JPC

JUILLET-AOUT 1987
NUMERO 46

Le numéro 40 F

A PROPOS DU CLUB

HP28

HP41

HP75

HP71

EDITORIAL

Chers bronzés,

Je tiens à vous remercier pour votre courrier. Il témoigne de la vie qui anime le Club.

La rubrique consacrée au HP-28C se développe petit à petit. Vous remarquerez trois articles, dont un qui nous a été communiqué à titre exceptionnel par Hewlett-Packard. Cette calculatrice fantastique connaît un succès grandissant, et nous ne nous en plaindrons pas...

La rubrique HP-75 ne contient toujours qu'un seul article. Si vous désirez la voir croître, n'hésitez plus à nous envoyer vos oeuvres.

Vous remarquerez dans ce numéro la signature de Tapani Tarvainen, Président du Club finlandais. Il nous présente un article très intéressant sur la traduction de caractères.

Enfin, et c'est la première fois que nous procédons ainsi, nous avons été obligé de scinder la parution d'un programme. La deuxième partie du programme de généalogie de Philippe Nicodème paraîtra dans le numéro de septembre.

D'ici là, passez de bonnes vacances, et profitez-en pour écrire des articles...

Pierre David (37)

PPC PARIS SE REUNIT UNE FOIS PAR MOIS

Comme vous le savez peut être déjà, PPC Paris se réunit une fois par mois, en plein coeur de Paris. Amenez votre matériel, votre bonne volonté et vos idées ! Plus vous en apporterez, et plus vous en trouverez chez vos collègues de PPC.

Ces réunions se déroulent de manière très libre, aucun ordre du jour, discussion ou autre n'étant imposé. Un membre du bureau est toujours présent. Ainsi, si vous désirez remettre votre article tout frais au Journal, si vous avez des suggestions à faire, si vous voulez vous procurer des anciens numéros de JPC, ce sera en principe toujours possible.

Si donc cela vous intéresse, n'hésitez plus un seul instant, venez nous rejoindre tous les premiers samedis de chaque mois (sauf en période de vacances scolaires) au :

Centre de Jeunesse et de Loisirs Jean Verdier
11 rue de Lancry
75010 Paris

et en montant au deuxième étage, vous entendrez des éclats de rire et des discussions passionnées vers la salle 215. Attention, toutefois, de venir entre 16 et 19h.

Pour l'accès en métro, trois possibilités s'offrent à vous :

- Métro Strasbourg Saint Denis :
Sortie porte St Martin / Bd St Denis, coté pairs

- Métro République :
Sortie Bd St Martin, coté pairs

- Métro Jacques Bonsergent :
Sortie Bd Magenta, coté impairs.

Ah, j'oubliais ! JPC est (souvent) distribué en avant première lors de ces réunions... A bon entendeur, salut !

Les dates des prochaines réunions sont :
Samedi 6 juin 1987
Samedi 5 septembre 1987
Samedi 3 octobre 1987
Samedi 21 novembre 1987
Samedi 5 décembre 1987
Samedi 16 janvier 1988
Samedi 20 février 1988
Samedi 5 mars 1988
Samedi 16 avril 1988
Samedi 7 mai 1988
Samedi 4 juin 1988

Pierre David (37)

AH ! VOUS ECRIVEZ

Vous vous sentez en verve, mais vous ne savez pas sous quelle forme "l'équipe de rédaction" souhaite recevoir votre prose. C'est ici que se trouvent les réponses à vos questions.

Dans la mesure du possible, vous devez nous envoyer vos écrits sur support magnétique (carte, cassette ou disquette). Soyez sans crainte, nous vous retournerons vos biens après copie.

Si vous ne pouvez pas utiliser de support magnétique, ou ne pouvez vous rendre aux réunions, alors et alors seulement faites le sur papier.

Que ce soit sur une feuille de papier, ou sur support magnétique, ne dépassez pas 50 caractères par ligne.

Pour nous épargner du travail, insérez dans votre texte les commandes de formattage suivantes (et non les commandes du formatteur HP) :

"^" centre un titre, par exemple :
^TITRE

"\" (CHR$(92)) marque le début et la fin d'un paragraphe. Par exemple :

\Début de paragraphe exprimant le contenu de vos idées qui, même si vous en doutez, intéressera certains des membres du Club. Surtout si vous vous sentez débutant. Les articles pour débutants écrits par des débutants sont ceux qui manquent le plus. Fin de paragraphe.\

N'oubliez pas de mettre les accents. Utilisez le jeu de caractères Roman8. Les possesseurs de HP71 utiliseront les redéfinitions de touches ci-dessous, ainsi que le fichier CHARLEX listé dans le coin des Lhex en fin de journal.

Jean-Jacques Dhénin (177)

```
DEF KEY 'fW', CHR$(197);      (é)
DEF KEY 'fE', CHR$(193);      (ê)
DEF KEY 'fR', CHR$(201);      (è)
DEF KEY 'fY', CHR$(203);      (ù)
DEF KEY 'fU', CHR$(195);      (û)
DEF KEY 'fI', CHR$(209);      (î)
DEF KEY 'fO', CHR$(194);      (ô)
DEF KEY 'f/', CHR$(92);       (\)
DEF KEY 'fA', CHR$(192);      (â)
DEF KEY 'fS', CHR$(200);      (à)
DEF KEY 'fD', CHR$(205);      (ë)
DEF KEY 'fJ', CHR$(207);      (ü)
DEF KEY 'fK', CHR$(221);      (ï)
DEF KEY 'f*', CHR$(124);      (|)
DEF KEY 'fC', CHR$(181);      (ç)
```

COURRIER DU COEUR

PPC-Paris
B.P. 604
75028 Paris Cedex 01

Vend :
Interfaces vidéo HP82163B (32 colonnes) neuves (dans leur boîte, avec documentation) : 600 F seulement.

Lecteurs de cartes magnétiques HP82400A neufs pour HP-71 (dans leur boîte, avec documentation et 5 cartes magnétiques) : 500 F seulement.

1 lot de cartes magnétiques pour HP-41, HP-67, HP-97 et même HP-65 : 100 F seulement.

1 lecteur de disquettes HP9114A : 4000 F.

1 lecteur de cassettes HP82161A : 2000 F.

Pierre David
33 Bd St Martin
75003 Paris
Tél : (1) 48 87 68 93

Vend :
Imprimante 80 colonnes HP82905B + papier + étiquettes : 2500 F.

Convertisseur HP82166A : 1000 F.

Alain Herreman
2, rue du parc Montsouris
75014 Paris
Tel : (1) 45 89 09 41

Vend :
HP-71 : 2500 F ou HP-71 + Module Forth-Assembleur + Lecteur de cartes : 4500 F. Livres : *Le Forth* (Mc Cabe) 100 F, *Thinking Forth* (Brodie) 150 F, *IDS III* et *IDS Forth-assembleur* : 300 F.

Alain Villatte
Appt 13F
9 rue du Colonel Dominé
75013 Paris
Tél : (1) 45 65 07 80

Vend :
HP-71 (06/84) : 2365 F, interface HPIL (03/85) : 770 F, lecteur de cartes (10/84) : 1046 F, module finance (10/86) : 609 F, module editeur de textes (11/84) : 479 F, module Forth / Assembleur (11/84) : 605 F, Translator Pac (12/86) : 1174 F, imprimante ThinkJet (03/85) : 3228 F. Tout le matériel est en parfait état. L'ensemble : 8500 F.

Pascal Tiennot
34 avenue du Colonel Driant
59130 Lambersart

Recherche :
Photocopies des manuels des modules Xfonctions et Navigation pour HP-41CV égarés suite à un déménagement.

Dominique Marcaillou
29, rue du Moulin
31320 Castanet Tolosan
Tél : 61 81 74 92

Recherche :
désespérément un membre de PPC sur Toulouse pour la collecte et l'échange de programmes ou encore la rédaction d'articles sur support magnétique.

COURRIER DES LECTEURS

Pierre Colignon
24 bis route de Corbeil
91360 Villemoisson sur Orge

Cher Trésorier,

J'ai reçu aujourd'hui, avec mon JPC de mai, un petit mot me rappelant que mon abonnement arrivait à son terme. Je m'empresse donc de faire parvenir au Club ma cotisation, pour ne pas risquer de manquer des numéros qui, si j'en juge par les nombreuses références «*A paraître*» du manuel, que dis-je du manuel, du **magnifique manuel de JPCLEX**, risquent d'être fort intéressants.

En attendant, bravo et merci à vous trois pour ce travail titanesque.

Amicalement,

Pierre Colignon (86)

Bertrand Sliwa
93 avenue de St Mandé
75012 Paris

Monsieur,

Ayant reçu avec le dernier JPC un rappel d'abonnement, je vous prie de bien vouloir trouver avec cette lettre un chèque de 300 F correspondant au renouvellement de mon abonnement au tarif étudiant.

Bien que je n'ai encore participé à aucune des réunions du Club (je rentre chez moi le week end), ni envoyé de programme (ce qui ne saurait tarder !), je tiens à vous féliciter pour le travail accompli :

- les numéros de JPC sont parfaitement présentés (vous me direz, pour le prix...)

- en général, les programmes proposés sont intéressants (c'est à dire ni trop spécialisés, ni trop «classiques») et de très bonne qualité pour mon niveau (lisibles, commentés...).

- enfin, l'idée de créer des Lex regroupant de nouvelles fonctions est tout simplement géniale ; une petite suggestion, cependant : pourquoi ne pas proposer un service de « programmathèque », comme le faisait PPC-T, ce qui éviterait de passer plusieurs quarts d'heure à taper les programmes (je pense surtout aux Lex ou aux longs programmes tels que DIFF).

Je ne vous retiens pas plus longtemps, et je vous souhaite une bonne continuation !

Bertrand Sliwa (314)

Cher Bertrand,

Je suis sensible à tes compliments sur la qualité du Journal. C'est une de nos préoccupations principales.

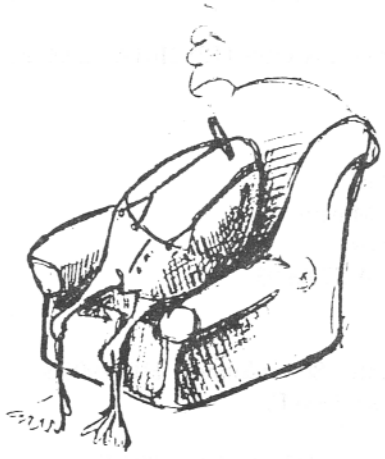
En revanche, je suis moins d'accord avec ton opinion quant au prix du dit Journal. A l'heure actuelle, les cotisations suffisent à peine à payer le Journal et son expédition. Si tu as des suggestions pour diminuer les coûts, écris-moi vite...

La « programmathèque » est un réel problème : les disquettes JPC contiennent tous les programmes parus pour le HP-71 depuis ses débuts dans JPC. La programmathèque existe, encore faut-il avoir le temps de s'en occuper. Nous recherchons quelqu'un qui pourrait prendre en charge l'envoi des programmes.

Je te remercie pour ta lettre.

Amicalement,

Pierre David (37)



HP28

AVEC OU SANS SOMMATIONS !

```
« -> a b c
  « 0 'R' STO
    a b
    FOR i
      i 'I' STO
      c EVAL
      'R'
      STO+
    NEXT
    R
  »
»
```

Personnellement, je l'ai appelé SOM.

Avant, les épreuves de calcul numérique des concours ne nécessitaient qu'une bête calculatrice 4 opérations. Il s'agissait en général de la somme de 3 termes avec 2 décimales exactes. Maintenant, ils veulent 6 décimales et la somme de centaines de termes !

Alors, avec ce programme, vous mettez sur la pile la borne inférieure, la borne supérieure, puis le terme général de la somme. Attention : la variable doit obligatoirement s'appeler I.

Exemple :

```
0 [ENTER]
15 [ENTER]
'1/FACT(I)' [ENTER]
[USER] [SOM]
```

... et vous reconnaissez *e*...

