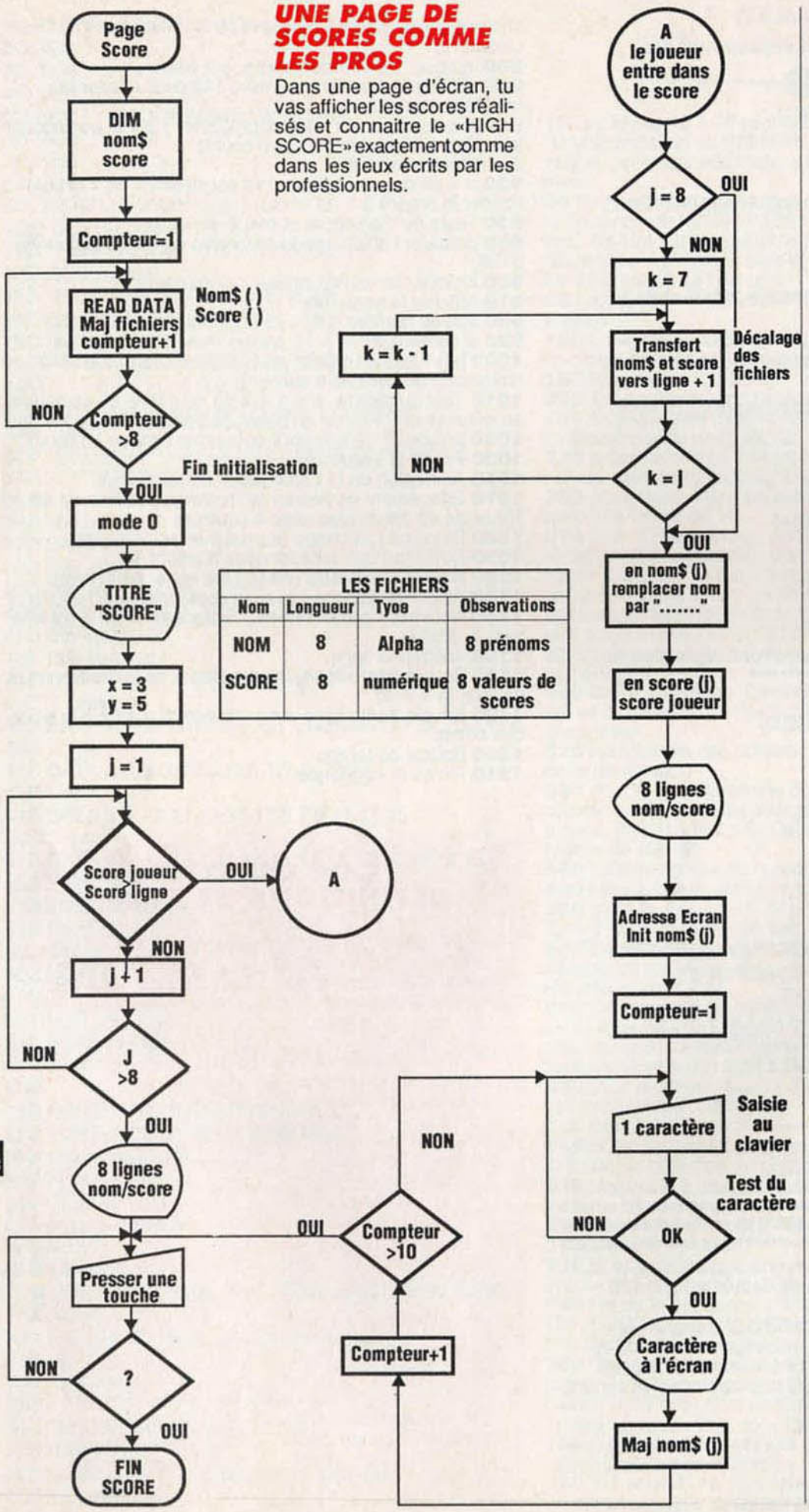
810 nouvelle séquence appelée depuis la ligne 120 (GOSUB 830)
830 instruction pour obtenir l'affichage transparent
840 effacement de l'écran et affichage possible de 25 lignes de 20 caractères avec 16 couleurs simultanées.
850 flag = drapeau ou repère d'animation des 14 caractères du titre
870 IF = si. THEN = alors. AND = et. si la valeur de x en cours (pos x) = la valeur finale x (x fin) et la valeur de y en cours (pos y) = la valeur finale y (y fin) alors se brancher à la ligne 950 (caractère en place).
880 les conditions ne sont pas remplies. Préparer l'affichage dans la même INK que le fond (effacement du caractère)
890 indiquer les coordonnées x, y à l'écran
900 afficher le caractère numéro 143 pour effacer la totalité du caractère
910 si x en cours < x final alors ajouter 1 à x et positionner le repère à 1 (déplacement en cours). Branchement en 930
920 si x en cours > x final alors soustraire 1 de x et positionner le repère à 1
930 Tests de y en cours et maj éventuelle
950 préparer l'affichage du caractère suivant son numéro d'INK
960 Indiquer les coordonnées x, y en cours
970 Afficher le caractère
980 Maj du numéro d'INK (changement de couleurs)
990 si numéro d'INK > 15 (valeur maxi 15) n° INK =1
1000 j=j+1. si j > 14 suite en 1010 sinon retour en 870 (animation du caractère suivant)
1010 Test du repère. si = 1 il reste au moins un caractère en mouvement. Retour en début de séquence en ligne 850
1020 Boucle de tempo pour conserver l'image à l'écran.
1030 Fin de la séquence
1050 Animation de la 2ème partie du générique
1070 Effacement de l'écran et affichage possible de 25 lignes de 40 caractères avec 4 couleurs
1080 Retour à l'affichage normal (non transparent)
1090 Définition des coordonnées d'affichage
1100 Boucle d'animation de la 1ère ligne. 6 passages.
1130 Afficher 2 espaces avec la couleur du fond = Effacer
1140 Nouvelles coordonnées. Déplacement de la gauche vers la droite
1160 Afficher la ligne
1180 Boucle 2ème ligne. Déplacement de la droite vers la gauche (x = x -2)
1190 Boucle 3ème ligne avec ton nom à mettre à la place des points
1200 Boucle de tempo
1210 Fin de la séquence

POUR QUELQUES DÉFINITIONS DE PLUS...

Avant d'aborder notre premier listing quelques définitions ou commentaires s'imposent.
Un **mot clé BASIC** est un terme du langage correspondant à un ordre (instruction) à exécuter. Dans le listing pour obtenir un maximum de clarté seuls les mots-clé figurent en majuscules.
Une **constante**, définie par le programmeur, avec un nom et une valeur, reste inchangée pendant toute son utilisation.
Une **variable** se différencie d'une constante par sa valeur qui va évoluer au cours du traitement.
Un **séparateur** est un symbole reconnu par le BASIC et utilisé pour séparer les instructions (:), les DATA (,) ou une succession de PRINT (;).
Une **ligne BASIC** s'identifie par son numéro, peut contenir de nombreuses instructions et représenter plusieurs lignes d'écran.
Le mode d'affichage utilisé dans ce programme sera le **mode caractère** : 25 lignes de 40 caractères en mode 1 ou de 20 caractères en mode 0. Les coordonnées x et y ont pour origine le coin haut gauche de l'écran (x = 1 à 20 ou 1 à 40 et y = 1 à 25). L'affichage à l'écran s'effectue à l'aide de 2 instructions : LOCATE x, y pour indiquer où commence l'affichage et PRINT pour dire ce qu'il faut afficher à l'écran.

UNE PAGE DE SCORES COMME LES PROS

Dans une page d'écran, tu vas afficher les scores réalisés et connaitre le «HIGH SCORE» exactement comme dans les jeux écrits par les professionnels.



LES FICHIERS			
Nom	Longueur	Tyoe	Observations
NOM	8	Alpha	8 prénoms
SCORE	8	numérique	8 valeurs de scores

Prévoyons huit lignes, c-a-d, huit prénoms et huit scores classés par ordre décroissant. Le programme contient les lignes d'origine avec des prénoms et des scores quelconque. Ensuite le joueur va entrer ou non, à chaque fin de partie, selon le score qu'il vient de réaliser. La seule difficulté dans ce module consiste à décaler les lignes. La saisie du prénom va, elle aussi, demander un peu d'attention.

Tu abordes un nouvel ordinogramme et, n'oublies surtout pas de bien comprendre le fonctionnement en suivant chaque test et ses branchements. Les instructions du programme reprendront très fidèlement le même cheminement :

- L'initialisation des fichiers
- Comparaison du score joueur avec le contenu du fichier score pour définir si le joueur entre dans le tableau. Si oui décalage du tableau et saisie du nom du joueur.
- Affichage de l'ensemble du tableau (8 lignes nom/score).

Je crois que ta semaine est bien remplie avec le listing du générique à bien assimiler et la page de score à préparer. Mais tu vois que finalement l'informatique par petites doses, passe très bien avec l'ordre et la logique comme clés de la réussite.

Bien joystiquement vôtre

François LE GRIGUER

LA SEMAINE PROCHAINE

le listing BASIC de la page de score avec, bien sûr, toutes les explications
la suite du jeu :
les constantes, les variables et les bonus
l'animation de l'engin du joueur
l'augmentation du nombre d'adversaires
l'inversion fuite/attaque des adversaires
l'initialisation du tableau de bord

PAR FRANCOIS LE GROLIER

J'APPRENDS

AUJOURD'HUI TU ENTRES VRAIMENT DANS L'ACTION AVEC LE DEPLACEMENT DU JOUEUR. LES ORDRES QUE TU DONNES AU JOYSTICK SE TRADUISENT PAR UNE EVOLUTION A L'ECRAN ET LA SEMAINE PROCHAINE LES ADVERSAIRES FONCTIONNERONT EGALEMENT. N'OUBLIE SURTOUT PAS QUE, LE PROGRAMME BASIC, EST LE REFLET EXACT DES ORDINOGRAMMES QUI REPRESENTENT LA LOGIQUE DU TRAITEMENT. LE BASIC EST UN LANGAGE PARMI BEAUCOUP D'AUTRES, AVEC SES MOTS-CLE ET SES REGLES D'ECRITURE, TRES BIEN ADAPTES AUX DEBUTS EN INFORMATIQUE.

UN VRAI TABLEAU DE BORD

Ça ne coûte pas cher d'avoir un vrai tableau de bord avec le nombre de vies, le score et le score à battre (HIGH). Tu le places en bas d'écran en ligne 25 avec une couleur de fond différente.

VIES 2Ø SCORE Ø HIGH 5Ø7ØØ.

En début de partie affichage:
- du mot «vies
- de la variable vies (valeur d'initialisation 20)
- du mot «score»
- de la variable score (valeur d'initialisation 0) en prévoyant la place pour 6 caractères
- du mot «HIGH»
- du contenu de la première ligne du fichier score (score (1)) (score le plus élevé).