Les dérivations :

Le HP-28C dérive, c'est fantastique. Mais, souvent, il faut «collecter» pour que le résultat ait une bonne tête, d'où la création de DE.

```
« o COLCT COLCT »
```

Les opérateurs :

Allons plus loin. Il existe en mathématique une catégorie curieuse et utile aux physiciens : les opérateurs. Ces fonctions curieuses ont en effet pour objet d'agir sur d'autres fonctions.

Par exemple, le gradiant prend un scalaire et donne un vecteur. Si *V* est le scalaire,

```
          dérivée de V par rapport à x
GRAD(V) = dérivée de V par rapport à y
          dérivée de V par rapport à z
```

Avec DE, la programmation est simple :

```
« -> V
  « V 'X' DE
    V 'Y' DE
    V 'Z' DE
  »
»
```

```
Par exemple :

'X+Y+Z' [ENTER]
[USER] [GRAD]
```

Résultat : 1 1 1.

La divergence est un autre opérateur, qui prend un vecteur et rend un scalaire :
$DIV(a,b,c) = oX(a) + oY(b) + oZ(c)$.

```
« -> x y z
  'oX(x)+oY(y)+oZ(z)'
  EVAL EVAL EVAL
  COLCT COLCT
»
```

Par exemple :

```
'X' [ENTER]
'Y' [ENTER]
'Z' [ENTER]
[USER] [DIV]
```

Et le 3 s'affiche au premier niveau...

Enfin, pourquoi ne pas parler du Laplacien qui prend une fonction (un scalaire) et rend une dérivée seconde assez curieuse :
$LAP(V) = oX(oX(V)) + oY(oY(V)) + oZ(oZ(V))$

Mais la programmation est simple :

```
« GRAD DIV »
```

Par exemple :

```
'X^3+Y^2+Z' [ENTER]
[USER] [LAP]
```

donne 2+6*Y.

J'espère que ces programmes seront utiles au maximum de «taupins».

Pierre Bassaler (269)

Nota : Taupin : forcené des maths qui a choisi les classes préparatoires pour y passer sa belle jeunesse.

Curiosité : éteignez le HP-28C, rallumez-le en maintenant la touche [ON] enfoncée plus d'une seconde, lâchez-la, puis appuyez sur [ ]. Le HP-28C s'éteint tout seul. Curieux, non ?

MODE HORLOGE

Le programme ci-après est commercialisé aux Etats-Unis. Hewlett-Packard nous a permis de le diffuser en France dans JPC.

Nous sommes très contents d'avoir obtenu cette autorisation. Maintenant, à vous de leur montrer ce que vous savez faire !

Le HP-28C contient une horloge à quartz qui fonctionne en permanence. La commande SYSEVAL donne accès à une commande système qui renvoie le l'heure courante sur la pile. En mode hexadécimal, la commande :

```
#123E SYSEVAL
```

retourne un nombre hexadécimal représentant la valeur de l'horloge en 1/8192ème de seconde. Les programmes suivants utilisent cette caractéristique.

TIME : Conversion en secondes

Note : assurez vous que la version du HP-28C est bien *1BB* et que vous êtes bien en mode hexadécimal (touche [HEX] du menu [BINARY]) quand vous entrez ce programme.

```
«
  #123E SYSEVAL              lit l'horloge
  RCWS 40 STWS SWAP B-R      convertit en réel
  8192 /                     convertit en secondes
  SWAP STWS                  restaure la taille de
»                            mot originale
```

Tapez ce programme (à la file, sans les commentaires), puis faites [ENTER] 'TIME [STO].

CLOCK : Visualisation de l'horloge en continu

CLOCK affiche en permanence, après avoir effacé l'écran, une horloge donnant l'heure à la seconde près. CLOCK tourne indéfiniment. Pour l'arrêter et retrouver l'affichage normal, il suffit d'appuyer sur une touche (si vous appuyez sur [ON], l'exécution sera interrompue mais divers objets pourront se trouver sur la pile). Dans la mesure où chaque itération de la routine TIME prend quelques dixièmes de seconde, la mise à jour des secondes sur l'afficheur peut ne pas être précise.

```
«
  4 FIX                      4 décimales
  CLLCD                      efface l'écran
  DO
    TIME 3600 / -HMS
    CORRECT HMS-
    12 MOD 2
    DISP                     horloge au format HMS
  UNTIL KEY END              tant qu'une touche
                             n'est pas pressée
  DROP                       enlève la chaîne
»
```

Tapez ce programme (toujours à la file, sans les commentaires) puis faites [ENTER] 'CLOCK [STO].

Pour calibrer CLOCK, vous devez stocker une facteur de correction dans la variable CORRECT. Pour determiner cette valeur, il faut :

- stocker 0 dans CORRECT (0 'CORRECT [STO]).

- exécuter [CLOCK].

- noter la différence entre l'heure affichée par CLOCK et l'heure exacte.

- calculer le facteur de correction de la manière suivante :
taper le nombre affiché par CLOCK, appuyer sur [ENTER],
taper l'heure exacte (en format *h.ms* sur 12 ou 24 heures) puis
appuyer sur [TRIG] [HMS-] pour calculer le facteur.

- stocker le facteur (*valeur de CLOCK - heure exacte*) dans CORRECT en faisant [USER] 'CORRECT [STO].

CLOCK devrait maintenant afficher l'heure exacte. Vous pouvez ajuster CLOCK à tout moment en modifiant CORRECT. Par exemple, si CLOCK retarde de 2 secondes, il suffit de retrancher (par [HMS-]) .0002 de la valeur de CORRECT.

Remarque importante

Le point d'entrée utilisé par SYSEVAL n'est valable que pour la version 1BB du HP-28C. Pour vérifier que vous avez cette version et pour passer dans la bonne base appuyez sur :

```
[BINARY] [DEC]          base 10
#10 SYSEVAL [ENTER]     affiche la version
[HEX]                   mode hexadécimal
```

SYSEVAL est seulement destiné à la programmation d'applications par Hewlett-Packard. Un usage incorrect de SYSEVAL peut corrompre la mémoire de la machine ou provoquer des pertes de données. N'utilisez SYSEVAL qu'avec les versions de HP-28C spécifiées et comme indiqué dans la documentation.

Ndlr : si vous préférez l'affichage en mode "24 heures", vous pouvez changer le 12 MOD en 24 MOD dans le programme CLOCK.

EXEMPLE D'UTILISATION DE LA PILE OPERATIONNELLE (ACTE II)

Je reçois à l'instant JPC 45 (Juin 87). Je me suis précipité sur l'article de David Dalila sur la HP-28 et la résolution de l'équation du second degré. C'est un vieux problème que j'ai traité dans mon livre *ENTER* en ce qui concerne la HP-41 et la HP-15. C'est vrai qu'il n'a pas grand sens en soi sur une HP-28 (qui possède une fonction propre pour résoudre le problème), mais c'est un exemple qui passionne tous les étudiants.

La solution de David Dalila traite le problème de façon brute et elle a valeur d'exemple. Si on veut optimiser, il est sans doute possible de le faire à partir du programme proposé. Mais ce cas de figure est un exemple type d'une règle non écrite qui veut que les améliorations obtenues par des astuces de programmation soient toujours minimes (même si elles sont agréables pour la vanité du programmeur) alors que les améliorations importantes sont obtenues en modifiant l'algorithme de résolution.

L'algorithme ci-dessous n'est pas de moi, je l'ai trouvé dans les revues américaines dont les auteurs l'avaient sans doute trouvé dans un manuel de math. Il consiste en une réorganisation de l'équation. La HP-28 est particulièrement utile pour suivre ces raisonnements, nous l'allons voir.

La fomule de base est : $(-b \pm \sqrt{b^2-4ac})/2a$. L'astuce consiste à regrouper différement cette équation en effectuant la division par 2a. On obtient : $-b/2a \pm \sqrt{(b/2a)^2-c/a}$. Il n'y a plus que deux termes à calculer : b/2a et c/a.

Voici le programme pour HP-28. Je l'ai fait en cinq minutes et il doit donc être facile à améliorer. Déjà en tant que tel, il ne comporte que 18 instructions et n'utilise que 4 niveaux de pile. Je n'ai pas sous la main l'encombrement mémoire des différentes instructions et la HP-28 ne sait pas nous dire quelle est la taille d'un programme ; je ne peux donc pas chiffrer ni optimiser.

Il est tout à fait inutile que je vous donne un «diagramme de la pile». En effet, il vous suffit de taper : A [ENTER] B [ENTER] C [ENTER] (tapez bien les lettres, pas un chiffre) et de taper chaque instruction en mode direct pour que vous voyez la formule se construire pour vous à l'affichage. N'oubliez pas que votre machine sait traiter le calcul symbolique (si, par hasard, vous avez des valeurs affectées à A, B ou C, purgez-les ou utilisez d'autres lettres). Vous pouvez aussi taper le programme en mode programme en le commençant par HALT et l'exécuter en pas à pas ([SST]).

```
«
  ROT SWAP    peut être remplacé par 3 PICK
  OVER /      calcule c/a
  ROT ROT     range c/a, amène a et b à la place
  DUP +       sans doute mieux que 2 * pour 2a
  /           donne b/2a
  NEG         l'équivalent en mode programme de
              CHS
  DUP SQ      b/2a au carré pour le discriminant
              et -b/2a
  ROT - v     racine du déterminant. N'oubliez
              pas que s'il est négatif, les
              racines existent et sont complexes
  -           première racine
  LAST +      deuxième racine
»
```

Suivre ce programme à l'écran en mode symbolique a un aspect magique : cette machine est fantastique.

Jean-Daniel Dodin (8)

HP41

LE COMPTE EST BON

Ceux qui, pour se détendre, regardent l'émission *Les chiffres et les lettres* seront très contents de pouvoir passer la main à leur HP-71 ou HP-41 quand ils se verront en difficulté.

Le programme pour HP-41 nécessite les modules XFonctions et Time. Le programme pour HP-71, quant à lui, ne nécessite ni module, ni Lex supplémentaire.

Cependant, il est possible que votre HP ne trouve pas la solution alors qu'elle existe : c'est une bogue. Je propose donc de changer l'ordre des plaques et de recommencer.

Le jour viendra peut-être où ce programme sera écrit sans bogue, en assembleur ou en Forth.

Tony Guilloux (235)

NDLR : le programme pour HP-71 se trouve listé avec les autres programmes Basic pour HP-71.

```
01*LBL "LCEB"
53  PSIZE

04*LBL 15
CF 00  CLRG  FIX 0  SIGN  STO 00  STO 40  STO 08
,02505  STO 07

14*LBL 20
"PLAQUE"  ARCL 40  "|- - ?"  PROMPT  INT  ABS
X=0?  GTO 20  STO IND 40  ISG 40  CLD  ISG 07
GTO 20

28*LBL 21
"CIBLE="  PROMPT  INT  ABS  X=0?  GTO 21  STO 40
1,011008  REGMOVE  CLX  STO 00  STO 07  SETSW
RUNSW

43*LBL 04
ISG 09  CLX  CLX  STO 10  RCL 09  20  +  -1
STO IND Y

53*LBL 00
RCL 09  20  +  ISG IND X  "TEST 0"  RCL IND X  10
+  RCL IND X  ,1  X=Y?  GTO 12

66*LBL 01
RCL 09  30  +  ISG IND X  "TEST 0"  7  RCL IND Y
X=Y?  GTO 01  10  +  RCL IND X  ,1  X=Y?  GTO 06
R^  RCL IND X  10  ST- Z  CLX  RCL IND Z  X=Y?
GTO 06

90*LBL 07
FS? 49  OFF  30  RCL 09  +  RCL IND X  RCL IND X
STO 37  1  X=Y?  GTO 06  10^X  R^  +  STO 38
DSE X  RCL IND X  STO 19  19  ST- Z  RCL IND Z
RCL IND X  STO 29  RCL 37  X=Y?  GTO 06  RCL 19
RCL 29  -  X<=0?  GTO 06  RCL 37  /  FRC  X=0?
GTO 11  RCL 19  RCL 29  X=0?  GTO 06  +  RCL 37  /
FRC  X=0?  GTO 14

137*LBL 06
30  RCL 09  +  RCL IND X  6  X>Y?  GTO 01  ,
STO IND T

147*LBL 12
20  RCL 09  +  RCL IND X  6  X>Y?  GTO 00  ,
STO IND T  LASTX  1  X=Y?  GTO 09

161*LBL 10
DSE 09  20  RCL 09  +  RCL IND X  RCL IND X  X<>Y
10  +  X<>Y  STO IND Y  LASTX  R^  +  RCL IND X
RCL IND X  X<>Y  10  +  X<>Y  STO IND Y  GTO 06

184*LBL 11
"+"  GTO 13

187*LBL 14
"-"

189*LBL 13
LASTX  STO IND 38  RCL 09  46  +  ASTO IND X  6  -
RCL IND X  X=0?  GTO 08  1  X=Y?  GTO 08  20
RCL 09  +  RCL IND X  7  X=Y?  GTO 02  X<>Y  10  +
,1  STO IND Y

216*LBL 02
30  RCL 09  +  RCL IND X  7  X=Y?  GTO 04  X<>Y
10  +  ,1  STO IND Y  GTO 04

230*LBL 08
STOPSW  70  STO 10

234*LBL 26
CF 00  40  RCL 09  +  RCL IND X  STO 19  X=0?
GTO 05  STO 37  1  X=Y?  SF 00  10^X  ST- Z
RCL IND Z  RCL IND X  STO 29  X=0?  GTO 05  ST* 37
1  X=Y?  SF 00  FS?C 00  GTO 27  XEQ 29  ISG 10
```

```
262*LBL 27
RCL 37  RCL 40  X=Y?  GTO 03  46  RCL 09  +
RCL IND X  STO 19  26  ST- Z  RCL IND Z  RCL IND X
STO 29  X=0?  GTO 05  39  RCL 09  +  RCL IND X
STO 00  RCL 29  X=Y?  GTO 05  RCL 00  RCL 37  X=Y?
GTO 05  SF 00  XEQ 29  RCL 40  RCL 00  X=Y?
GTO 03  ISG 10

298*LBL 05
DSE 09  GTO 26

301*LBL 03
CLA   FIX 4  RCLSW  ATIME  PROMPT  "AUTRE CAS ?"
AVIEW


309*LBL 28
GETKEY  X=0?  GTO 28  42  X=Y?  GTO 15  "BYE"
PROMPT  RTN


319*LBL 09
STOPSW  "NO SOLUTION"  PROMPT  GTO 03


324*LBL 29
CLA   FC? 00  ARCL 19  FS? 00  ARCL 37  FC? 00
"|-*"  FS? 00  ARCL 19  ARCL 29  "|-="  FC? 00
ARCL 37  FS? 00  ARCL 00  PROMPT  END
```

CLASSEMENTS SUR HP-41

Voici un petit programme pour HP-41 sans prétention que j'ai établi un peu pour mon plaisir, indépendamment de nombreuses applications professionnelles.

Le but est d'introduire des nombres, de les classer et d'afficher les résultats en continu.

Classement simple

Il faut d'abord exécuter CLRG, puis XEQ "CLASS" pour entrer une première série de nombres.

Chaque nombre est rentré à l'affichage de "NOMBRE", puis validé par appui sur [R/S].

On peut ensuite revoir ces nombres en faisant XEQ "READ". Si on a branché une imprimante, la liste est imprimée. Si on a un branchement vidéo par interface, la liste est affichée sur l'écran qu'on peut effacer en faisant XEQ "LIBRE". Si on n'a ni imprimante, ni interface vidéo, on peut lire les nombres sur l'écran du HP-41 en ayant pris soin de remplacer la ligne 8 du programme READ par 08 PSE. On peut renouveler l'opération autant de fois que l'on désire.

Classement compétition

Il faut d'abord exécuter CLRG pour effacer les registres.

Le but est de classer les performances de concurrents. Chaque concurrent porte un dossard numéroté (deux chiffres). Ainsi, le coureur numéro 25 faisant une performance 1214 sera codé comme le nombre 1214.25. Le coureur 06 faisant une performance 932 sera codé comme 932.06.

On lit et on affiche les nombres comme ci-dessus. A chaque nouvelle arrivée, on saisit la nouvelle donnée avec le programme CLASS, par [R/S].

Le tableau des résultats donne le rang, le numéro et le score de chaque concurrent.

Ce système peut convenir pour une course contre la montre, une compétition de ski, un jumping, un concours d'entrée à une école, etc.

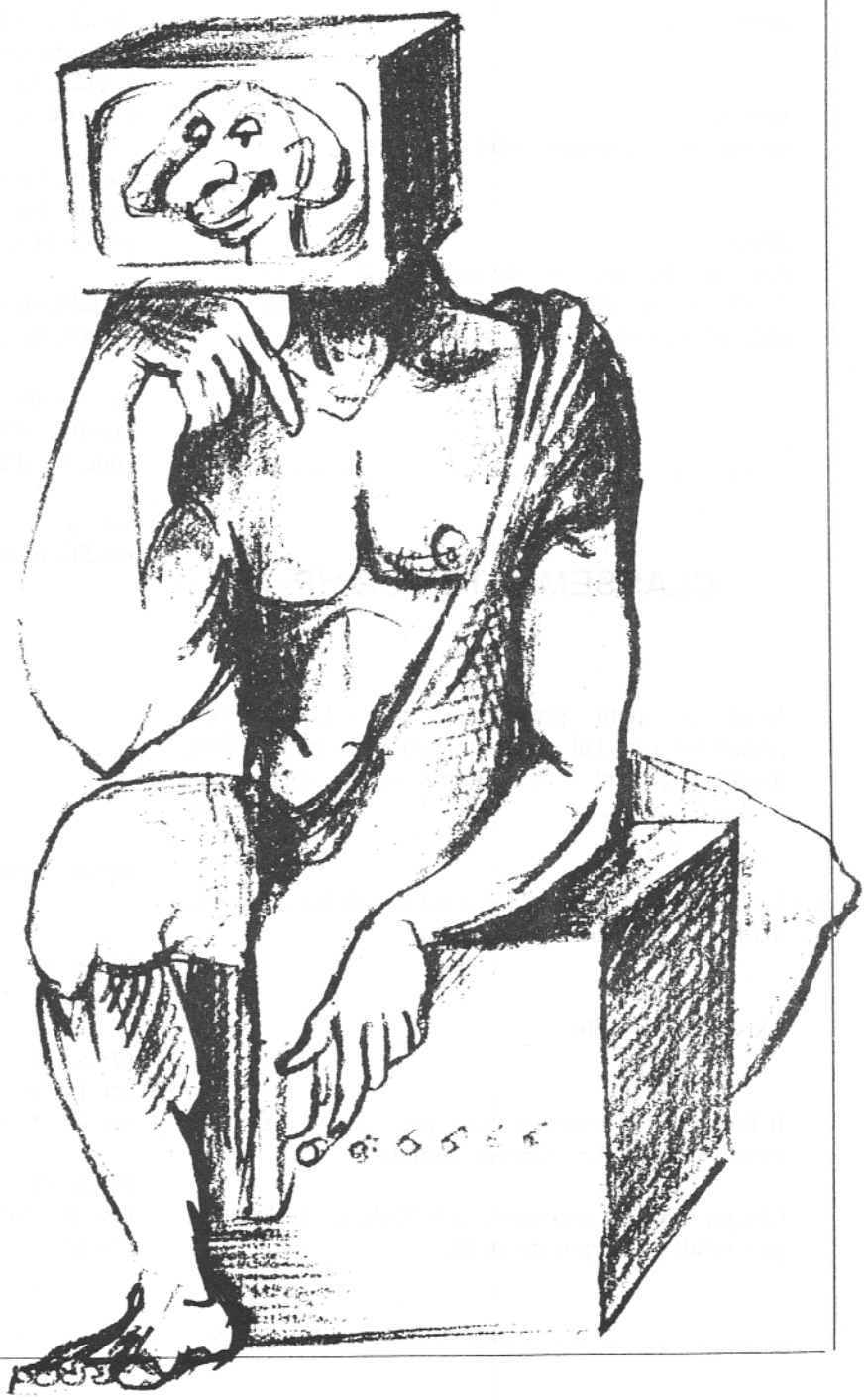
S'il y a plus de 99 concurrents, une légère modification du programme est à prévoir...

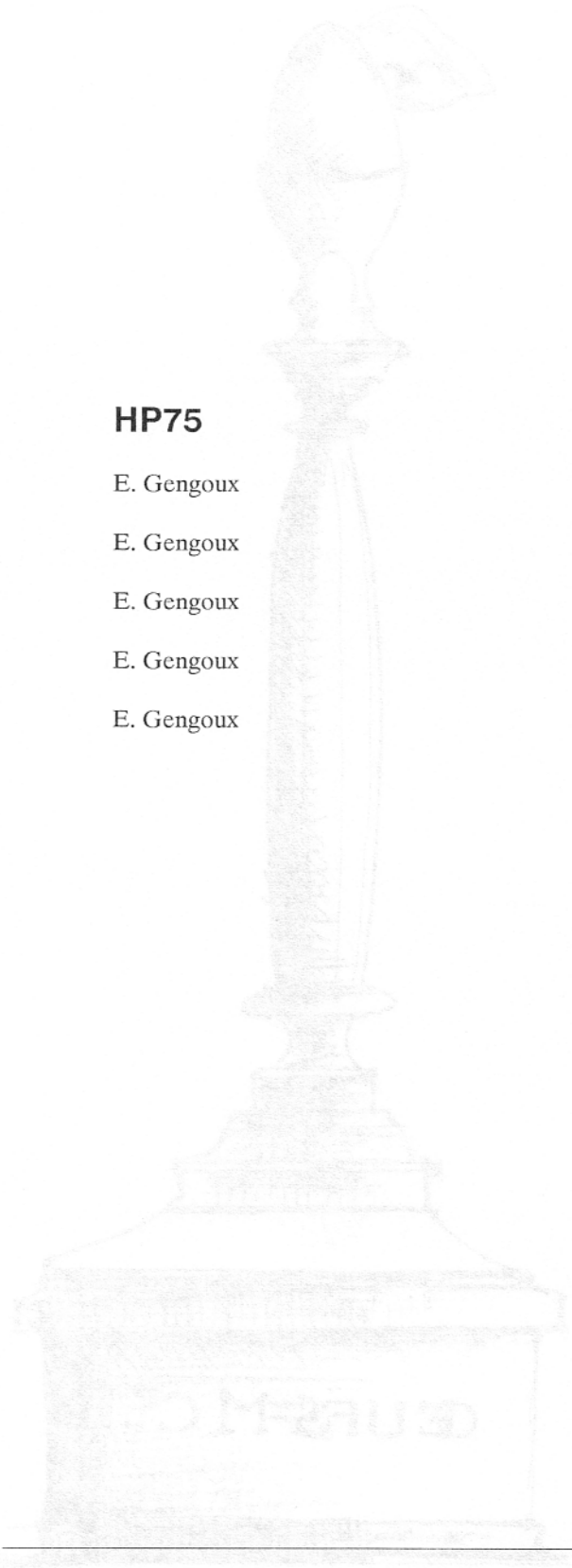
Amicales salutations

René Dine (293)

```
01*LBL "CLASS"


02*LBL 04
3  STO 00  "NOMBRE"  PROMPT

07*LBL 02
RCL IND 00  X=0?  GTO 00  X>Y?  GTO 03  RDN
RCL 00  1  +  STO 00  RDN  GTO 02

20*LBL 03
RDN  X<> IND 00  RCL 00  1  ST+ 00  RDN  RDN
GTO 02
```

```
29*LBL 00
RDN  X<> IND 00  RCL 00  STO 01  GTO 04  END

01*LBL "SCORES"
"        RESULTA"  "|-TS"  PRA  ADV
"   RANG  NUM  "  "|-SCORES"  PRA  ADV  RCL 01
,002  +  STO 00  0  STO 02

16*LBL 00
RCL IND 00  4  SKPCHR  1  ST+ 02  RCL 02  ACX  RDN
3  SKPCHR  RDN  RDN  RDN  ENTER^  FRC  100  *  ACX
5  SKPCHR  RDN  RDN  INT  ACX  PRBUF  DSE 00
GTO 00  END

01*LBL "READ"
RCL 01  ,002  +  STO 00

06*LBL 00
RCL IND 00  PRX  DSE 00  GTO 00  END

01*LBL "LIBRE"
27  ACCHR  69  ACCHR  END
```





HP75

PETITS UTILITAIRES
MINI-FONCTIONS VISICALC

Voici quelques utilitaires sans grande prétention, mais qui peuvent bien simplifier l'existence quand on utilise fréquemment, comme c'est mon cas, VisiCalc.