En cours de partie 2 zones seulement seront réaffichées selon les gains ou pertes du joueur (2 sous-programmes) :
- la variable vies
- la variable score.

JOYSTICK ATTACK

DES CONSTANTES ET DES VARIABLES...

Ce module va être répété à chaque début de partie, il est capital de le définir avec précision pour que le jeu puisse fonctionner correctement et corresponde au scénario. Ici pas d'ordinogramme mais deux tableaux descriptifs.

Constantes à déclarer

nom	valeur	observations
bordf	15	Couleur flash BORDER (perte d'une vie)
fondØ	13	INK du fond attaque adversaires
fond 1	11	INK du fond fuite adversaires
point	1ØØ	valeur des points (normal)
pointm	3ØØ	valeur des points (mutlplié)
bonus	1ØØØ	valeur de base du Bonus
engjou$	CHR$ (231)	caractère représentant l'engin du joueur
engz$	" "	caractère pour l'effacement engin du joueur (Espace)
adv$	CHR$ (232)	caractère représentant un adversaire
advz$	" "	caractère pour l'effacement d'un adversaire

Variables à déclarer

nom	valeur	observations
xjou	4	coordonnées engin joueur
yjou	aléatoire	" " mini=3 maxi=2Ø
vies	2Ø	nombres de vies
score	Ø	score en cours
adv	Ø	nombre d'adversaires en cours
adv max	1	nombre maximum d'adversaires
v point	point	valeur en cours des points (normal ou multiplié)
tbonus	15ØØ	délai pour prochain bonus
pladv	1ØØØ	délai pour augmenter le nombre d'adversaires
code	1	1=fuite adversaires Ø=attaque adversaires

LA PAGE DE SCORE EN INSTRUCTIONS

```
15Ø ' INITIALISATION PAGE DE SCORE
16Ø GOSUB 128Ø
17Ø RUN
124Ø '*********************************
125Ø '
126Ø '  PAGE DE SCORE
127Ø '*********************************
128Ø DIM nom$(8)
129Ø DIM score(8)
13ØØ DIM advers(5,2)
131Ø DATA
JOYSTICK,FRANCK,PAUL,DENIS,SOPHIE,YVAN,MAGALIE,ALPHA
132Ø DATA
5Ø7ØØ,3Ø1ØØ,152ØØ,1Ø1ØØ,73ØØ,69ØØ,51ØØ,45ØØ
133Ø FOR j=1 TO 8:READ nom$(j):NEXT
134Ø FOR J=1 TO 8:READ score(j):NEXT
135Ø '
136Ø MODE Ø:PEN 1:LOCATE 5,2:PRINT "S C O R E "
137Ø x=3:y=5
138Ø FOR j=1 TO 8:IF score>score(j) THEN 142Ø
139Ø NEXT
14ØØ GOSUB 162Ø
141Ø GOTO 161Ø
142Ø '
143Ø '
144Ø '   decaler les lignes de score
145Ø '
146Ø '
147Ø IF j=8 THEN 151Ø
148Ø FOR k=7 TO j STEP -1
149Ø nom$(k+1)=nom$(k):score(k+1)=score(k)
15ØØ NEXT k
151Ø nom$(j)="....."
152Ø score(j)=score
153Ø GOSUB 162Ø
154Ø LOCATE x,5+(j-1)*2:nom$(j)=""
155Ø FOR k=1 TO 1Ø
156Ø n$=INKEY$:IF n$="" THEN 156Ø
157Ø n=ASC(n$):IF n<32 OR n>122 THEN 156Ø
158Ø PRINT n$;
159Ø nom$(j)=nom$(j)+n$
16ØØ NEXT k
161Ø IF INKEY$="" THEN 161Ø ELSE RETURN
162Ø FOR m=1 TO 8
163Ø LOCATE x,y
164Ø PRINT nom$(m)
165Ø LOCATE 13,y
166Ø IF score(m)>99999 THEN LOCATE x+9,y
167Ø IF score(m)<1ØØØØ THEN LOCATE x+11,y
168Ø IF score(m)<1ØØØ THEN LOCATE x+12,y
169Ø PRINT score(m)
17ØØ y=y+2
171Ø NEXT m
172Ø RETURN
173Ø '*********************************
```

150 Ces lignes s'ajoutent dans le programme principal
1280 Dimensionnement des fichiers nom$ et score prévus pour 8 lignes (8 noms avec leurs scores)
1300 tu prévois déjà le fichier des adversaires. 5 adversaires et 2 zones (x et y) on en reparlera.
1310 Prénoms d'origine. Tu peux les remplacer suivant ton inspiration.
1320 Idem pour les scores
1330 Boucles d'initialisation des fichiers nom$ et score. Lecture DATA et transfert en fichier
1360 Affichage du titre de la page.
1370 Déclaration de x et y pour l'affichage des lignes
1380 Boucle pour rechercher si le score du joueur est > que l'un des scores du tableau. Si oui branchement en 1420.
1400 Branchement dans le sous-programme d'affichage de la page
1410 suite en 1610
1420 Le joueur entre dans les scores
1470 Si l'entrée est en 8ème ligne il n'y a pas de décalage à effectuer
1480 Boucle de décalage. La ligne 7 passe en 8, la 6 en 7 etc.:. jusqu'à la ligne d'entrée du joueur contenu en j.
1490 Le contenu du fichier nom$ (K) passe en nom$ (K + 1) idem pour score
1500 le compteur K est décrémenté de 1 (-1) par STEP -1 de la ligne 1480
1510 le contenu de nom$ (j) est remplacé par des points pour matérialiser la zone de composition du nom du joueur
1520 mise en fichier score du score du joueur
1530 affichage de la page score par sous-programme
1540 indiquer les coordonnées écran pour la saisie du nom y = 5 + (numéro de ligne (en j) -1) * 2 (2 interlignes).
1550 Boucle de saisie de 10 caractères (nom du joueur)
1560 n$ contient le caractère composé. Sans composition retour en 1560
1570 Affiche du caractère
1590 Mise en fichier nom$ du caractère
1610 Attente d'une touche pour sortir de la sèquence
1620 Sous-programme pour l'affichage des 8 lignes nom/score
1640 Affichage du nom
1660 Modification de la position d'affichage si le score = 6 caractères
1670 idem si le score = 4 caractères
1680 idem si le score = 3 caractères pour conserver l'alignement de l'affichage
1690 Affichage du score
1700 Maj de y (2 interlignes)
1720 Fin du sous-programme.

A PROPOS DES ADVERSAIRES

Le fichier «adv» contient pour chaque adversaire ses coordonnées x, y. Pour différencier ceux qui sont actifs des autres on décide que si x = Ø l'adversaire est inactif. Donc, tu dois prévoir la remise à zéro de ce fichier. Un autre domaine qui demande quelques explications c'est la définition de délais avant qu'une séquence ne soit exécutée. Deux ordres BASIC utilisables : AFTER (après) et EVERY (chaque) accompagnés du délai en 1/50eme de seconde, du numéro de chronomètre (Ø a 3) et de l'adresse de la séquence à exécuter.

Exemple : AFTER 400,Ø GOSUB 311Ø

ce qui va générer après 400/50eme de seconde, au chrono Ø, l'interruption du programme pour exécuter la séquence en ligne 311Ø.

Assignation des chronomètres	
N° chrono	Utilisation
Ø	fuite/attaque des adversaires
1	augmentation du nombre d'adversaires
2	bonus

PLUS DE PIMENT AVEC UN BONUS

Dans les constantes et variables tu as déclaré la valeur de base bonus = 1ØØØ et le délai d'exécution tbonus = 15ØØ. Il s'agit donc d'un sous-programme qui fonctionne automatiquement lorsque le délai est atteint sur le chronomètre numéro 2. Le délai définit une période de temps pendant laquelle le joueur ne doit pas perdre de vies. Ce contrôle s'effectuera dans la séquence de gestion des collisions.

Opérations à prévoir lors de l'exécution du bonus :

- Augmenter le prochain délai du bonus de 100
- Initialiser le chronomètre numéro 2 (AFTER)
- Afficher au milieu de l'écran: BONUS (valeur)
- Ajouter la valeur du bonus au score et afficher le nouveau score
- Prévoir une boucle de tempo (temps d'affichage)
- Effacer «BONUS.....»
- Modifier la variable vpoint qui prend la valeur de pointm.

Les points marqués sont multipliés jusqu'à la prochaine perte.

DES ADVERSAIRES PLUS NOMBREUX

Le nombre d'adversaires maximum à l'écran se définit par la variable advmax initialisée à 1 en début de partie. Le délai pour l'augmentation du nombre en pladv déclenche à intervalles réguliers le sous-programme advmax + 1 avec une limite que tu fixes à 5 pour des raisons de rapidité de traitement. L'initialisation des nouveaux adversaires est prévue dans une séquence à part «gestion des adversaires».

LA FUITE ET L'ATTAQUE

Encore un sous-programme, déclenché à intervalles réguliers, par le chronomètre numéro 0 pour inverser la situation fuite ou attaque des adversaires.

si = 1 modifier les variables code = Ø et INK Ø,fond Ø

si = Ø modifier les variables code = 1 et INK Ø,fond 1.

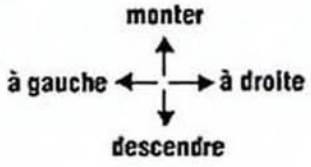
Le changement de INK Ø donne une couleur de fond différente, indication pour le joueur du nouvel état. Tu dois, toujours dans le sous-programme, réinitialiser le nouveau délai par AFTER. Le temps, pour être variable, peut, par exemple, se définir ainsi : 1ØØ + (yjou + xjou) * 4. Les variables yjou et xjou prennent des valeurs très diverses et donnent au calcul un résultat aléatoire.

TU DEPLACES TON ENGIN DANS 8 DIRECTIONS

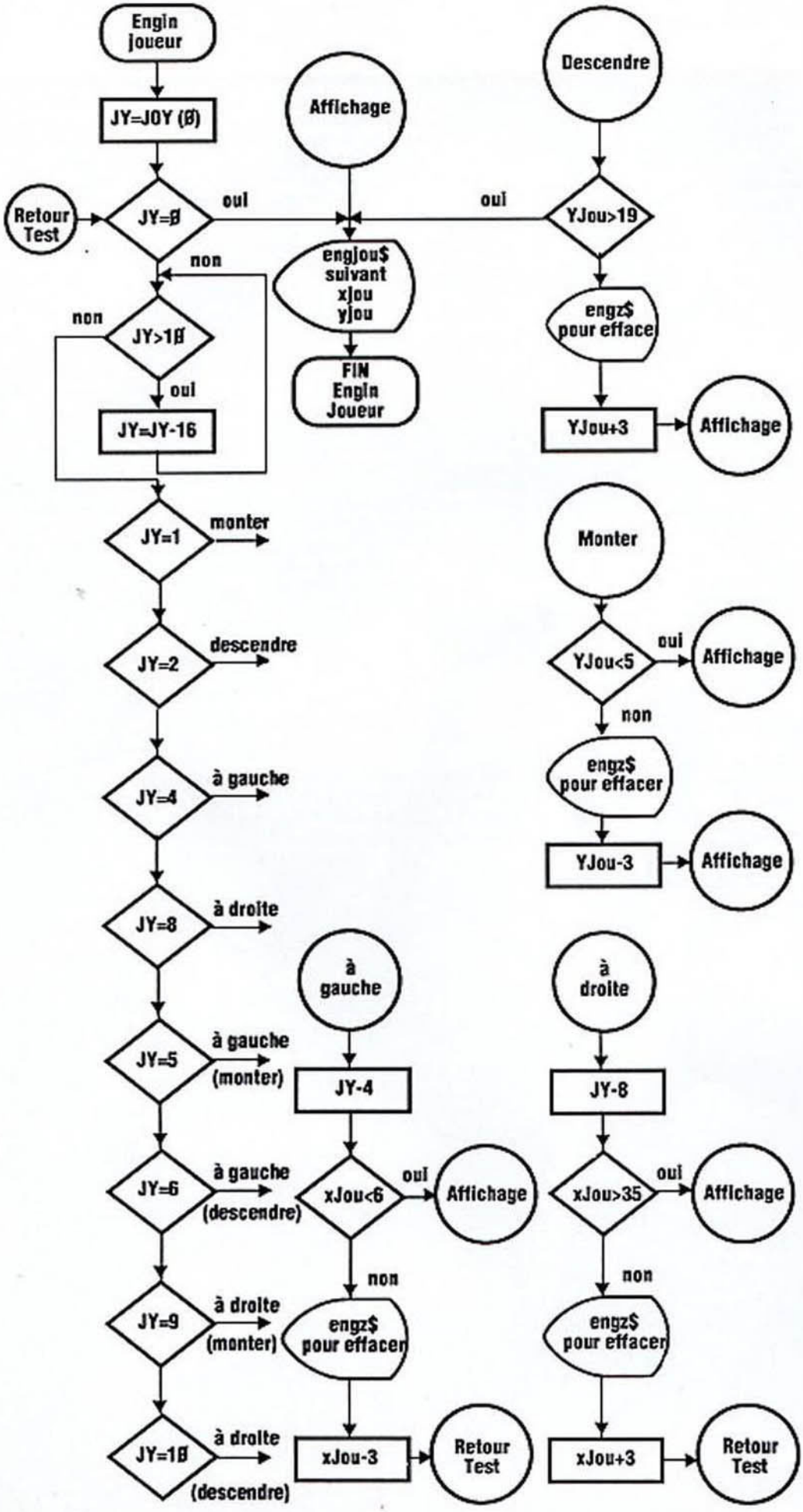
Ton joystick bien en main, tu pilotes ton engin à gauche ou à droite, tu montes et tu descends voilà déjà 4 directions et tu ajoutes les 4 diagonales. Le joystick comporte 4 contacts internes (plus FIRE) et chacun, s'il est activé, envoie une valeur:

Tu obtiens les diagonales avec le joystick en biais (2 contacts simultanés).

Tu affectes maintenant à chaque position le sens de progression



Les diagonales indiquent deux fonctions à la fois. Les valeurs générées par les boutons FIRE sont 16 ou 32. Enfin la progression du joueur en x et en y est de + 3 pour pouvoir rattraper les adversaires. Maintenant je te laisse découvrir l'ordinogramme qui te donne tous les détails du module.



POUR QUELQUES DÉFINITIONS DE PLUS...

Au lieu d'écrire plusieurs fois la même chose, un **sous-programme** représente un ensemble d'instructions qui peut être exécuté depuis différents points du programme (GOSUB numéro ligne). L'instruction de fin de sous-programme est RETURN. Tu confonds, peut-être, INK et couleur. Ton AMSTRAD possède 27 couleurs (numéro 0 a 26) allant du plus foncé au plus clair. Tu disposes de 16 INK en MODE Ø, 4 INK en MODE 1 et de 2 INK en MODE 2. Tu vas donc associer à un numéro d'INK le numéro de couleur que tu as choisi. Tu veux, par exemple, un fond (INK Ø) bleu ciel (couleur numero 11). Tu écris dans ton programme : INK Ø,11. Et si tu voulais une invertion cyclique (flash) de ce bleu ciel vers le bleu pastel (couleur numéro 14) tu écrirais : INK Ø,11,14. Lorsque tu affiches à l'écran par PRINT, tu choisis l'INK de fond par PAPER et l'INK des caractères par PEN. Exemple : PAPER 1 : PEN 2.

Ou en es-tu ? Ca m'intéresse beaucoup de connaître tes difficultés. Actuellement je prépare ton prochain jeu, souhaites-tu quelque chose de plus compliqué ? J'attends ton courrier.

Bien joystiquement vôtre
François LE GRIGUER

LA SEMAINE PROCHAINE

- Les listings avec toujours un maximum d'explications
- La suite de ton jeu avec : • la gestion des adversaires
• le controle des collisions.

PAR FRANÇOIS LE GRIGUER

JOYSTICK ATTACK

J'APPRENDS

TU PROGRESSES, DE MODULE EN MODULE, AVEC CETTE SEMAINE LES ADVERSAIRES ET LES COLLISIONS. L'ENSEMBLE VA TOURNER. ENCORE UN TOUT PETIT PEU DE TRAVAIL ET TU VAS BIENTOT POUVOIR JOUER.

UN CONDENSE DU BASIC

Ton manuel basic est assez volumineux, mais tous les mots-clés ne t'intéressent pas forcément, certains s'utilisent plus rarement. Voici la liste à connaître obligatoirement et que tu as déjà vue dans JOYSTICK-ATTACK pour une bonne part:

AFTER	**FOR**	**MEMORY**	**PRINT**
AUTO	**GOSUB**	**MODE**	**RANDOMIZE**
BORDER	**GOTO**	**MOVE**	**READ**
CALL	**HIMEM**	**MOVER**	**RELEASE**
CAT	**IF**	**NEW**	**REM**
CHR$	**INK**	**NEXT**	**REMAIN**
CLS	**INKEY**	**ON GOSUB**	**RENUM**
DATA	**INKEYS**	**ON GOTO**	**RESTORE**
DELETE	**INPUT**	**ON BREAK**	**RETURN**
DIM	**INT**	**GOSUB**	**RND**
DRAW	**JOY**	**PAPER**	**RUN**
DRAWR	**KEY**	**PEEK**	**WEND**
EDIT	**KEYDEF**	**PEN**	**WHILE**
ENT	**LIST**	**PLOT**	**WINDOW**
ENV	**LOAD**	**PLOTR**	**WINDOW**
EVERY	**LOCATE**	**POKE**	**SWAP**

Petit à petit tous ces termes vont te devenir familier. Tu sauras, pour chaque mot-clé, ce qu'il signifie, qu'elle est sa fonction. Tu seras en mesure de définir instantanément, le mot-clé à choisir en fonction de la ligne BASIC en cours. Tout cela c'est le but de J'APPRENDS.

LES INSTRUCTIONS DES SEQUENCES ET SOUS-PROGRAMMES

1760 Déclaration des coordonnées x et y de l'engin joueur
1780 Définition d'un nombre aléatoire mini = 3 et maxi = 20 pour y jou
1800 Déclaration des variables et constantes suivant notre tableau de la semaine dernière
1860 Boucle de programme pour la mise à zéro de x en fichier advers
1900 code = 1 = fuite adversaires
1910 Initialisation chrono n° 0 pour l'inversion fuite/attaque
1920 Couleur du fond = fuite adversaires
1930 Initialisation du chrono n° 1 pour l'augmentation des adversaires
1940 Initialisation du chrono n° 2 pour l'exécution du bonus
1950 Effacement écran et affichage 25 lignes/40 caractères/4 couleurs

```
17Ø ' INIT DEBUT PARTIE
18Ø GOSUB 176Ø
19Ø ' INIT TABLEAU DE BORD
2ØØ GOSUB 211Ø
21Ø ' GESTION ENGIN JOUEUR
22Ø GOSUB 218Ø
23Ø RUN
173Ø '*********************************
174Ø ' INIT DEBUT PARTIE JOUEUR/ADVERSAIRES
175Ø '*********************************
176Ø xjou=4:SPEED INK 1,1
177Ø mini=3:maxi=2Ø
178Ø GOSUB 77Ø
179Ø yjou=nombre
18ØØ bordf=15
181Ø fondØ=13:fond1=11
182Ø point=1ØØ:pointm=3ØØ
183Ø bonus=1ØØØ
184Ø vies=2Ø:score=Ø:adv=Ø
185Ø advmax=1
186Ø FOR j=1 TO 5:advers(j,1)=Ø:NEXT j
187Ø vpoint=point
188Ø tbonus=15ØØ
189Ø pladv=1ØØØ
19ØØ code=1
191Ø AFTER 4ØØ,Ø GOSUB 311Ø
192Ø INK Ø,fond1
193Ø EVERY pladv,1 GOSUB 316Ø
194Ø AFTER tbonus,2 GOSUB 298Ø
195Ø MODE 1
196Ø engjou$=CHR$(231)
```

1960 Déclaration des constantes pour l'affichage et l'effacement du joueur et des adversaires
2000 Fin de séquence
2100 Séquence d'initialisation du tableau de bord en bas d'écran
2110 n° d'INK particuliers pour PEN et PAPER
2120 Affichage des Libellés et des variables vies et scores (1)
2130 Affichage du score par sous-programme
2140 Fin du sous-programme
2170 Séquence gestion de l'engin du joueur
2180 JOY (0) = mot-clé qui donne la valeur du joystick 0
2190 si = 0 pas de commande du joystick. Suite en affichage
2200 si > 10 indique l'utilisation de FIRE. Soustraire 16 et retour sur le test
2210 tu pourrais écrire 1 ligne de test par valeur mais ON GOTO simplifie l'écriture. GOTO (branchement) s'effectue au premier numéro de ligne si jy = 1 au 2ème jy = 2... au 10ème si jy = 10
2250 Afficher l'engin joueur et fin de séquence
2290 Tester la limite de déplacement. Si impossible suite en affichage sinon effacer la position en cours et mettre à jour y jou
2330 Principe identique à la ligne 2290
2370 Soustraire la valeur joystick droite (8) pour le cas des diagonales (2 ordres simultanés). Test et Maj même principe qu'en 2290. Retour en test de jy (2190) pour un 2ème ordre éventuel.
2410 Principe identique à la ligne 2370.
2960 sous-programme exécuté suivant le chrono n° 2
2980 Augmentation du prochain délai
2990 Initialisation du nouveau délai
3000 Affichage. La valeur = bonus * adv (1000 * nb. adversaires)
3010 MAJ de score + valeur du bonus
3020 Boucle de tempo pour temps d'affichage
3030 Effacer BONUS à l'écran
3040 Déclarer v point avec la valeur point m
3050 Afficher le nouveau score
3060 Fin du sous-programme
3080 Sous-programme exécuté suivant le chrono n° 0
3110 Si le code = 1 alors le code = 0 et le fond = fond 0 sinon le code = 1 et le fond = fond 1
3120 Initialiser le nouveau délai en utilisant les variables xjou et yjou pour varier
3140 Sous-programme exécuté suivant le chrono n° 1
3150 Ajouter 1 à la variable advmax si < 5
3180 sous-programme pour l'affichage de la valeur du score dans le tableau de bord
3200 Afficher avec les couleurs PEN et PAPER du tableau de bord et rétablir les couleurs d'origine
3220 sous-programme pour l'affichage du nombre de vies dans le tableau de bord
3240 Mêmes précautions que pour l'affichage du score

HOT LINE

La solution pratique à vos problèmes
Chaque semaine, cette nouvelle rubrique vous proposera l'astuce, le conseil, ou la solution pour vous aider à résoudre vos problèmes.
JOYSTICK Hebdo, s'est entouré d'une équipe de programmeurs qui examinera les questions les plus souvent posées et les problèmes que vous rencontrez sur les listings de "JEUX CRACK" ou dans votre programmation sur notre initiation ainsi que sur les différents langages machine. Nous nous efforcerons d'y apporter une solution accessible à tous. Mais cette rubrique est avant tout la vôtre. Nous publierons également les trucs et astuces issus de votre expérience.