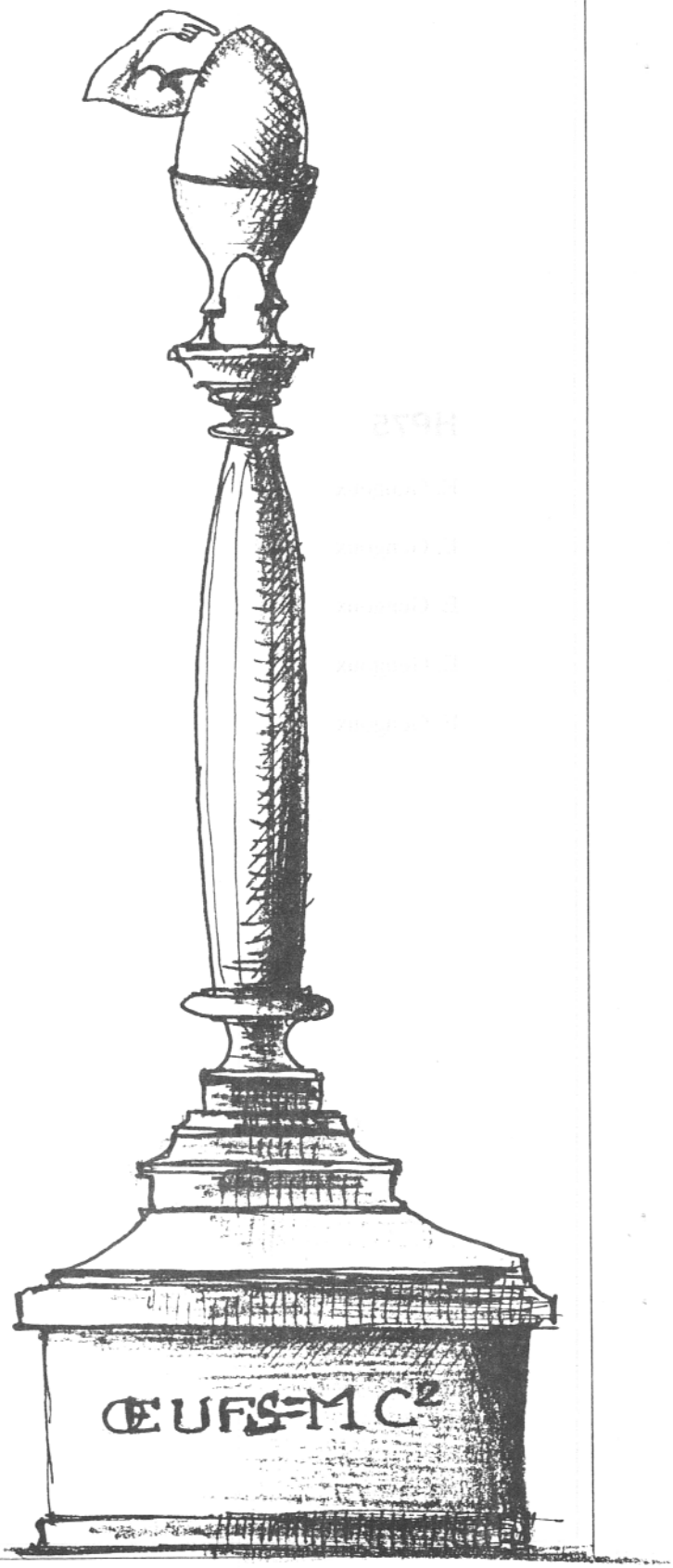
Les deux premiers, PCT9 et DPCT, sont les fonctions de calcul de valeur d'un poste en pourcentage d'un total, et de progression en pourcentage. La syntaxe est proche de celle des fonctions `%T` et `Delta%` de la HP-12 (la dernière s'appelle aussi `%CH` sur la HP-41).

Les deux derniers, VCCLFORM et VCLISTF, servent l'un à faire le ménage (effacement des formules dans tout ou partie de feuille sans toucher aux labels ou aux valeurs), l'autre à lister les formules avec les en-têtes par défaut. Ceux-ci qui sont souvent plus lisibles que les en-têtes utilisateurs, car la fonction `/PF` (Print Formulas) de VisiCalc les tronque à 8 caractères, et la présence des crochets et autres "^" n'arrange rien ! L'utilisation est entièrement guidée par menu, il suffit donc de charger le programme et la feuille de travail et de faire `RUN <programme>`...

Pour leur part, DPCT et PCT9 peuvent servir entre autres à analyser un compte d'exploitation ou un bilan en pourcentages (structure), ou à traduire des évolutions d'un exercice à l'autre. C'est moins spectaculaire que la classique fonction `IRR`, mais d'un usage plus courant... Et ça montre bien l'extraordinaire souplesse des fonctions dites étendues de ce VisiCalc, tellement plus puissantes que des «macros»... Elles s'utilisent comme toute autre fonction étendue ou formule, donc à partir de n'importe quelle case ; la syntaxe étant rappelée au début du listing, je ne vois guère d'autre chose à ajouter...

Bonne exploration. Pour tout problème à base de tableurs, le HP-75 demeure meilleur que le HP-71, même avec le module *WorkBook*, pourtant fort bien fait.

Eric Gengoux (108)



ASSEMBLEUR

BASIC

LE COIN DES LHEX

L'ASSEMBLEUR DU HP-71
(ACTE II)

Voici donc la suite de notre initiation à l'assembleur. Dans l'article précédent, nous avions vu ce qu'est l'assembleur et comment en faire. Aujourd'hui, nous allons voir plus en détail le microprocesseur du HP-71, ce qu'il permet de faire, quels sont ses registres et ses instructions.

Tout d'abord, il faut rappeler que contrairement à la plupart des ordinateurs du marché, l'unité de mémoire du HP-71 n'est pas l'octet (ou byte en anglais) mais le demi octet (encore appelé quartet ou nibble en anglais). Le microprocesseur du HP-71 peut ainsi adresser une mémoire maximale de 1 million de quartets soit donc 512 ko que l'on partage comme on veut entre Ram et Rom. Je rappelle que la Rom (Read Only Memory) est la partie de mémoire que l'on peut juste lire. Elle contient, dans le HP-71, le système d'exploitation et le Basic (en tout, 64 Ko), les modules Math, Forth et autres sont constitués de ce type de mémoire. La Ram (Random Acces Memory) est la mémoire utilisée pour le stockage des données ou programmes de l'utilisateur. Le HP-71 possède 17,5 Ko de ce type de mémoire que l'on peut étendre par ajout de modules 4 Ko de chez HP ou par ceux de HHP ou CMT.

Entrons maintenant dans le vif du sujet...

Comme je le disais la dernière fois, le HP-71 a été optimisé pour effectuer des opérations mathématiques. Il y a deux types de registres dans ce microprocesseur : ceux utilisés pour les opérations arithmétiques et ceux utilisés pour les programmes et la gestion du système.

Registres arithmétiques :

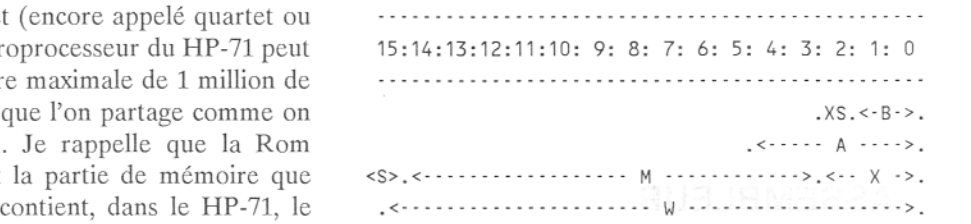
Il existe 9 registres arithmétiques de 64 bits chacun dont 4 principaux qui sont A, B, C et D. Ce sont les registres que l'on utilise le plus souvent. Il faut signaler que seuls les registres A et C permettent d'accéder à la mémoire du HP-71. Les 5 registres restants sont des registres de «brouillon». On ne peut y faire que des lectures ou des écritures. Ils permettent donc de stocker des valeurs intermédiaires lors de calculs. Ces registres se nomment R0, R1, R2, R3 et R4. Attention : une partie de R4 est réservée au système, il faut donc faire attention en l'utilisant.

Voyons maintenant les autres registres...

Deux registres de 20 bits permettent l'accès à la mémoire du 71, ce sont D0 et D1. Les registres IN (16 bits) et OUT (12 bits) permettent de gérer le clavier. ST contient 16 drapeaux (flags) de statut. HW contient 4 bits de gestion du système. Enfin, il existe un registre P de 4 bits dont je parlerai plus loin et, bien sûr, le registre PC (pour Program Counter) qui contient l'adresse de la prochaine instruction à exécuter.

Les 4 registres de travail sont séparés en plusieurs champs, on peut donc travailler sur une partie seulement d'un registre.

L'organisation d'un registre est la suivante :



Ainsi, on peut manipuler dans un seul registre plusieurs objets indépendamment. Le champ WP est un champ spécial qui s'étend du quartet 0 au quartet dont la valeur est stockée dans P. Celui-ci contient une valeur de 0 à 15, soit 0 à F en hexadécimal (ce qui correspond bien à 4 bits). Il existe également un champ nommé P qui correspond au quartet pointé par P. Voici maintenant les significations des abréviations employées :

P	chiffre pointé par P.
WP	de la position 0 jusqu'à celle pointée par P.
XS	signe de l'exposant.
X	exposant et son signe.
M	mantisse.
B	exposant ou octet (Byte en anglais).
W	mot entier (Word).
A	champ adresse.

Voyons maintenant comment utiliser ces différents registres.

Pour cela, la meilleure solution est de les employer en écrivant un programme en assembleur. Comme nous l'avons vu précédemment, il suffit d'écrire un programme source Bin. Pour écrire un tel fichier, il suffit d'employer EDTEXT (fourni dans le module Forth / Assembleur). Pour notre premier Bin, nous allons utiliser une routine écrite par HP et contenue dans la Rom système du HP-71. Cette routine fait partie du module BEEP de cette Rom et s'appele BP+C. Pour savoir comment l'utiliser, il suffit de se référer aux *IDS II*, page 11-19 ou *IDS III*, page 4 du module BEEP. Cette routine se trouve à l'adresse 0EB40.

D'une manière générale, pour pouvoir utiliser une routine système il suffit de savoir quels sont ses paramètres d'entrée et ceux de sortie. La routine est donc vue comme une boîte noire et il n'est pas nécessaire de savoir comment elle fonctionne pour l'utiliser. Bien sûr, il peut être intéressant de l'étudier. Nous n'étudierons pas ici la routine BP+C. Il suffit de savoir que cette routine exécute un beep de fréquence et de durée spécifiées exactement comme l'ordre BEEP du Basic. Voici donc ce que l'on trouve concernant cette routine dans les *IDS* :

Entrée :
C[A] = durée en msec. (en hexa.)
D[A] = fréquence en Hz. (en hexa).
Mode Hexa.