Alors, partagez vos connaissances et écrivez-nous nombreux.

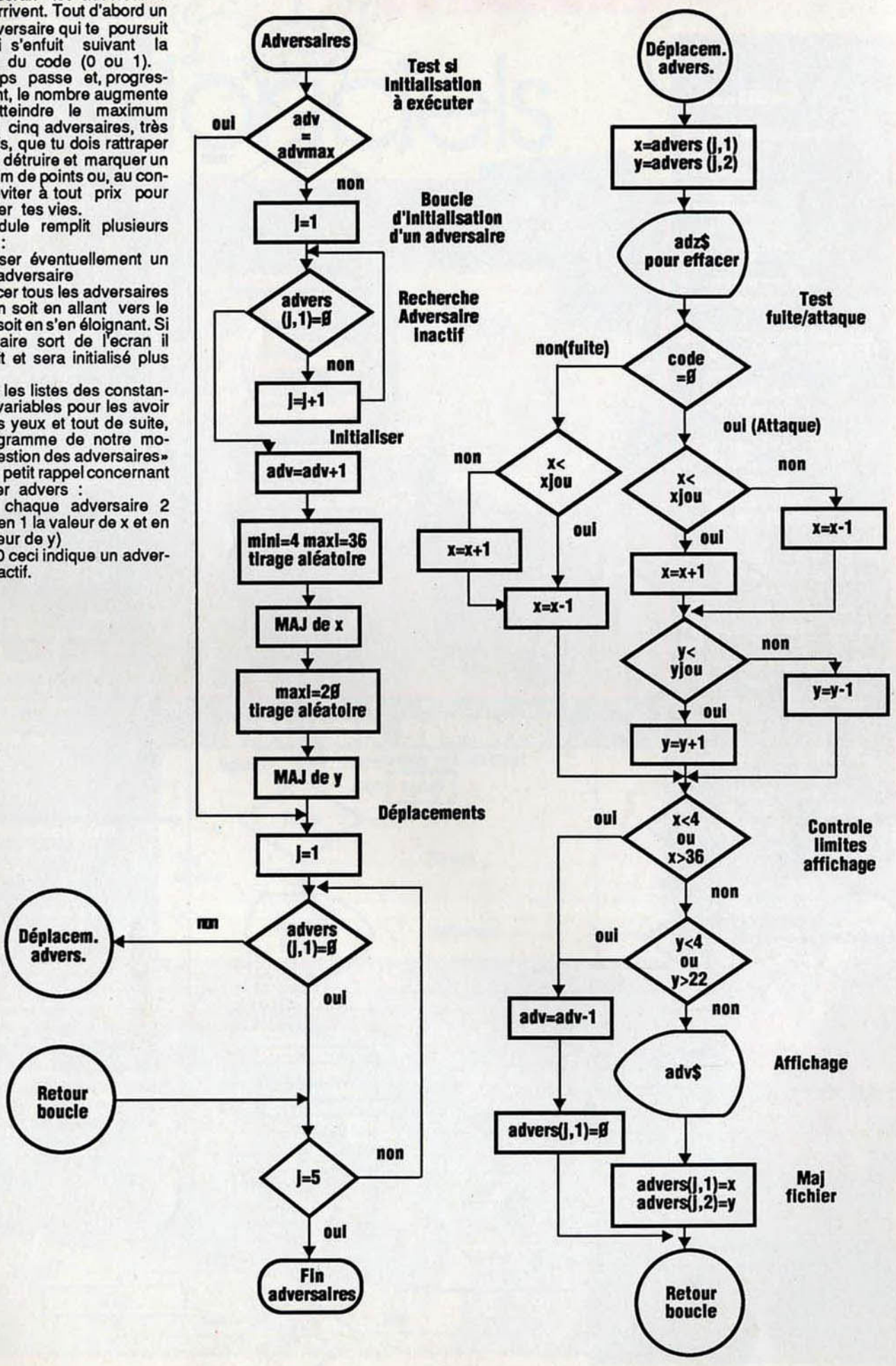
JOYSTICK Hebdo
HOT LINE
177, rue St Honoré
75001 Paris

```
197Ø engz$=» «
198Ø adv$=CHR$(232)
199Ø adz$=» «
2ØØØ RETURN
2Ø8Ø '*********************************
2Ø9Ø '  INIT TABLEAU DE BORD
21ØØ '*********************************
211Ø LOCATE 3,24:PEN 1:PAPER 3
212Ø  PRINT» VIES «;vies;»SCORE        HIGH»;score(1)
213Ø GOSUB 32ØØ
214Ø RETURN
215Ø '*********************************
216Ø '  GESTION ENGIN JOUEUR
217Ø '*********************************
218Ø jy=JOY(Ø)
219Ø IF jy=Ø THEN 225Ø
22ØØ IF jy>1Ø THEN jy=jy-16:GOTO 22ØØ
2205 dir=jy
221Ø ON jy GOTO
2290,2330,1Ø,2410,2410,2410,1Ø,2370,2370,2370
2220 '-----------------------
2230 ' afficher engin joueur
2240 '-----------------------
225Ø LOCATE xjou,yjou:PRINT engjou$:RETURN
226Ø '-----------------------
227Ø ' joy=1 monter
228Ø '-----------------------
229Ø IF yjou <5 THEN 225Ø ELSE LOCATE xjou,yjou:PRINT
engz$:yjou=yjou-3:GOTO 225Ø
23ØØ '-----------------------
2310 ' joy=2 descendre
2320 '-----------------------
233Ø IF yjou>19 THEN 225Ø ELSE LOCATE xjou,yjou:PRINT
engz$:yjou=yjou+3:GOTO 225Ø
234Ø '-----------------------
235Ø ' JOY=8 A DROITE
236Ø '-----------------------
237Ø jy=jy-8:IF xjou>35 THEN 219Ø ELSE LOCATE
xjou,yjou:PRINT engz$:xjou=xjou+3:GOTO 219Ø
238Ø '-----------------------
239Ø '  JOY=4 A GAUCHE
24ØØ '-----------------------
241Ø jy=jy-4:IF xjou<6 THEN 219Ø ELSE LOCATE
xjou,yjou:PRINT engz$:xjou=xjou-3:GOTO 219Ø
242Ø '*********************************
292Ø '
293Ø ' S O U S - P R O G R A M M E S
294Ø '
295Ø '*********************************
296Ø '     BONUS
297Ø '*********************************
298Ø tbonus=tbonus+1ØØ
299Ø AFTER tbonus,2 GOSUB 298Ø
3ØØØ LOCATE 12,12:PRINT»B O N U S  «;bonus*adv
3Ø1Ø score=score+bonus*adv
3Ø2Ø FOR b=1 TO 3ØØ:NEXT
3Ø3Ø PAPER Ø:LOCATE 12,12:PRINT»          «
3Ø4Ø vpoint=pointm
3Ø5Ø GOSUB 32ØØ
3Ø6Ø RETURN
3Ø7Ø '*********************************
3Ø8Ø '   INVERSION ADVERSAIRES
3Ø9Ø ' CODE 1=FUITE  Ø=ATTAQUE
31ØØ '*********************************
311Ø IF code=1 THEN code=Ø:INK Ø,fondØ ELSE code=1:INK
Ø,fond1
312Ø AFTER 1ØØ+(yjou+xjou)*4,Ø GOSUB 311Ø:RETURN
313Ø '*********************************
314Ø ' AUGMENTER LE NOMBRE ADVERS MAXI
315Ø '*********************************
316Ø IF advmax=5 THEN RETURN ELSE
advmax=advmax+1:RETURN
317Ø '*********************************
318Ø '  AFFICHER SCORE
319Ø '*********************************
32ØØ LOCATE 22,24:PEN 1:PAPER 3:PRINT score:PEN
1:PAPER Ø:RETURN
321Ø '*********************************
322Ø ' AFFICHER VIES
323Ø '*********************************
324Ø LOCATE 9,24:PEN 1:PAPER 3:PRINT vies:PEN
1:PAPER Ø:RETURN
325Ø '*********************************************************
326Ø '*********************************************************
```

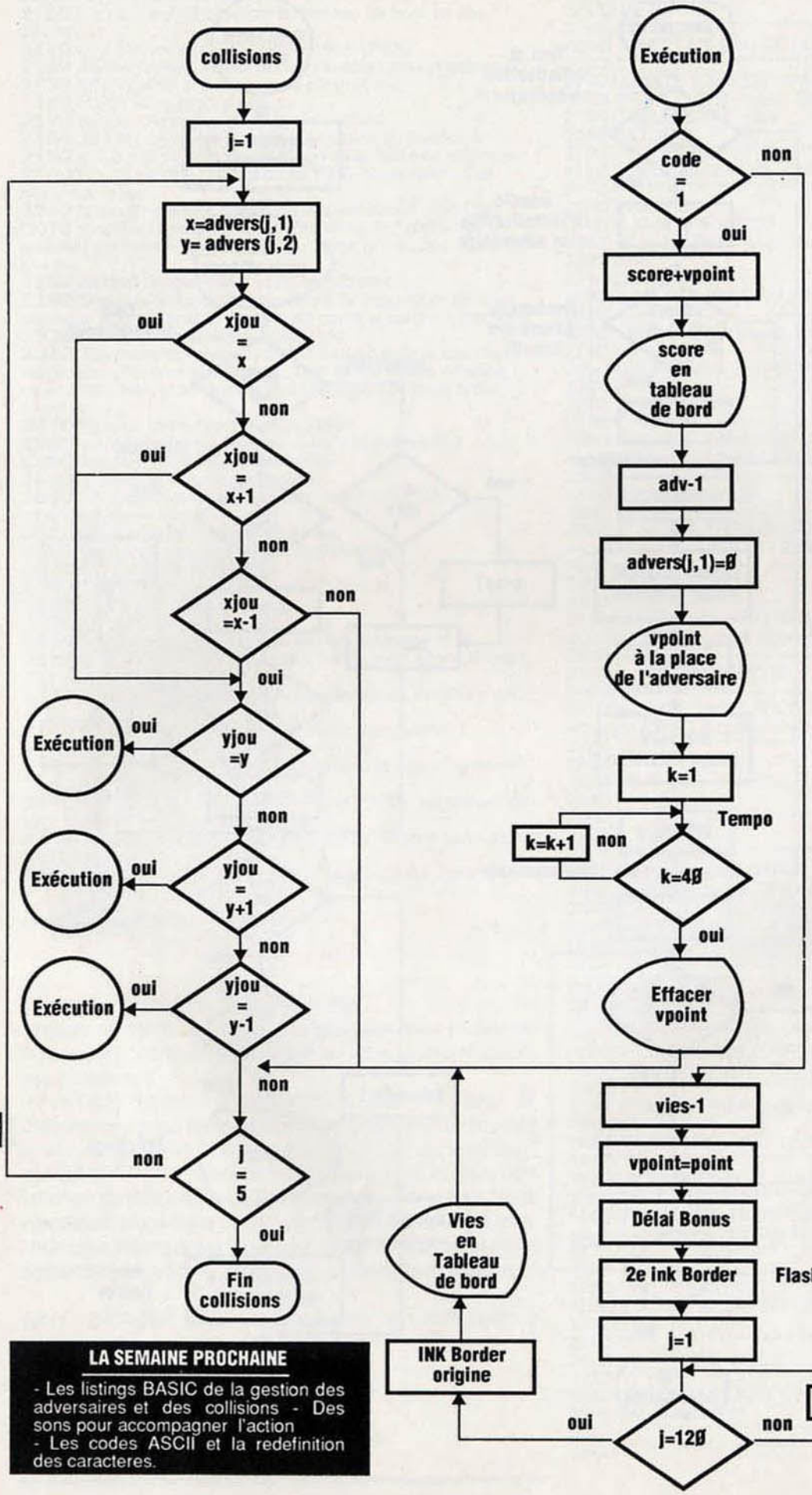
DES ADVERSAIRES AGRESSIFS

Tu n'es plus seul à l'écran les adversaires arrivent. Tout d'abord un petit adversaire qui te poursuit ou qui s'enfuit suivant la position du code (0 ou 1).
Le temps passe et, progressivement, le nombre augmente pour atteindre le maximum fixé de cinq adversaires, très agressifs, que tu dois rattraper pour les détruire et marquer un maximum de points ou, au contraire, éviter à tout prix pour préserver tes vies.
Ce module remplit plusieurs taches :
- Initialiser éventuellement un nouvel adversaire
- Déplacer tous les adversaires à l'écran soit en allant vers le joueur, soit en s'en éloignant. Si l'adversaire sort de l'ecran il disparait et sera initialisé plus tard.

Tu sors les listes des constantes et variables pour les avoir sous les yeux et tout de suite, l'ordinogramme de notre module «gestion des adversaires» avec un petit rappel concernant le fichier advers :
- pour chaque adversaire 2 zones (en 1 la valeur de x et en 2 la valeur de y)
- si x = 0 ceci indique un adversaire inactif.



LES COLLISIONS



Le joueur se déplace, les adversaires aussi, ton programme doit maintenant contrôler les collisions. Ces rencontres entre joueur et adversaires génèrent, suivant le code (attaque ou fuite) la perte d'une vie ou au contraire une augmentation du score. Le contrôle s'effectue ici par comparaison des coordonnées avec un nombre limité de tests. Comment va-t-on matérialiser ces collisions à l'écran ? Pour les gains, tu remplaces l'adversaire détruit par la valeur des points marqués pendant un cours instant.

La perte d'une vie donne un flash rapide de la bordure d'écran.

Voilà pour la partie visible mais n'oublie pas qu'une perte affecte le fonctionnement du bonus (défini pour une période de temps sans perte) et ramène la valeur des points au niveau de base, s'ils étaient multipliés depuis le précédent bonus. Il te reste, bien sûr, à prévoir l'affichage dans le tableau de bord des vies ou du score. Aprés cette séquence le programme principal doit tester le nombre de vies et s'il est égal à 0 la partie s'achève par GAME OVER. Rien de tel qu'un petit ordinogramme pour schématiser la suite logique du module.

GAME OVER

Une toute petite séquence pour compléter ton jeu comme un «pro» lors de la fin de partie. Une page d'écran toute simple en mode 0 avec «GAME OVER» en plein milieu et tu prévois une tempo pour le temps d'affichage.

Le nombre de modules reste très limité et il faut impérativement que tu maitrises bien l'ensemble, que tu saches où retrouver la mise à jour d'une variable, quel est le rôle d'une séquence, ou encore, à quoi correspond la valeur d'une constante. Tu dois te sentir à l'aise à n'importe quel endroit de ton programme et, si ce n'est pas le cas, revois l'ordinogramme et le listing, tu deviens le programmeur et non plus, le spectateur qui entre un listing sans comprendre ce qu'il fait.

Bien joystiquement vôtre
François LE GRIGUER

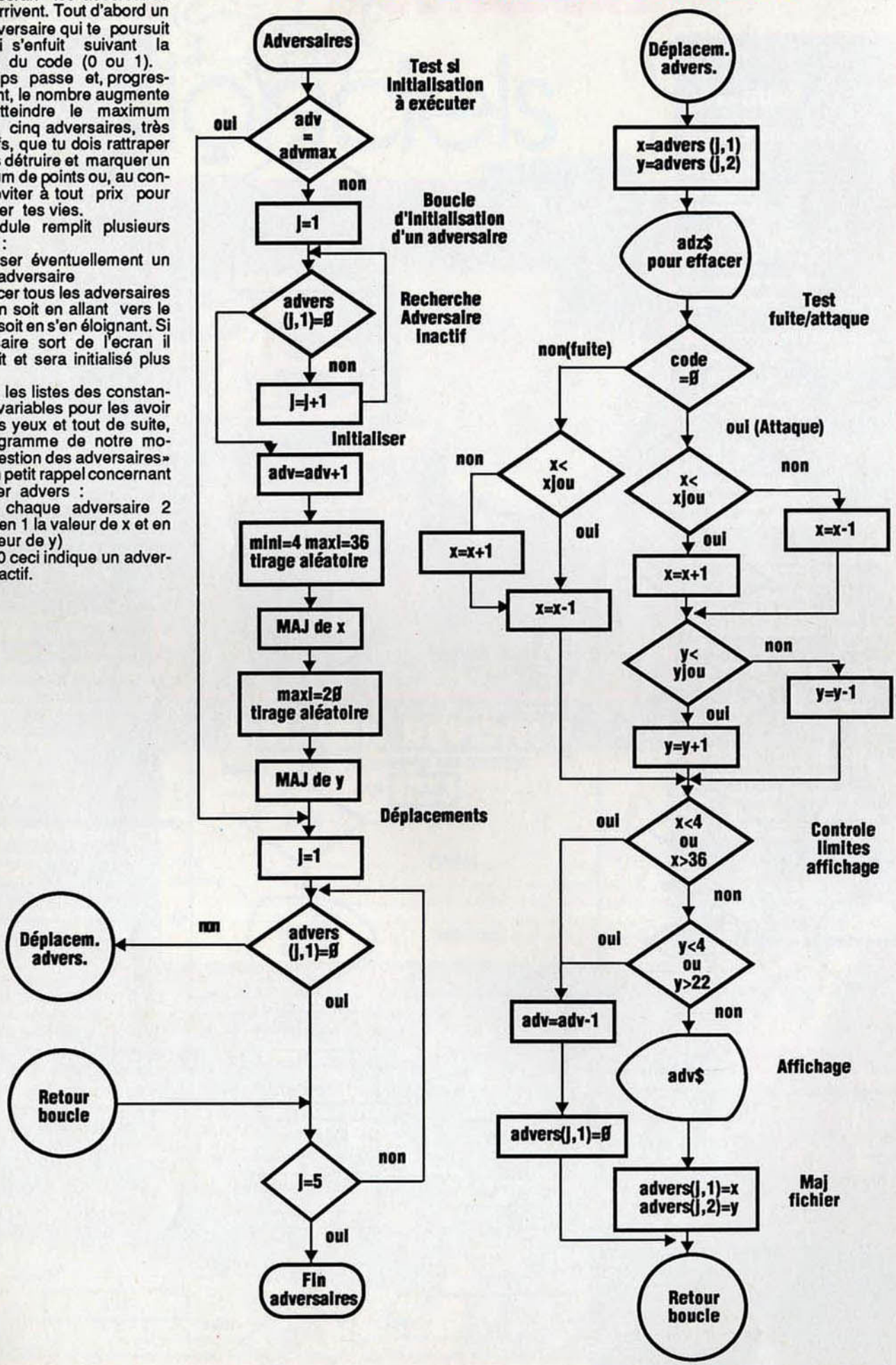
LA SEMAINE PROCHAINE

- Les listings BASIC de la gestion des adversaires et des collisions - Des sons pour accompagner l'action
- Les codes ASCII et la redefinition des caracteres.

DES ADVERSAIRES AGRESSIFS

Tu n'es plus seul à l'écran les adversaires arrivent. Tout d'abord un petit adversaire qui te poursuit ou qui s'enfuit suivant la position du code (0 ou 1).
Le temps passe et, progressivement, le nombre augmente pour atteindre le maximum fixé de cinq adversaires, très agressifs, que tu dois rattraper pour les détruire et marquer un maximum de points ou, au contraire, éviter à tout prix pour préserver tes vies.
Ce module remplit plusieurs taches :
- Initialiser éventuellement un nouvel adversaire
- Déplacer tous les adversaires à l'écran soit en allant vers le joueur, soit en s'en éloignant. Si l'adversaire sort de l'ecran il disparait et sera initialisé plus tard.

Tu sors les listes des constantes et variables pour les avoir sous les yeux et tout de suite, l'ordinogramme de notre module «gestion des adversaires» avec un petit rappel concernant le fichier advers :
- pour chaque adversaire 2 zones (en 1 la valeur de x et en 2 la valeur de y)
- si x = 0 ceci indique un adversaire inactif.