Sortie :
Mode hexa.

Registres utilisés :
A,B,C,D,D0,P

Comme on pouvait s'en douter cette routine ne renvoie aucune valeur. Il suffit donc de mettre les valeurs desirées dans les champs A des registres C et D pour obtenir le beep souhaité.

Nous pouvons donc écrire le fichier texte suivant :

```
        BIN     'MYBEEP' indique à l'assembleur
*                        de créer un fichier
*                        de type BIN.
        CHAIN   -1       indique l'absence
*                        de sous-programme.

ENDBIN  EQU     #0764B   retour d'un BIN.
BP+C    EQU     #0EB40   exécute un beep

freq    EQU     1000     la fréquence sera 1000 Hz
duree   EQU     500      la durée 500 msec.

debut   SETHEX           passage en mode hexadécimal
        P=      0        indique de charger C en
*                        partant du nibble 0.
        LC(5)   freq
        D=C     A        transfère C danc D.
        LC(5)   duree

saut    GOSBVL  BP+C     exécute le beep.

fin     GOVLNG  ENDBIN   retourne au BASIC

        END              fin du fichier source
```

Avant d'étudier ce programme voyons de quoi se compose un programme source assembleur.

Une ligne d'un fichier comme celui qui précède se compose de quatre parties appelées encore champs. Ces champs sont (le jeu de mot est fort, non ?) : le champ étiquette, le champ instruction, le champ opérande et le champ commentaire. Pour l'assembleur, un espace suffit pour délimiter les différents champs mais pour des raisons de plus grande lisibilité des fichiers sources, il vaut mieux aligner les champs les uns au dessus des autres (il existe d'ailleurs un LEX qui fait cela automatiquement : ASFLEX). Voir à ce propos le paragraphe *Line Format* (page 46) du manuel du module Forth / Assembleur.

Voyons alors comment fonctionne notre programme...

Les lignes 1 et 2 indiquent à l'assembleur de créer un fichier de type Bin qui n'utilisera pas de sous programmes. Toutes les lignes commençant par un "*" sont ignorées par l'assembleur et sont des commentaires. Les 4 lignes suivantes comportent la directive EQU. On appelle directive une instruction qui ne concerne que le programme d'assemblage et ne donnera donc aucun code objet. Cette directive assigne à un identificateur une valeur. Ainsi dans la suite du texte, ENDBIN sera remplacé par sa valeur hexadécimal 0764B et de même pour BP+C, *duree* et *freq*.

Le programme commence à partir de l'étiquette *debut*. L'instruction SETHEX positionne le CPU en mode hexadécimal et P=0 initialise P. On charge alors le champ A du registre C avec la valeur de la fréquence que l'on recopie ensuite dans D[A]. Il faut savoir que l'instruction LC(*n*) *valeur* charge dans C les n chiffres de poids faible en commençant au quartet pointé par P et en continuant vers la gauche. Dans notre cas, il faut donc mettre P à zéro avant de commmencer le chargement. Il reste alors à mettre la durée dans C[A] pour avoir ainsi les bons paramètres pour l'appel de la routine BP+C. L'instruction GOSBVL (pour *GOSub Very Long*) appelle cette routine.

Pour finir, il suffit de partir en appellant la routine ENDBIN qui se charge de revenir au Basic avec l'instruction GOVLNG (pour *GO Very LoNg*). Pour indiquer à l'assembleur que son travail est terminé, on place la directive optionnelle END. Les instructions GOSBVL et GOVLNG sont équivalentes respectivement à GOSUB et GOTO, mais permettent des branchements très long nécessaires pour pouvoir appeler une routine système à partir d'un programme qui peut résider n'importe où en mémoire.

Voilà alors notre programme terminé, il reste à l'assembler. Pour cela, il faut passer en Forth (en utilisant l'ordre FORTH du Basic) et appeler le

programme d'assemblage en tapant " MYBEEP" ASSEMBLE. S'affiche alors le mot "Pass 1" suivi de points qui indiquent l'évolution du travail puis arrive la deuxième phase de l'assemblage avec "Pass 2" et toujours les points qui se baladent à l'écran. Si l'assembleur détecte une erreur (ou plusieurs), il affiche un message avec le numéro de la ligne qui lui pose des problèmes.

Une fois ceci terminé, on n'a plus qu'à revenir au Basic en tapant BYE, puis à essayer ce programme en faisant RUN MYBEEP.

Voilà... C'est fini pour aujourd'hui. J'espère avoir été suffisamment clair. N'oubliez pas que vos remarques seront les bienvenues. Je vous dis donc à la prochaine fois en vous laissant réfléchir sur le Lex suivant...

```
       BIN    'MYBEEP2'
       CHAIN  -1

ENDBIN EQU    #0764B
BP+C   EQU    #0EB40

duree1 EQU    500
freq1  EQU    1000
freq2  EQU    2000
freq3  EQU    3000

debut  SETHEX

       P=     0
       LC(5)  freq1
       D=C    A
       LC(5)  duree1
       GOSBVL BP+C

       P=     0
       LC(5)  freq2
       D=C    A
       LC(5)  duree1
       GOSBVL BP+C

       P=     0
       LC(5)  freq3
       D=C    A
       LC(5)  duree1
       GOSBVL BP+C

fin    GOVLNG ENDBIN

       END
```

Jacques Baudier (3*2^6)

CONVERSION DE CARACTERES

L'article ci-dessous a été publié pour la première fois dans STaK *(numéro 5), le Journal du Club finlandais. Son auteur, Tapani Tarvaïnen nous l'a traduit en anglais, et Janick Taillandier l'a traduit pour* JPC *en français. Ceux qui ont l'Eprom JPC ont déjà pu apprécier ce Lex, puisque Tapani nous l'a donné pour y être inclus. Merci Tapani !*

L'existence de différents jeux de caractères pour les ordinateurs ou les imprimantes semble être un problème éternel. En particulier, les caractères n'appartenant pas au jeu ASCII standard comme Ä ou Ö posent le plus de problèmes. Le HP-71 utilise ses propres codes, les imprimantes, par exemple la ThinkJet, utilisent les leurs et d'autres ordinateurs des caractères encore différents. Ceci nécessite un programme pouvant convertir des chaînes de caractères et / ou des fichiers entiers d'un jeu de caractères dans un autre. Ceci est particulièrement vrai pour nous qui utilisons des langues autres que l'anglais.

La manière la plus simple de représenter la conversion de chaînes est probablement la suivante : soit une chaîne (O$ dans la suite) contenant les caractères qui doivent être changés et une autre (N$), de même longueur, contenant les caractères de remplacement, dans le même ordre. Par exemple, si nous convertissons les caractères ÄäÖö du HP-71 en leur équivalent ThinkJet, nous aurons : O$="ÄäÖö" et N$=CHR$(204)&CHR$(206)&CHR$(216)&CHR$(218).

J'ai d'abord essayé avec le petit programme Basic suivant :

```
10 SUB MAP(A$,O$,N$)
20 FOR I=1 TO LEN(O$)
30 P=POS(A$,O$[I,I])
40 IF P THEN A$[P,P]=N$[P,P] @ GOTO 30
50 NEXT I
60 END
```

Ceci peut être quelque peu simplifié grâce à STRINGLX :

```
10 SUB MAP(A$,O$,N$)
20 P=MEMBER(A$,O$)
30 IF P THEN A$[P,P]=N$[POS(O$,A$[P,P])][1,1] @
   GOTO 20
40 END
```

Pour convertir tout un fichier texte (utilisez d'abord TRANSFORM si vous voulez travailler sur un fichier Basic) vous pouvez procéder ainsi :

```
100 FOR I=0 TO FILESZR(F$)-1 @ READ #1,I;R$
110 CALL MAP(R$,O$,N$) @ REPLACE #1,I;R$ @ NEXT I
```

Ceci présuppose que F$ contient le nom de fichier et que ASSIGN #1 TO F$ a été exécuté. Si vous ne disposez pas de EDLEX, vous pouvez faire (avec sortie dans un deuxième fichier) :

```
100 ON ERROR GOTO 120
110 READ #1;R$ @
    CALL MAP(R$,O$,N$) @
    PRINT #2;R$ @
    NEXT I
120 OFF ERROR
```

Ceci fonctionne très bien mais très lentement : dès que le fichier est de taille correcte, cela semble durer une éternité.

La fonction SEARCH présente une toute autre approche, particulièrement lorsqu'il s'agit de convertir tout un fichier :

```
100 FOR I=1 TO LEN(O$) @ S$=O$[I,I] @ T$=N$[I,I]
110 L=0 @ C=0 @ E=FILESZR(F$)
120 S=SEARCH(S$,C,L,E,1) @ IF NOT S THEN 160
130 C=1000*FP(S)
140 READ #1;R$ @ R$[C,C]=T$
150 REPLACE #1,IP(S);R$ @ GOTO 120
160 NEXT I
```

Ceci serait bien n'était la manière dont SEARCH traite le caractère "\" qui correspond à Ö dans les jeux de caractères les plus courants en Finlande. Corriger ceci est difficile car SEARCH ne trouve "\" que triplé, et encore pas en fin de ligne. De plus, il n'y a pas de quoi être fier de la vitesse du programme.

Arrivé à ce point, j'ai décidé d'abandonner Basic et de me tourner vers Forth. Dans ce langage, vous traitez les chaînes comme des vecteurs de caractères. En Basic, la chaîne doit être préalablement copiée sur la pile ce qui est plus sûr, mais aussi plus lent. Un fichier peut être traité directement sans que l'enregistrement soit d'abord copié dans un variable. Il est alors possible d'envisager un algorithme complètement différent de ceux qui précèdent : créer un vecteur de 256 octets appelé CHARMAP et dont le $n^{ème}$ élément contient la valeur de remplacement de ce caractère (n = 0 à 255). Pour convertir un caractère, il suffit de l'utiliser comme index dans le vecteur. La plupart des caractères restant inchangés, il suffit d'avoir CHARMAP(N) = N.

Forth, comparé à Basic, a moins de fonctions de base : pour écrire un programme, vous devez commencer à un niveau plus bas par des opérations simples. Cependant, il s'est avéré que l'implémentation de l'algorithme décrit plus haut a été facile et que le code n'est pas très gros.

```
HERE HEX

( la longueur de l'enregistrement n'a pas ses    )
( octets inversés                                )

: RECLEN ( addr -- length )
  DUP 2- @ FF00 AND SWAP 2+ C@ OR ;

( le caractère n contient le code de remplacement)

CREATE CHARMAP 100 ALLOT

( initialise CHARMAP : CHARMAP[n] = n            )

: CLEAR-MAP
  CHARMAP 100 0
  DO
    DUP I 2* + I SWAP C!
  LOOP
  DROP ;

: INITIALIZE-MAP (addr1 addr2 len )
  CLEAR-MAP
  2* 0
  DO
    OVER I + C@ 2* CHARMAP + OVER I +
    C@ SWAP C! 2
  +LOOP
  2DROP ;

: MAPCHAR ( c1 -- c2 )
  2* CHARMAP + C@ ;

: MAPSTRING ( addr len )
  ?DUP
  IF
    2* 0
    DO
      DUP I + DUP C@ MAPCHAR
      SWAP C! 2
    +LOOP
  THEN
  DROP ;

( ajoute 1 si impair                             )
: UPTOEVEN ( addr1 -- addr2 )
  1+ 2/ 2* ;

: NEXTREC ( addr1 len -- addr2 )
  UPTOEVEN 2* + 4 + ;
```

```
: NOT-EOF ( len -- flag )
  FFFF <> ;

: MAPTEXT (addr )
  BEGIN
    DUP RECLEN DUP NOT-EOF
  WHILE
    OVER 4 + OVER MAPSTRING NEXTREC
  REPEAT
  2DROP ;

DECIMAL

: SKIPHEADER ( fha -- addr )
  37 + ;

: FTYPE ( fha -- n )
  16 + 4N@ ;

: TEXT? ( fha -- flag )
  FTYPE 1 = ;

: FINDTF ( str -- fha )
  FINF DUP 0= ABORT" File not Found"
  DUP TEXT? NOT ABORT" Non-TEXT File" ;

: MAPRECS ( str1 str2 addr )
  >R DROP SWAP INITIALIZE-MAP R> MAPTEXT ;

: MAPFILE ( str1 str2 str3 )
  FINDTF SKIPHEADER MAPRECS ;

20 STRING FILENAME
96 STRING OLDCHARS
96 STRING NEWCAHRS

: ASKSTRING ( str1 str2 )
  TYPE QUERY 13 WORD COUNT 2SWAP S! ;

: ?MAP
  FILENAME " Fichier ? " ASKSTRING
  OLDCHARS " caractères à changer ? " ASKSTRING
  NEWCHARS " remplacés par ? " ASKSTRING
  OLDCHARS NEWCHARS FILENAME MAPFILE ." Fini " ;

HERE SWAP - . .( nibs )
```

Ceci peut être appelé depuis Basic par FORTHX "MAPFILE",O$,N$,F$ où O$, N$ et F$ ont la même signification que précédemment. C'est nettement plus rapide que les programmes Basic vus plus haut. Le code fonctionne aussi bien avec le module Forth / Assembleur qu'avec le Translator Pac. Cependant, comme FORTHX n'a pas le même token dans les deux module (pourquoi ?), un programme Basic ne fonctionnera qu'avec le module pour lequel il a été écrit. VER$ peut aider à determiner quel module est présent, mais tout ceci est désagréable.