JOYSTICK ATTACK

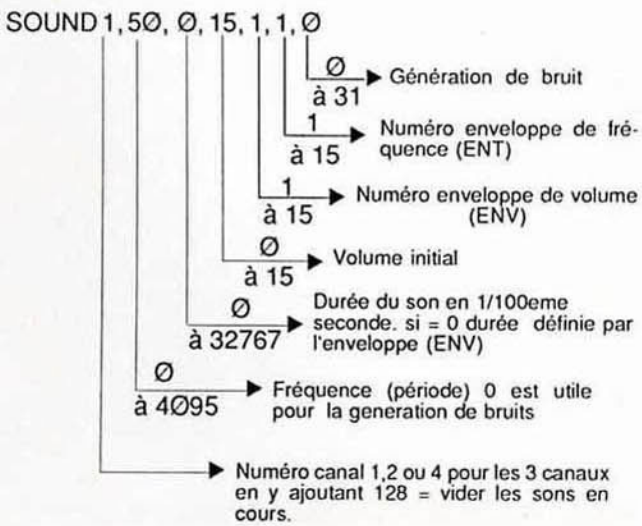
PAR FRANÇOIS LE GRIGUER

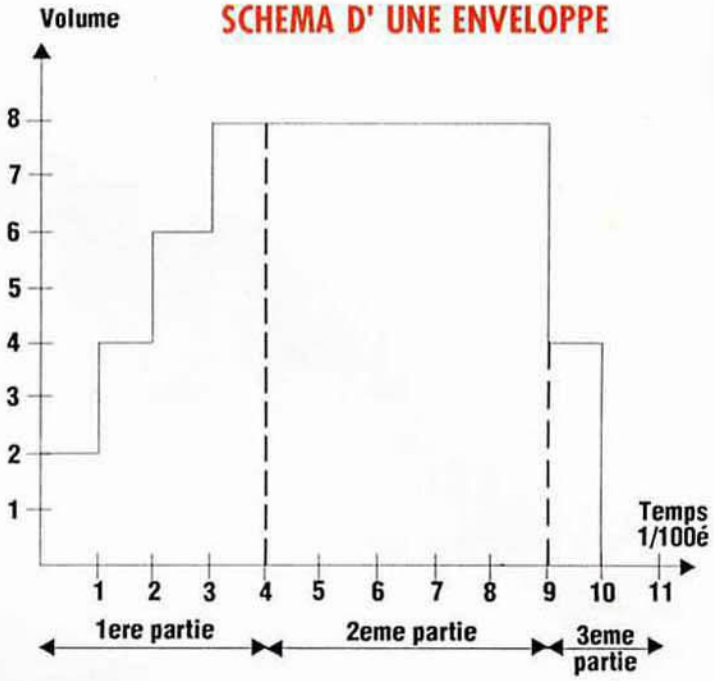
J'APPRENDS

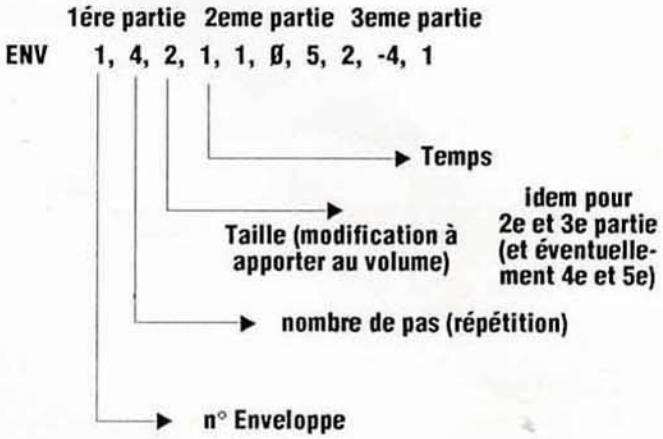
AVEC LES LISTINGS DESADVERSAIRES ET DES COLLISIONS ET L'ADJONCTION DE SONS, JOYSTICK ATTACK S'ACHEVE AUJOURD'HUI. UN PETIT BILAN ET ON ENCHAINE IMMEDIATEMENT AVEC UN PROGRAMME PLUS COMPLEXE.

UN PEU DE BRUIT

Tu y pensais certainement, tu dois trouver ton jeu un peu trop silencieux. Les sons accompagnent l'action ou soulignent les points forts, les moments importants, comme les gains, les pertes, le bonus ou le GAME OVER. Je trouve les grandes explications fastidieuses et, le son, est un domaine un peu compliqué alors je te propose directement les instructions à incorporer dans ton programme avec un minimum de commentaires. L'instruction SOUND produit un son (note musicale ou bruit) à partir des 7 paramètres qu'elle contient :



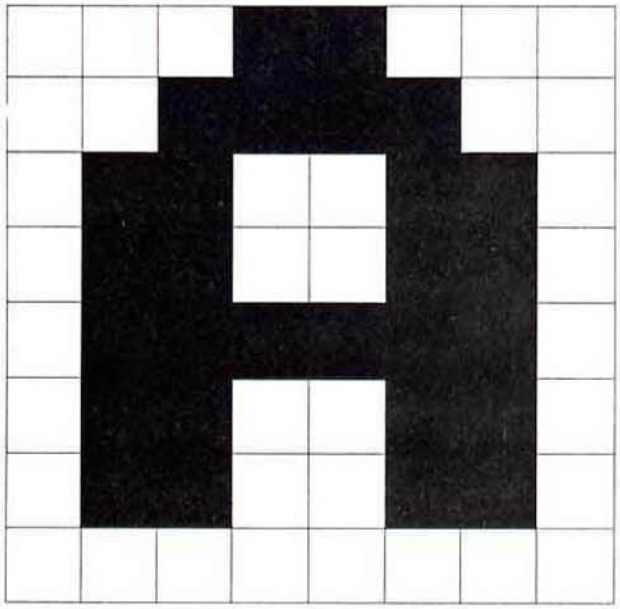




Une enveloppe définit le profil (attaque et chute) du son: ENT pour le profil de la fréquence ENV pour le profil du volume.
Un profil (ou forme) se dessine à partir, du temps en 1/100eme de seconde sur l'axe des x et, du volume ou de la fréquence sur l'axe des y. Voici, ci-dessus à titre d'exemple, un profil de volume en trois parties (il peut y en avoir 5 au maximum), écrit en BASIC.
Après ces quelques explications, tu vas mieux comprendre le contenu des sons à ajouter dans ton programme, et rien ne t'empêche de les modifier ou même d'en ajouter d'autres. Les lignes qui suivent sont à créer ou à compléter dans ton programme:
735 Initialisation d'une enveloppe de fréquence et d'une enveloppe de volume.
2060 Ligne à modifier pour ajouter l'instruction SOUND et CALL & BCA7 (vider tous les sons en cours).
2205 Conserver la dernière direction du joueur pour modifier le son en collisions.
2750 Ajouter CALL & BCA7.
2785 Instruction SOUND pour un adversaire détruit avec un son différent suivant la direction du joueur.
2835 Boucle avec SOUND pour accompagner le flash et la perte de vie.
3015 Boucle avec SOUND pour le bonus.
NOTA: l'instruction CALL & BCA7 fait appel à une routine système. Tu auras l'occasion, dans le prochain jeu, de te familiariser avec ces routines qui sont de petits sous-programmes écrits en assembleur, et voilà le pourquoi du CALL au lieu du GOSUB que tu connais. Le symbole & indique une valeur en base 16 (hexadécimal).

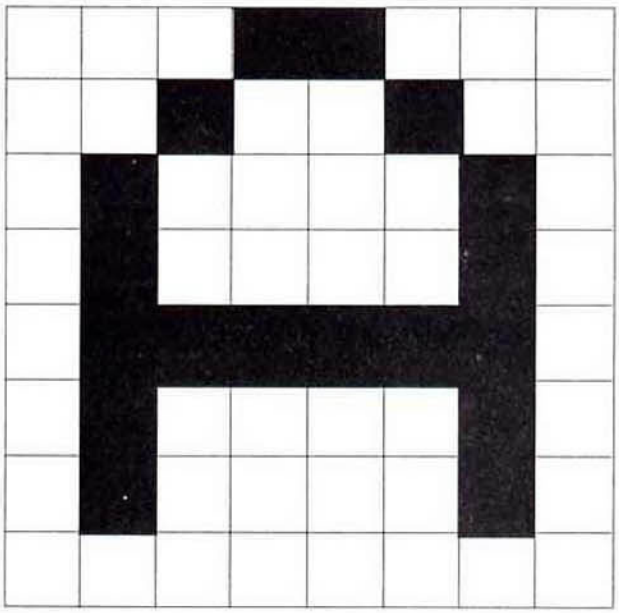
Utiliser directement les caractères prévus par le systeme, c'est simple mais banal et, tu as surement envie de les modifier. Ceci nous permet, au passage, de voir rapidement la codification ASCII (American Standard Code for Information Interchange - en français : code standard americain pour l'echange d'information).
Il s'agit d'une codification internationale donnant a chaque caractère un numéro de 0 a 127 :
0 à 31 : caractères spéciaux.
32 à 47 : caractères de ponctuation et symboles.
48 à 57 : 10 caractères numériques (0 a 9).
58 à 64 : caractères de ponctuation et algébriques.
65 à 90 : alphabet majuscules (A a Z).
97 à 122 : alphabet minuscules (a à z).
De 128 à 255 ce sont des caractères libres définis simplement par le constructeur, en dehors de la codification ASCII. Pour visualiser, à l'écran, tout cet ensemble de caractères, tu composes : MODE 1 : FOR j = 32 TO 255 : PRINT CHR$(j); : NEXT.
Et pour saisir les différences entre les 3 modes d'ecran retape la ligne avec MODE 0 ou avec MODE 2 au lieu de MODE 1.
Apres cet exposé rapide du code ASCII tu vas, maintenant, comprendre plus facilement comment changer la forme d'un caractère, le personnaliser suivant tes goûts.

Avec du papier quadrillé 5 x 5 tu traces un carré de 8 x 8 pour la matrice du caractère. Chaque division représente un point (pixel).
Voici la lettre A majuscule (code ASCII 65).
Tu décides de redessiner le caractère A et, en face de chaque ligne tu écris de la gauche vers la droite 0 pour un pixel clair et 1 pour un pixel foncé:



Lettre A majuscule (code ASCII 65)

Les pixels foncés s'affichent à l'écran avec l'INK définie par PEN et les pixels clairs avec l'INK de PAPER.