En plus de l'incompatibilité entre les deux modules Forth, il y a un autre inconvénient à cette solution : il faut un fichier FORTHRAM (ou FTH41RAM), ce qui limite la mémoire disponible. Comme cet algorithme me paraissait facile à coder en assembleur, j'ai donc décidé d'en faire un fichier Lex.

La première version de MAPLEX ne contenait qu'un mot-clé, MAP, utilisé ainsi : MAP <fichier>,O$,N$ où *<fichier>* est un spécificateur de fichier et *O$* et *N$* ont la même signification que précédemment. Il est ensuite apparu que l'ajout d'une fonction travaillant sur une chaîne de caractères ne serait pas très coûteux. Ainsi est apparu MAP$ dont la syntaxe est MAP$(<chaîne>,O$,N$) où *<chaîne>* est la chaîne de caractères à convertir. C'est sous cette forme que MAPLEX a été publié pour la première fois dans *STaK* numéro 5.

Par la suite, Stefano Tendon a suggéré de modifier MAP afin de permettre de ne convertir qu'une partie du fichier et de permettre l'utilisation du numéro de canal à la place du spécificateur de fichier. L'ajout de numéros d'enregistrements était facile et une amélioration naturelle. J'ai également envisagé de permettre de spécifier les colonnes de début et de fin : est-ce que ceci intéresserait quelqu'un ? Quelle syntaxe utiliser ? Comment cette option devrait-elle interagir avec la spécification de numéros d'enregistrements ?

Le problème du numéro de canal était plus difficile. L'appréciation de la meilleure solution (rapide, facile à utiliser) dépend de l'utilisation que vous envisagez. J'ai donc décidé d'offrir les deux options. Cette solution me paraît tellement évidente que je me demande pourquoi elle n'est pas plus courante. Il y a une bonne raison à cela : toute fonction accédant à un fichier par son nom doit soit s'assurer qu'un canal éventuellement ouvert sur ce fichier reste valide, soit fermer le canal comme le font PURGE ou TRANSFORM. La première condition est assez difficile à satisfaire sauf dans le cas d'opérations qui n'affectent pas le fichier (comme CAT, COPY ou FILESZR). Il est par ailleurs difficile de savoir si un canal est ouvert sur un fichier connaissant son nom et de trouver son FIB. Fort heureusement, le fonctionnement de MAP est tel qu'il ne requiert pas de changement au FIB. Il ne fait que remplacer des caractères sans bouger quoi que ce soit.

La nouvelle syntaxe de MAP est donc :

```
MAP <fichier>,
    <source>,<destination>,
    [<debut>[,<fin>]]
```

ou bien :

```
MAP #<canal>,
    <source>,<destination>,
    [<debut>[,<fin>]]
```

où les crochets (caractères [et]) désignent un champ optionnel.

Pour terminer, quelques exemples d'utilisation de MAP :

```
MAP F$,'abcdefghijklmnopqrstuvwxyzäö',
       'ABCDEFGHIJKLMNOPQRSTUVWXYZÄÖ'
```

convertit tout un fichier en majuscules.

```
DIM L$[128],U$[128] @
FOR I=0 TO 127 @
L$[I+1]=CHR$(I) @
U$[I+1]=CHR$(I+128) @
NEXT I @
MAP F$,L$&U$,U$&L$,B,E
```

fera passer en vidéo inverse les enregistrements B à E du fichier F$ (si l'interface vidéo utilise le bit de poids fort pour reconnaître ce type d'attribut. La même commande remettera le fichier dans son état original. Le bit de poids fort est inversé et non pas simplement mis à 1.

C'est tout ! N'hésitez pas à m'adresser vos suggestions et commentaires (de préférence en finnois ou en anglais, à la rigueur en allemand ou en suédois).

Tapani Tarvainen (328)

NDLR : si votre finnois n'est plus ce qu'il était, vous pouvez toujours nous envoyer vos remarques. Nous les traduirons (mais pas en finnois !) et les ferons parvenir à l'Auteur.

```
       LEX    'MAPLEXB'
       ID     #E1
       MSG    0
       POLL   POLLH
       ENTRY  MAP$
       CHAR   #F      fonction
       ENTRY  MAP
       CHAR   #D      statement
       KEY    'MAP$'
       TOKEN  76
       KEY    'MAP'
       TOKEN  77
       ENDTXT
***********************************************
* EQUates

* Points d'entree
=?PRFI+ EQU #17380
=ARGERR EQU #0BF19
=CHKmem EQU #012C7
=COMCK  EQU #036CD
=COMCK+ EQU #032AE
=DROPDC EQU #05470
=EXPEX- EQU #0F178
=EXPR   EQU #0F23C
=FIBADR EQU #11457
=FILDC* EQU #05759
=FILXQ^ EQU #09B76
=FINDF+ EQU #09F63
=FLTDH  EQU #1B223
=FSPECe EQU #02F02
=FSPECp EQU #03CC5
=GETCH# EQU #11427
=MFERR  EQU #09393
=#CK    EQU #03356
=NTOKEN EQU #0493B
=NUMC++ EQU #03690
=NUMCK  EQU #0369D
=NXTSTM EQU #08A48
=OUTBYT EQU #02CE8
=OUT1T+ EQU #02CDF
=OUT1TK EQU #02CEB
=POP1S  EQU #0BD38
=RAMROM EQU #0A5F7
=RESPTR EQU #03172
=STRNGP EQU #0379D
=SYNTXe EQU #02E2B

* Numeros d'erreurs
=eEOFIL EQU #36       End of File
=eFACCS EQU #3C       Illegal Access
=eFSPEC EQU #3A       Invalid File Spec
=eFTYPE EQU #3F       Invalid File Type
=eFnFND EQU #39       File Not Found

* Adresses systeme
=STMTD1 EQU #2F896    statement scratch
=STMTR0 EQU #2F871    statement scratch
=STMTR1 EQU #2F881    statement scratch

* Symbole local
sMAP$  EQU 3

* Poll handler pour VER$
 POLLH  ?B=0   B        poll VER$ ?
        GOYES  VER$     oui
        RTNSXM          ce n'est pas VER$, retour
 VER$   C=R3
        D1=C
        D1=D1- (Ve)-(Vs)-2
        CD1EX
        A=R2            AVMEMS
        ?C<A   A
        GOYES  rtnsxm   plus de memoire
```

```
        D1=C
        R3=C
 Vs     LCASC  ' MAP:B'
 Ve     DAT1=C (Ve)-(Vs)-2
 rtnsxm RTNSXM
***********************************************
* decompile de MAP
 dMAP   LCASC  '#'
        ?A=C   B        #<canal>?
        GOYES  dropdc   oui, DROPDC ok
        GOSBVL =FILDC*  decompile fichier
 dropdc GOVLNG =DROPDC  decompile liste
***********************************************
* parse de MAP
 pMAP   GOSBVL =#CK     #?
        GOC    pfsp     non, est-ce un fichier ?
        GOSBVL =OUT1T+  envoie le token de #
        GOSUB  numck    canal <-> expr. numerique
        GONC   pstr1
 pfsp   GOSBVL =FSPECp  parse filespec
        GOC    badf     nom invalide
        GOSBVL =NTOKEN  lit le token suivant
 pstr1  GOSBVL =COMCK+  ',' ? -> envoie le token
        GONC   syntx    non, erreur de syntaxe
        GOSBVL =STRNGP  <str1>
* les virgules suivantes ne seront pas
* tokenisees, si bien qu'un appel a EXPEXC va
* evaluer tous les parametres. DROPDC ajoute une
* virgule.
        GOSUB  comck    virgule ?
        GONC   syntx    non, erreur de syntaxe
        GOSBVL =STRNGP  <str2>
        GOSUB  comck    virgule ?
        GONC   resptr   non, fini
        GOSUB  numck    <num1>
        GOSUB  resptr
        GOSUB  comck    une autre virgule ?
        GONC   resptr
        GOSUB  numck    <num2>
 resptr GOVLNG =RESPTR  fin d'analyse
* points d'entree appeles plus de 3 fois et mis en
* sous-programme pour economiser la memoire
 comck  GOVLNG =COMCK
 numck  GOVLNG =NUMCK
* erreurs d'analyse
 badf   GOVLNG =FSPECe  "Invalid Filespec"
 syntx  GOVLNG =SYNTXe  "Syntax"
***********************************************
* Fonction MAP$
* Syntaxe :
* MAP$(chaine0,chaine1,chaine2)
        NIBHEX 44433    3 parametres chaines
 MAP$   ST=1    sMAP$    indicateur MAP$
        CD0EX           sauvegarde de D0 ..
        R0=C            .. dans R0
        GOTO   bldtbl   creer la table de conv.
***********************************************
* execution de MAP
        REL(5) dMAP     decompile
        REL(5) pMAP     parse
 MAP    A=DAT0 B
        LCASC  '#'
        ?A#C   B
        GOYES  xfsp     ce n'est pas un canal
***********************************************
* c'est un canal, trouver l'adresse du fichier
* son type a partir du FIB
        D0=D0+ 2        passe le token de #
        GOSBVL =GETCH# A(B) := canal
        GOSBVL =FIBADR trouve le FIB
* locating FIB entry before EXPEXC below means
* trouble with UDFs monkeying with channel #,
* but anybody doing that deserves what he gets
        D0=D0+ 2        passe la virgule
        GOSBVL =EXPEX-  met les autres arguments
          *             sur la math-stack
        CD1EX
        D0=C
        D1=(5) =STMTD1  recupere l'adresse du FIB
        C=DAT1 A
        D1=C
        D1=D1+ 5
        C=0    A
        C=DAT1 4        type de fichier
        D1=D1+ 4
        A=DAT1 S        protection
        D1=D1+ 3
        C=DAT1 S        code de l'appareil
        D=C    S
        D1=D1+ 9
        A=DAT1 A        adresse des donnees
        D1=A
        D1=D1- 5
        GONC   chktyp
***********************************************
* nous avons un nom
* lecture du type, etc a partir de l'en-tete
 xfsp   GOSBVL =FILXQ^ execute le nom
        GOC    fspOK    nom de fichier valide
        LC(2)  =eFSPEC "Invalid Filespec"
        GONC   mferr    B.E.T.
* Le resultat de FILXQ^ (nom de fichier et port)
* est stocke dans Statement Scratch pendant
* EXPEX-. On ne peut appeler FINDF d'abord
* puisque EXPEX- peut changer l'adresse du fichier.
 fspOK  D1=(5) =STMTR0  sauve A et D
        DAT1=A W        nom de fichier
        D1=(2) =STMTR1
        C=D    W
        DAT1=C W        information sur le port
        D0=D0+ 2        passe la virgule
        GOSBVL =EXPEX-  chaines sur mathstack
        CD1EX           FINDF+ utilise D1 seul
        D0=C            D1 est sauve dans D0
        D1=(5) =STMTR0  lit le resultat de FILXQ^
        A=DAT1 W
```

```
        D1=(2) =STMTR1
        C=DAT1 W
        D=C    W
        GOSBVL =FINDF+  trouve le fichier
        GOC    mferr    non trouve, erreur
        D1=D1+ 16
        C=0    A
        C=DAT1 4        type de fichier
        D1=D1+ 4
        A=DAT1 S        protection
        D1=D1+ 12
************************************************
* le fichier est trouve, nous avons :
* C(A) = type de fichier
* A(S) = quartet de protection
* D(S) = code de l'appareil
* D1 = ^ link-field
* D0 = pointeur de math-stack
 chktyp C=C-1  A
        ?C=0   A        fichier Text ?
        GOYES  textf    oui
        LC(2)  =eFTYPE  "Invalid File Type"
 mferr  GOVLNG =MFERR
 textf  GOSBVL =RAMROM  en RAM ?
        LC(2)  =eFACCS  "Illegal Access"
        GONC   mferr    non erreur
        GOSBVL =?PRFI+
        GOC    mferr    fichier protege
* nous avons un fichier en Text, en Ram,
* non protege
        CDOEX           pointeur de math-stack
        CD1EX           dans D1
        R0=C            R0 := ^ link-field
        ST=0   sMAP$    nous sommes dans MAP
* on met sMAP$ a 0 seulement maintenant car
* EXPEX- peut changer les flags
************************************************
* Avons-nous des limites d'enregistrement
* y-a-t-il des nombres sur la math-satck
        C=0    A        enreg de debut = 0
        R1=C
        C=C-1  A        nombre d'enreg infini
        R2=C            en fait 2^20-1
        D=0    S        premier passage
* lecture des nombres  (2 max) sur la pile
 num?   LCHEX  9        P=0
        A=DAT1 W        lit le sommet de pile
        ?A>C   P        nombre reel ?
        GOYES  bldtbl   non, voyons les chaines
        D1=D1+ 16       passe le nombre
        GOSBVL =FLTDH   conversion en hexa
        GONC   argerr
        AR1EX           devient l'enreg de debut
        D=D-1  S
        GOC    num?     il y a un autre nombre
        C=R1            devient l'enreg. de fin
        A=A-C  A        <fin>-<debut>
        GONC   b<=e     debut > fin ?
        A=0    A        oui, converti un enreg
 b<=e   R2=A            nombre d'enreg. - 1
************************************************
* MAP et MAP$ se rejoignent ici
 bldtbl GOSBVL =POP1S   pop chaine2
        CD1EX
        D0=C            D0 := ^ chaine2
        B=A    A        B[A] := longueur chaine2
        A=A+C  A
        D1=A            D1 apres la chaine
        GOSBVL =POP1S   pop chaine1
        ?A=B   A        longueurs egales ?
        GOYES  strsOK   oui
 argerr GOVLNG =ARGERR  non, erreur
 strsOK ACEX   A        A[A] := ^ chaine2
        D=0    M
        D=C    A        longueur en octets
        DSRB            longueur en nibs
        CD1EX
        B=C    A        B[A] := ^ chaine1
* la table de conversion est logee au dessus de
* AVMEMS. Verifions d'abord s'il y a assez de
* place pour la loger.
        LC(5)  512      nb de quartets necessaires
        GOSBVL =CHKmem
        GOC    mfer1    pas assez de place
        D1=A            debut de la memoire libre
* On initialise la table en faisant correspondre
* chaque caractere avec lui-meme.
        C=0    B        debut CHR$(0)
 init   DAT1=C B        ecrit le caractere
        D1=D1+ 2        ^ prochain caractere
        C=C+1  B        prochain caractere
        GONC   init
* Puis, nous changeons la table de telle facon
* que si le Ieme caractere de chaine1 est N, alors
* le Neme caractere de chaine2 vaut I.
        ABEX   A        A[A] := ^ debut table
        AD1EX           D1 := ^ chaine1
* A ce point :
* D1 := ^ chaine1
* D0 := ^ chaine2
* B[A] := adresse de la table
* D[A] := longueur de chaine1 (chaine2)
        D=D-1  A        ajuste le compteur
        GOC    nop      si chaine1 est nulle
 setup  A=0    A
        A=DAT1 B        caractere de chaine1
        A=A+A  X        deplacement en quartets
        A=A+B  A        ajout adresse debut table
        AD1EX
        C=DAT0 B        caractere de chaine2
        DAT1=C B        stockage dans la table
        AD1EX           D1 := ^ chaine1
        D1=D1+ 2        ^ caractere suivant ..
        D0=D0+ 2        .. dans les deux chaines
        D=D-1  A        decremente compteur
        GONC   setup
```

```
* la table est constituee, son adresse est
* dans B[A]
 nop    ?ST=0  sMAP$    execute-t-on MAP ?
        GOYES  stmt     oui
************************************************
* Prepare la conversion de str0 sur la Math-Stack
        CD1EX          ^ chaine0
        D1=C           D1 := ^ chaine0
        RSTK=C         sauve sur pile de retour
        A=0    M       pour ASRB plus loin
        D1=D1+ 2       passe le type (0F ou 5F)
        A=DAT1 A       longueur de chaine0
        ASRB           convertie en octets
        D1=D1+ 14      passe reste de l'en-tete
        GOTO   iconv   suite de l'operation
* Fichier endommage
 Eof   LC(2)  =eEOFIL
 mfer1 GOTO   mferr
************************************************
* Prepare la conversion de fichier
 stmt   A=R0           restaure addresse fichier
        D1=A
        D1=D1+ 16      ^ link-field
        A=DAT1 A       adresse relative de fin
        D1=D1+ 5       ^ debut des donnees
* Nous avons besoin de l'adresse de fin du
* fichier pour verifier si aucune longueur de
* ligne ne pointe au-dela.
        CD1EX
        D1=C
        C=A+C  A       ^ fin du fichier
        D=C    A       sauve dans D[A]
        D=0    S       phase initiale
* Boucle de conversion pour un fichier
* D1 = ^ longueur du prochain enregistrement
* B[A] = adresse de la table de conversion
* D[A] = adresse de fin de fichier
* D[S] = 0 on passe des enregistrements au debut
* R1[A] = nombre d'enregistrements a passer/traiter
* R2[A] = nb maximum d'enregistrements a traiter
* P = 0
 nxtrec C=R1           enregistrements restant
        C=C-1  A
        GONC   cont
        ?D=0   S
        GOYES  action  on a passe les
          *            enregistrements souhaites
 nxtstm GOVLNG =NXTSTM termine
 action D=D+1  S
        C=R2
 cont   R1=C
* lecture de l'en-tete de l'enregistrement
        A=0    XS
        A=DAT1 B       1er octet de longueur
        D1=D1+ 2
        ASL    A       decalage a gauche
        ASL    A
        A=DAT1 B       lecture du 2eme octet
        D1=D1+ 2
        P=     3
        A=A+1  WP
        P=     0
        GOC    nxtstm  fin de fichier atteinte
        LCHEX  E
        A=A&C  P       octet de poids faible := 0
        CD1EX          verifie fichier OK
        D1=C
        C=C+A  A
        C=C+A  A       ajoute 2 * longueur
        ?C>D   A       apres fin de fichier
        GOYES  Eof     oui, erreur
        ?D#0   S       phase initiale ?
        GOYES  iconv   non, conversion
        D1=C
        GONC   nxtrec
************************************************
* Ici, MAP et MAP$ se rejoignent a nouveau.
* D1 ^ premier car. de l'enreg. ou de la chaine
* A[A] = nb de caracteres a convertir
* B[A] = adresse de la table de conversion
 iconv  A=A-1  A        ajuste le compteur
        GOC    nullst  sortie si chaine nulle
* boucle interne
 conv   C=0    A
        C=DAT1 B        lit un caractere
        C=C+C  X        double -> offset en nibs.
        C=C+B  A        ajout adresse debut table
        D0=C
        C=DAT0 B        lit le caractere
          *             correspondant dans la
          *             table
        DAT1=C B        caractere remplace
        D1=D1+ 2        caractere suivant
        A=A-1  A
        GONC   conv     pas termine
 nullst ?ST=0  sMAP$    execution de MAP ?
        GOYES  nxtrec   enregistrement suivant
************************************************
* non, fin de la fonction MAP$
        C=RSTK          restaure D1
        D1=C
        C=R0            restaure D0
        D0=C
        GOVLNG =EXPR    retour a Basic

        END
```

GENEALOGIE ASSISTEE PAR ORDINATEUR

Le programme que je vous propose a pour objet de vous aider à y voir plus clair dans votre généalogie.

Le programme comprend deux parties. La première (GENEALOGIE) crée les fichiers nécessaires au fonctionnement du programme, à savoir deux fichiers de données et un fichier d'assignations de touches (Keys). La seconde partie (GAO) est le programme proprement dit.

MODE D'EMPLOI

Chargement

Le programme étant stocké sur cassette ou disquette, faire :

```
CHAIN GENEALOGIE:TAPE
```

Cette opération charge la première partie du programme, l'exécute puis charge la seconde partie et la démarre.

Le HP-71 affiche alors : `MENU PRINCIPAL ^v`. Les indicateurs USER, 0 et PRGM sont allumés.

Lors des utilisations ultérieures, le programme démarre par `[RUN]`.

En cas d'altération des fichiers MAN ou FIC, il suffit de détruire ces fichiers et d'exécuter à nouveau la première partie.

L'exécution de la première partie crée 14 nouveaux caractères (minuscules accentuées), qui sont affectés à 14 touches accessibles par `[f]` en mode User.

Fonctionnement des menus

Chaque menu possède une série d'options. Leur défilement est assuré par les touches `[^]` et `[v]`. Un indicateur en fin de ligne précise le sens de défilement. L'option choisie est appelée par une pression sur `[ENDLINE]`. Lorsque l'option est exécutée, le programme revient au menu de départ. A tout moment, il est possible d'afficher l'heure (touche `[f][BEEP]`) ou la date (touche `[f][ADD]`).

Chaque menu débute par un titre (`[ENDLINE]` est alors sans effet), et se termine par une option de fin :

`FIN` pour le menu principal, soit l'arrêt du programme, ou

`QUITTER` pour les autres menus, soit le retour au menu initial.

Le menu principal

Les différentes options du menu principal sont :

`FICHIER` : gestion d'un fichier (70 fiches au maximum), chaque fiche comprenant 12 rubriques.

`NOTES` : bloc-note, permet l'écriture, la lecture et l'édition de notes (100 lignes de 20 caractères).

`DATEUR` : calcul du jour de la semaine, du nombre de jours entre deux dates et conversion d'une date républicaine en date grégorienne.

`FIN` : arrêt du programme.

L'option « fichier »

L'option fichier gère le fichier FIC. Ses caractéristiques sont :

- 840 enregistrements
- 100 fiches
- 12 rubriques par fiche
- 15 caractères par rubrique

Les fiches sont donc organisées en rubriques, dont voici la liste :

- Nom
- Prénom
- Date de naissance
- Lieu de naissance
- Date de décès
- Lieu de décès
- Nom du père
- Prénom du père
- Nom de la mère
- Prénom de la mère
- Nom du conjoint
- Prénom du conjoint

Les options du menu de gestion de fichier sont :

`AJOUTER` : ajoute une fiche. Les rubriques sont validées par `[ENDLINE]`.

`MODIFIER` : modifie une fiche. Il faut entrer le numéro de la fiche à modifier, puis entrer à nouveau toute la fiche.