1ere ligne	ØØØ11ØØØ
2eme ligne	ØØ1ØØ1ØØ
3eme ligne	Ø1ØØØØ1Ø
4eme ligne	Ø1ØØØØ1Ø
5eme ligne	Ø1111110
6eme ligne	Ø1ØØØØ1Ø
7eme ligne	Ø1ØØØØ1Ø
8eme ligne	

Il reste à entrer cette nouvelle codification de A dans ton programme. Une instruction préliminaire indique au système à partir de quel code caractère les redéfinitions vont intervenir. Par exemple : SYMBOL AFTER 32 permet de redéfinir tous les caractères du code 32 au code 255. Ensuite, la nouvelle codification de chaque caractère est effectuée par l'instruction SYMBOL. Pour ton nouveau A :
SYMBOL65,& X 11000, &X100100, etc... lignes 3 a 7 numéro du caractère codification 1ere ligne codification 2eme ligne. Trois remarques sur la codification des lignes :
- &X veut dire que le nombre qui suit est en base de numération binaire.
- mettre directement le premier 1 de la ligne, les 0 devant sont inutiles.
- la 8eme ligne ne contenant que des 0 n'est pas à entrer. Pour visualiser le nouveau caractère A :
- frappes A majuscule au clavier
- ou PRINT»A»
- ou PRINT CHR$(65).
Tu sais tout sur les caractères et leur redéfinition et, avec un peu d'imagination, tu vas obtenir quelque chose de vraiment personnalisé à ajouter au début de tous tes programmes.
Par exemple, pour notre jeu, JOYSTICK ATTACK :

```
85 GOSUB 5000
5000 "sous-programme redéfinition des caractères
5010 SYMBOL AFTER 32
5020 SYMBOL..... a suivre
```

une instruction SYMBOL pour chacun des caractères et en fin RETURN.
Dans JOYSTICK ATTACK, tu vas redèfinir la forme de l'engin du joueur et des adversaires.

LES ADVERSAIRES EN BASIC

2450 Si le nombre d'adversaires = le nombre maxi pas d'initialisation à exécuter.
2480 Boucle pour rechercher un adversaire inactif (x = 0)
2490 Initialisation. Nombre d'adversaires + 1
2500 Tirage d'un nombre aléatoire mini 4 et maxi 36 pour x
2520 Tirage aléatoire pour y
2550 Déplacement de tous les adversaires actifs (x ≠ 0)
2580 Effacer l'adversaire dans sa position actuelle
2590 Test du code pour la définition de la Maj de x et y
2610 Maj de x en attaque
2620 Maj de y en attaque
2640 Maj de x en fuite. Pas de Maj de y mais tu peux l'ajouter
2650 Contrôle des limites d'affichage pour x et y. S'il y a dépassement l'adversaire disparait (adv -1 et advers (j,1) = 0)
2670 Affichage de l'adversaire suivant ses nouvelles coordonnées et Maj des coordonnées en fichier
2680 Contrôle de boucle et fin séquence

Le programme des collisions
2730 x et y de l'adversaire
2740 Comparaison de x avec x joueur. Si le résultat est oui suite en comparaison de y
2750 Contrôle fin de boucle et fin de la séquence
2770 Comparaison de y. si oui = collision. Si non = suite boucle
2780 Si le code = 1 = le joueur marque. Maj de score et affichage en tableau de bord. -1 en adversaires. Affichage des points marqués en x, y adversaire.
2790 Tempo pour temps d'affichage des points à l'écran
2800 Effacer les points
2810 0 en x advers. L'adversaire disparait du fichier et suite boucle.
2830 Le joueur perd une vie
2840 Retour de la valeur des points au minimum
2850 Init d'un nouveau délai pour l'exécution du bonus
2860 Faire flasher le Bord de l'écran avec l'indication de 2 couleurs (0, bord f).
2870 boucle de tempo pour temps de flash.
2880 Retour des bord de l'écran à une seule couleur
2890 Affichage du nombre de vies en tableau de bord.
2900 Retour dans la boucle principale

UN PETIT BILAN ?

JOYSTICK ATTACK est terminé. C'était le premier volet d'une longue série. Pour certains, il s'agissait d'une véritable découverte de la programmation, pour d'autres d'un perfectionnement. Dans tous les cas, si tu as vraiment bien compris l'ensemble de la logique et du programme, l'objectif est atteint. Tu réalises aussi, peut être, un peu mieux la masse de travail que représente le développement de tes jeux favoris. Mais l'informatique ne s'apprend pas en un jour et nécessite une pratique régulière. Avec le nouveau jeu que je te propose, tu vas pouvoir aller plus loin, aborder de nouveaux problèmes.

Bien Joystiquement vôtre.
FRANÇOIS LE GRIGUER

LA SEMAINE PROCHAINE

JOYSTICK-BREAKER

Un casse-briques avec 40 tableaux redéfinissables

```
230 ' GESTION COLLISIONS
240 GOSUB 2720
250 IF vies<1 THEN GOSUB 2040:GOSUB 1360:GOTO 180
260 '  GESTION DES ADVERSAIRES
270 GOSUB 2450
280 GOTO 220
2010 '*********************************
2020 ' GAME OVER
2030 '*********************************
2040 MODE 0
2050 LOCATE 6,12:PRINT»GAME OVER»
2060 FOR j=1 TO 300:NEXT
2070 RETURN
2420 '*********************************
2430 '  GESTION DES ADVERSAIRES
2440 '*********************************
2450 '
2460 IF adv=advmax THEN 2550
2470 '    init adversaires
2480 FOR j=1 TO 5:IF advers(j,1)=0 THEN 2490 ELSE NEXT j
2490 adv=adv+1
2500 mini=4:maxi=36:GOSUB 770
2510 advers(j,1)=nombre
2520 maxi=20:GOSUB 770
2530 advers(j,2)=nombre
2540 '     AFFICHER ADVERS
2550 FOR j=1 TO 5
2560 IF advers(j,1)=0 THEN 2680
2570 x=advers(j,1):y=advers(j,2)
2580 LOCATE x,y:PRINT adz$
2590 '  LE CODE=1=ADVERS FUITE . CODE=0=ADVERS
ATTAQUE
2600 IF code=1 THEN 2640
2610 IF x<xjou THEN x=x+1 ELSE x=x-1
2620 IF y<yjou THEN y=y+1 ELSE y=y-1
2630 GOTO 2650
2640 IF x<xjou THEN x=x-1 ELSE x=x+1
2650 IF x<4 OR x>36 THEN advers(j,1)=0:adv=adv-1:GOTO
2680
2660 IF y<4 OR y>22 THEN advers(j,1)=0:adv=adv-1:GOTO
2680
2670 LOCATE x,y:PRINT adv$:advers(j,1)=x:advers(j,2)=y
2680 NEXT j:RETURN
2690 '*********************************
2700 '  CONTROLE COLLISIONS
2710 '*********************************
2720 FOR j=1 TO 5
2730 x=advers(j,1):y=advers(j,2)
2740 IF xjou=x OR xjou=x-1 OR xjou=x+1 THEN 2770
2750 NEXT j:RETURN
2760 '
2770 IF yjou=y OR yjou=y-1 OR yjou=y+1 OR yjou=y-2 OR
yjou=y+2 THEN 2780 ELSE 2750
2780 IF code =1 THEN score=score+vpoint:GOSUB
3200:adv=adv-1:LOCATE x,y:PRINT vpoint ELSE 2830
2790 FOR k=1 TO 40:NEXT k
2800 LOCATE x,y:PRINT»   «
2810 advers(j,1)=0
2820 GOTO 2750
2830 vies=vies-1
2840 vpoint=point
2850 AFTER tbonus,2 GOSUB 2980
2860 BORDER 0,bordf
2870 FOR j=1 TO 120:NEXT
2880 BORDER 0
2890 GOSUB 3240
2900 GOTO 2750
2910 '*********************************
```

L'ENSEMBLE DU LISTING DE JOYSTICK-ATTACK

```
10 REM**** JOYSTICK ATTACK **********
20 REM
30 REM ::::::::::::::::::::::::::::
40 REM ::::::::::::::::::::::::::::
50 REM     PROGRAMME PRINCIPAL
60 REM ::::::::::::::::::::::::::::
70 REM ::::::::::::::::::::::::::::
80 CLS
90 ' INITIALISATION DES FICHIERS
100 GOSUB 360
110 ' 1 ere PARTIE DU GENERIQUE
120 GOSUB 830
130 ' 2eme PARTIE DU GENERIQUE
140 GOSUB 1070
150 ' INITIALISATION PAGE DE SCORE
160 GOSUB 1280
170 ' INIT DEBUT PARTIE
180 GOSUB 1760
190 ' INIT TABLEAU DE BORD
200 GOSUB 2110
210 ' GESTION ENGIN JOUEUR
220 GOSUB 2180
230 ' GESTION COLLISIONS
240 GOSUB 2720
250 IF vies<1 THEN GOSUB 2040:GOSUB
1360:GOTO 180
260 '  GESTION DES ADVERSAIRES
270 GOSUB 2450
280 GOTO 220
290 '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
300 '       S E Q U E N C E S
310 '>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
320 '*****************************
330 ' INITIALISATION DES FICHIERS
GENERIQUE
340 '*****************************
350 '
360 nb=14
370 DIM xlin(nb)
380 DIM ylin(nb)
390 DIM coul(nb)
400 DIM posx(nb)
410 DIM posy(nb)
420 DIM litre$(nb)
430 ' ********* DATA GENERIQUE
*************
440 '  couleurs
450 DATA
13,4,8,11,14,15,16,17,18,19,20,21,22,23,24,25
460 '  litre
470 DATA J,O,Y,S,T,I,C,K,A,T,T,A,C,K
480 '  xlin
490 DATA 3,5,7,9,11,13,15,17,5,7,9,11,13,15
500 '  ylin
510 DATA
11,11,11,11,11,11,11,11,15,15,15,15,15,15
520 '*****************************
530 FOR j=0 TO 15
540 READ a
550 INK j,a
560 NEXT j
570 '
580 FOR j=1 TO nb
590 READ litre$(j)
600 NEXT j
610 '
620 FOR j=1 TO nb:READ xlin(j):NEXT
630 FOR j=1 TO nb:READ ylin(j):NEXT
640 mini=1:maxi=20
650 FOR j=1 TO nb
660 GOSUB 770
670 posx(j)=nombre
680 NEXT
690 maxi=25
700 FOR j=1 TO nb:GOSUB
770:posy(j)=nombre:NEXT
710 coul=1
720 FOR j=1 TO nb:coul(j)=coul:coul=coul+1
730 NEXT
735 ENT 1,60,10,2:ENV 1,10,15,12
740 RETURN
750 '*****************************
760 ' SOUS-PROGR ALEATOIRE
770 RANDOMIZE TIME
780 nombre=INT(RND*(maxi-mini+1))+mini
790 RETURN
800 '*****************************
810 'GENERIQUE 1ere partie deplacement des
caracteres du litre
820 '*****************************
830 PRINT CHR$(22)+CHR$(1)
840 MODE 0
850 flag=0
860 FOR j=1 TO nb
870 IF posx(j)=xlin(j) AND posy(j)=ylin(j)
THEN 950
880 PEN 0
890 LOCATE posx(j),posy(j)
900 PRINT CHR$(143)
910 IF posx(j)<xlin(j) THEN
posx(j)=posx(j)+1:flag=1:GOTO 930
920 IF posx(j)>xlin(j) THEN posx(j)=
posx(j)-1:flag=1
930 IF posy(j)<ylin(j) THEN
posy(j)=posy(j)+1:flag=1:GOTO 950
940 IF posy(j)>ylin(j) THEN posy(j)=
posy(j)-1:flag=1
950 PEN coul(j)
960 LOCATE posx(j),posy(j)
970 PRINT litre$(j)
980 coul(j)=coul(j)+1
990 IF coul(j)>15 THEN coul(j)=1
1000 NEXT J
1010 IF flag=1 THEN 850
1020 FOR j=1 TO 500:NEXT
1030 RETURN
1040 '*****************************
1050 'GENERIQUE 2eme partie «une realisation
de.....
1060 '*****************************
1070 MODE 1
1080 PRINT CHR$(22)+CHR$(0)
1090 y=11:x=1
1100 FOR j=0 TO 5
1110 LOCATE x,y
1120 PEN 0
1130 PRINT "  "
1140 x=x+2:LOCATE x,y
1150 PEN 1
1160 PRINT"une  realisation"
1170 NEXT
1180 x1=39:FOR j=0 TO 8:LOCATE x1,y+2:
PEN 0
:PRINT "  ":LOCATE x1-2,y+2:
PEN 2:PRINT"de":x1=x1-2:NEXT
1190 x=2:FOR j=0 TO 4:LOCATE x,y+5:PEN 0
:PRINT "  ":LOCATE x+2,y+5:PEN 3
:PRINT".....":x=x+2:NEXT
1200 FOR J=1 TO 500:NEXT
1210 RETURN
1220 '*****************************
1230 '  FIN DU GENERIQUE
1240 '*****************************
1250 '
1260 '  PAGE DE SCORE
1270 '*****************************
1280 DIM nom$(8)
1290 DIM score(8)
1300 DIM advers(5,2)
1310 DATA JOYSTICK,FRANCK,PAUL,DENIS,
SOPHIE,YVAN,MAGALIE,ALPHA
1320 DATA 50700,30100,15200,10100,
7300,6900,5100,4500
1330 FOR j=1 TO 8:READ nom$(j):NEXT
1340 FOR J=1 TO 8:READ score(j):NEXT
1350 "
1360 MODE 0:PEN 1:LOCATE 5,2:PRINT "S  C
O R E "
1370 x=3:y=5
1380 FOR j=1 TO 8:IF score>score(j) THEN 1420
1390 NEXT
1400 GOSUB 1620
1410 GOTO 1610
1420 '
1430 '
1440 '   decaler les lignes de score
1450 '
1460 '
1470 IF j=8 THEN 1510
1480 FOR k=7 TO j STEP -1
1490 nom$(k+1)=nom$(k):score(k+1)=score(k)
1500 NEXT k
1510 nom$(j)="....."
1520 score(j)=score
1530 GOSUB 1620
1540 LOCATE x,5+(j-1)*2:nom$(j)=""
1550 FOR k=1 TO 10
1560 n$=INKEY$:IF n$="" THEN 1560
1570 n=ASC(n$):IF n<32 OR n>122 THEN 1560
1580 PRINT n$;
1590 nom$(j)=nom$(j)+n$
1600 NEXT k
1610 IF INKEY$="" THEN 1610 ELSE RETURN
1620 FOR m=1 TO 8
1630 LOCATE x,y
1640 PRINT nom$(m)
1650 LOCATE 13,y
1660 IF score(m)>99999 THEN LOCATE x+9,y
1670 IF score(m)<10000 THEN LOCATE x+11,y
1680 IF score(m)<1000 THEN LOCATE x+12,y
1690 PRINT score(m)
1700 y=y+2
1710 NEXT m
1720 RETURN
1730 '*****************************
1740 ' INIT DEBUT PARTIE JOUEUR/
ADVERSAIRES
1750 '*****************************
1760 xjou=4:SPEED INK 1,1
1770 mini=3:maxi=20
1780 GOSUB 770
1790 yjou=nombre
1800 bord!=15
1810 fond0=13:fond1=11
1820 point=100:pointm=300
1830 bonus=1000
1840 vies=20:score=0:adv=0
1850 advmax=1
1860 FOR j=1 TO 5:advers(j,1)=0:NEXT j
1870 vpoint=point
1880 !bonus=1500
1890 pladv=1000
1900 code=1
1910 AFTER 400,0 GOSUB 3110
1920 INK 0,fond1
1930 EVERY pladv,1 GOSUB 3160
1940 AFTER !bonus,2 GOSUB 2980
1950 MODE 1
1960 engjou$=CHR$(231)
1970 engz$="  "
1980 adv$=CHR$(232)
1990 adz$="  "
2000 RETURN
2010 '*****************************
2020 ' GAME OVER
2030 '*****************************
2040 MODE 0
2050 LOCATE 6,12:PRINT"GAME OVER"
2060 FOR j=1 TO 300:
SOUND 132,j,0,15,1,1:NEXT:CALL &BCA7
2070 RETURN
2080 '*****************************
2090 '  INIT TABLEAU DE BORD
2100 '*****************************
2110 LOCATE 3,24:PEN 1:PAPER 3
2120  PRINT" VIES ";vies;"
SCORE        HIGH";score(1)
2130 GOSUB 3200
2140 RETURN
2150 '*****************************
2160 '  GESTION ENGIN JOUEUR
2170 '*****************************
2180 jy=JOY(0)
2190 IF jy=0 THEN 2250
2200 IF jy>10 THEN jy=jy-16:GOTO 2200
2205 dir=jy
2210 ON jy GOTO 2290,2330,10,2410,2410,
2410,10,2370,2370,2370
2220 '_____________
2230 ' afficher engin joueur
2240 '_____________
2250 LOCATE xjou,yjou:PRINT engjou$:RETURN
2260 '_____________
2270 ' joy=1 monter
2280 '_____________
2290 IF yjou <5 THEN 2250 ELSE
LOCATE xjou,yjou
:PRINT engz$:yjou=yjou-3:GOTO 2250
2300 '_____________
2310 ' joy=2 descendre
2320 '_____________
2330 IF yjou>19 THEN 2250 ELSE
LOCATE xjou,yjou
:PRINT engz$:yjou=yjou+3:GOTO 2250
2340 '_____________
2350 ' JOY=8 A DROITE
2360 '_____________
2370 jy=jy-8:IF xjou>35 THEN 2190 ELSE
LOCATE xjou,yjou
:PRINT engz$:xjou=xjou+3:GOTO 2190
2380 '_____________
2390 '  JOY=4 A GAUCHE
2400 '_____________
2410 jy=jy-4:IF xjou<6 THEN 2190 ELSE
LOCATE xjou,yjou
:PRINT engz$:xjou=xjou-3:GOTO 2190
2420 '*****************************
2430 ' GESTION DES ADVERSAIRES
2440 '*****************************
2450 '
2460 IF adv=advmax THEN 2550
2470 '    init adversaires
2480 FOR j=1 TO 5:IF advers(j,1)=0 THEN 2490
ELSE NEXT j
2490 adv=adv+1
2500 mini=4:maxi=36:GOSUB 770
2510 advers(j,1)=nombre
2520 maxi=20:GOSUB 770
2530 advers(j,2)=nombre
2540 '     AFFICHER ADVERS
2550 FOR j=1 TO 5
2560 IF advers(j,1)=0 THEN 2680
2570 x=advers(j,1):y=advers(j,2)
2580 LOCATE x,y:PRINT adz$
2590 '  LE CODE=1=ADVERS FUITE .
CODE=0=ADVERS ATTAQUE
2600 IF code=1 THEN 2640
2610 IF x<xjou THEN x=x+1 ELSE x=x-1
2620 IF y<yjou THEN y=y+1 ELSE y=y-1
2630 GOTO 2650
2640 IF x<xjou THEN x=x-1 ELSE x=x+1
2650 IF x<4 OR x>36 THEN
advers(j,1)=0:adv=adv-1:GOTO 2680
2660 IF y<4 OR y>22 THEN
advers(j,1)=0:adv=adv-1:GOTO 2680
2670 LOCATE x,y:
PRINT adv$:advers(j,1)=x:advers(j,2)=y
2680 NEXT j:RETURN
2690 '*****************************
2700 '  CONTROLE COLLISIONS
2710 '*****************************
2720 FOR j=1 TO 5
2730 x=advers(j,1):y=advers(j,2)
2740 IF xjou=x OR xjou=x-1 OR xjou=x+1 THEN
2770
2750 NEXT j:CALL &BCA7:RETURN
2760 '
2770 IF yjou=y OR yjou=y-1 OR yjou=y+1 OR
yjou=y-2 OR yjou=y+2 THEN 2780 ELSE 2750
2780 IF code =1 THEN
score=score+vpoint:GOSUB 3200
:adv=adv-1:LOCATE x,y:PRINT vpoint ELSE 2830
2785 SOUND 129,50*dir,0,15,1,1
2790 FOR k=1 TO 40:NEXT k
2800 LOCATE x,y:PRINT"  "
2810 advers(j,1)=0
2820 GOTO 2750
2830 vies=vies-1
2835 FOR k=1 TO 10:
SOUND 132,k,0,15,1,1:NEXT k
2840 vpoint=point
2850 AFTER !bonus,2 GOSUB 2980
2860 BORDER 0,bord!
2870 FOR j=1 TO 120:NEXT
2880 BORDER 0
2890 GOSUB 3240
2900 GOTO 2750
2910 '*****************************
2920 '
2930 ' S O U S - P R O G R A M M E S
2940 '
2950 '*****************************
2960 '    BONUS
2970 '*****************************
2980 !bonus=!bonus+100
2990 AFTER !bonus,2 GOSUB 2980
3000 LOCATE 12,12:PRINT"B O N U S
";bonus*adv
3010 score=score+bonus*adv
3015 FOR b=10 TO 2 STEP -2:SOUND
129,50*b,0,15,1,1:NEXT b
3020 FOR b=1 TO 300:NEXT
3030 PAPER 0:LOCATE 12,12:
PRINT"          "
3040 vpoint=pointm
3050 GOSUB 3200
3060 RETURN
3070 '*****************************
3080 '   INVERSION ADVERSAIRES
3090 ' CODE 1=FUITE  0=ATTAQUE
3100 '*****************************
3110 IF code=1 THEN code=0:INK 0,fond0
ELSE code=1:INK 0,fond1
3120 AFTER 100+(yjou+xjou)*4,0 GOSUB
3110:RETURN
3130 '*****************************
3140 ' AUGMENTER LE NOMBRE ADVERS MAXI
3150 '*****************************
3160 IF advmax=5 THEN RETURN
ELSE advmax=advmax+1:RETURN
3170 '*****************************
3180 '  AFFICHER SCORE
3190 '*****************************
3200 LOCATE 22,24:PEN 1:PAPER 3
PRINT score:PEN 1:PAPER 0:RETURN
3210 '*****************************
3220 '  AFFICHER VIES
3230 '*****************************
3240 LOCATE 9,24:PEN 1:PAPER 3
PRINT vies:PEN 1:PAPER 0:RETURN
3250
*****************************
3260/FIN DU LISTING
*****************************
```