EFFACER : efface la totalité du fichier FIC. Par mesure de sécurité, il faut confirmer (touche [O]) l'opération.

SAUVER : copie le fichier FIC sur mémoire de masse. Cette option sauve sur la première mémoire de masse sur la boucle (:TAPE(1)).

CHARGER : charge le fichier FIC à partir de la première mémoire de masse rencontrée sur la boucle.

AIDE : affiche un texte d'aide. On fait défiler le texte par des pressions sur les touches [^] ou [v] (indicateur en fin de ligne). Le retour au menu de gestion de fichier se fait par [ENDLINE].

EDITER : propose un sous-menu pour l'édition des fiches.

Le sous-menu d'édition :

Les options disponibles sont :

UNE FICHE : édite une seule fiche, soit 12 rubriques.

LE FICHIER : édite toutes les fiches.

PAR CRITERES : édite les fiches possédant un ou deux critères. Un critère est une rubrique (12 choix possibles). La recherche se fait suivant un critère, deux critères liés par la relation *et*, ou encore deux critères liés par la relation *ou*.
Par exemple, pour rechercher tous ceux qui sont nés à Paris (un seul critère), ou ceux qui sont nés à Lille et ayant pour prénom Louis (deux critères liés par *et*).
Après le choix du nombre de critères (1 ou 2), le choix des critères se fait en parcourant une liste et en validant le choix par [ENDLINE]. Il faut ensuite entrer la valeur du critère, et le choix de la liaison (entrer ET ou OU).

AVEC FAMILLE : édite une fiche et les fiches des parents (père, mère, conjoint ou enfants). Entrer un nom et un prénom. Si la fiche n'est pas dans FIC, le programme répond INCONNU.

EN ARBRE : édite une fiche et le parent (père ou mère) de l'individu, puis le parent du parent et ainsi de suite jusqu'au dernier choix du parent (entrer P pour père, M pour mère). Les nom et prénom initiaux sont introduits comme dans l'option AVEC FAMILLE.

L'option « notes »

Ce menu permet d'exploiter les 100 premiers enregistrements du fichier MAN. Les options disponibles sont :

LIRE : lecture des notes déjà écrites. Le défilement se fait à l'aide des touches [^] et [v] (indicateur de sens). [ENDLINE] sert à quitter cette phase de lecture.

ECRIRE : écriture d'une note. La note est introduite ligne par ligne, en validant par [ENDLINE]. Pour terminer la note, il faut simplement entrer F.

IMPRIMER : édition de toutes les lignes écrites.

DETRUIRE : effacement de toutes les notes. Par mesure de sécurité, il faut confirmer (touche [O]) l'opération.

AIDE : affiche un texte d'aide. On fait défiler le texte par des pressions sur les touches [^] ou [v] (indicateur en fin de ligne). Le retour au menu de gestion de fichier se fait par [ENDLINE].

L'option « dateur »

Ce menu propose une série de calcul de dates. Les options disponibles sont :

JOUR DE LA SEMAINE : calcule le jour de la semaine correspondant à une date quelconque postérieure à 1582.

Nb DE JOURS ENTRE 2 DATES : affiche la première date (avec jour), la seconde date (avec jour), et le nombre de jours entre ces deux dates.

CALENDRIER : conversion d'une date républicaine (du 1er Vendémiaire an 1 au 10 Nivose an 14) en date grégorienne.
Il faut entrer le jour, puis le mois en toutes lettres, et l'an. Pour les jours supplémentaires, il faut entrer le jour. A la question "mois ?", il faut entrer JOUR. Puis il faut entrer l'an, et le programme affiche la date (avec jour).

AIDE : affiche un texte d'aide. On fait défiler le texte par des pressions sur les touches [^] ou [v] (indicateur en fin de ligne). Le retour au menu de gestion de fichier se fait par [ENDLINE].

REMARQUE IMPORTANTE

En cas d'arrêt du programme (erreur ou [ATTN]), il est fortement conseillé de relancer le programme par [CONT] et d'en sortir avec les options QUITTER et FIN. Ceci respecte les procédures d'ouverture et de fermeture des fichiers et évite les blocages ultérieurs. Autant que possible, *ne pas* relancer le programme par [RUN].

ANNEXE

Structure du fichier MAN :

L'enregistrement 0 est la taille du bloc-notes, soit le nombre de lignes entrées.

Les enregistrements 1 à 100 constituent le bloc-note, c'est à dire les lignes de texte.

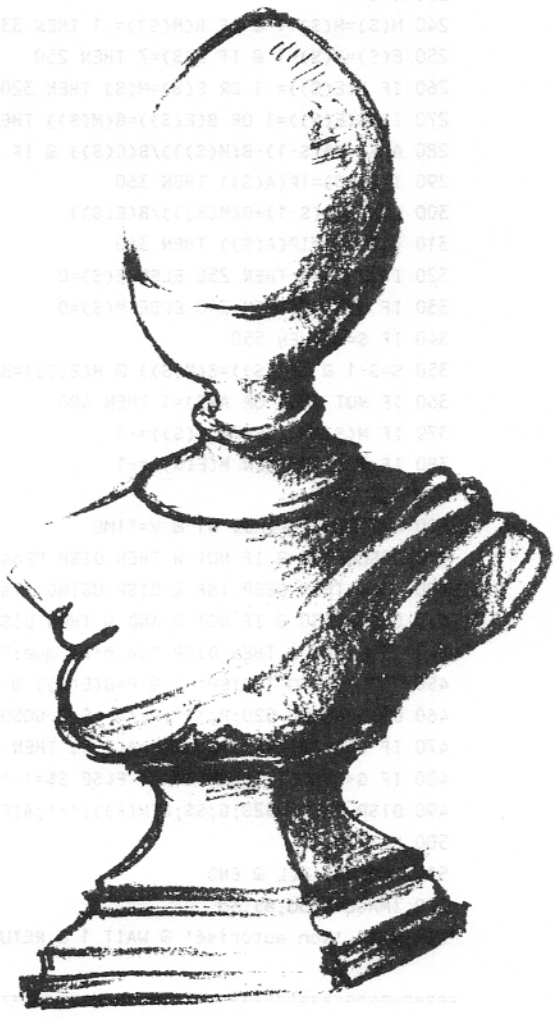
Les enregistrements 101 à 221 contiennent les menus et textes d'aide. Voir la numérotation et l'affectation de ces enregistrements sur la liste du programme GENEALOG.

Structure du fichier Keys :

```
DEF KEY 'fP','î';
DEF KEY 'f7','é';
DEF KEY 'f8','è';
DEF KEY 'f9','ê';
DEF KEY 'f/','ë';
DEF KEY 'f=','ï';
DEF KEY 'f4','æ';
DEF KEY 'f5','ù';
DEF KEY 'f6','û';
DEF KEY 'f*','ü';
DEF KEY 'f1','ç';
DEF KEY 'f2','à';
DEF KEY 'f3','â';
DEF KEY 'f-','ô';
```

Bonnes recherches généalogiques...

Philippe Nicodème (237)



Programme "LCEB" (Le compte est bon, voir rubrique HP-41)

```
 10 OPTION BASE 0 @ DESTROY ALL @ INPUT "T. de réflexion:","60";T @ DIM S$[1],K$[4]
 20 FOR I=1 TO 6
 30 RESTORE
 40 DISP "Plaque";I;":";
 50 ON ERROR GOTO 30
 60 INPUT B(I) @ Y=0
 70 FOR J=1 TO 14
 80 READ Q @ IF Q=B(I) THEN Y=1
 90 NEXT J
100 IF Y=1 THEN 110 ELSE GOSUB 530 @ GOTO 30
110 GOSUB 590
120 IF F=3 THEN GOSUB 530 @ GOTO 30
130 NEXT I
140 OFF ERROR
150 ON ERROR GOTO 180
160 INPUT "Nb de 3 chif. maxi.:";A(0) @ DISP @ Z(1)=A(0) @ OFF ERROR
170 IF IP(LOG(A(0)))<=2 THEN 190
180 GOSUB 530 @ GOTO 150
190 ON TIMER #1,T GOTO 410
200 X=TIME @ O=0 @ W=0
210 S=0 @ B(8)=0 @ M(0)=0 @ E(0)=0 @ B(7)=0
220 FOR D=1 TO 6 @ H(D)=B(D) @ NEXT D
230 S=S+1
240 M(S)=M(S)+1 @ IF H(M(S))=-1 THEN 330
250 E(S)=E(S)+1 @ IF E(S)=7 THEN 250
260 IF H(E(S))=-1 OR E(S)=M(S) THEN 320
270 IF B(E(S))=1 OR B(E(S))=B(M(S)) THEN 320
280 A(S)=(A(S-1)-B(M(S)))/B(E(S)) @ IF A(S)<0 THEN 300
290 IF A(S)=IP(A(S)) THEN 360
300 A(S)=(A(S-1)+B(M(S)))/B(E(S))
310 IF A(S)=IP(A(S)) THEN 360
320 IF E(S)<6 THEN 250 ELSE E(S)=0
330 IF M(S)<7 THEN 240 ELSE M(S)=0
340 IF S=1 THEN 550
350 S=S-1 @ H(M(S))=B(M(S)) @ H(E(S))=B(E(S)) @ GOTO 320
360 IF NOT A(S) OR A(S)=1 THEN 400
370 IF M(S)#7 THEN H(M(S))=-1
380 IF E(S)#8 THEN H(E(S))=-1
390 GOTO 230
400 W=1 @ OFF TIMER #1 @ V=TIME
410 GOSUB 580 @ IF NOT W THEN DISP "Pas de solution" @ GOTO 510
420 IF W THEN BEEP INF @ DISP USING "4a,x,3d,x,3a,x5d.d.a";"J'ai",A(0),"en :",U-X,'""'
430 GOSUB 630 @ IF NOT O AND W THEN DISP "L.C.E.B"
440 IF W AND O THEN DISP "Je n'ai que:";A(0)
450 FOR F=5 TO 1 STEP -1 @ P=B(E(F)) @ G=R*P @ IF R=G OR P=G THEN 470
460 DISP USING 520;R;'*';P;'=';G @ GOSUB 630
470 IF G=A(F-1) OR B(M(F))=A(F-1) THEN 500
480 IF G<A(F-1) THEN S$='+' ELSE S$='-'
490 DISP USING 520;G;S$;B(M(F));'=';A(F-1) @ GOSUB 630
500 NEXT F
510 DESTROY ALL @ END
520 IMAGE 2(3d,a),3d
530 DISP 'Non autorisé' @ WAIT 1 @ RETURN
```

==========

```
 540 DATA 1,2,3,4,5,6,7,8,9,10,25,50,75,100

============================================================
 550 O=1 @ A(0)=A(0)-1
 560 IF A(0)>100 THEN 210
 570 DISP "Je n'ai pas trouvé" @ BEEP @ GOTO 510
 580 OFF TIMER #1 @ RETURN
 590 X=0 @ FOR K=1 TO I
 600 IF B(K)=B(I) THEN X=X+1
 610 NEXT K @ IF X>=3 THEN F=3 @ RETURN
 620 F=1 @ RETURN
 630 K$=KEY$ @ IF K$='' THEN 630
 640 RETURN
```

****

Programme "PCT9" (Utilitaire VisiCalc, voir rubrique HP-75)

```
   - PCT9(élément,case total), %T, sans arrondi.
  20 OPTION BASE 1
  30 ASSIGN # 255 TO "VISIDATA"
  40 READ # 255,0 ; N1,N2 @ IF N1#2 OR N2#2 THEN END
  50 READ # 255 ; P,T
  60 P=P*100/T
  70 PRINT # 255,0 ; P
```

****

Programme "DPCT" (Utilitaire VisiCalc, voir rubrique HP-75)

```
   - DPCT(n1,n2), delta% sans arrondi
  20 OPTION BASE 1
  30 ASSIGN # 255 TO "VISIDATA"
  40 READ # 255,0 ; N1,N2 @ IF N1#2 OR N2#2 THEN END
  50 READ # 255 ; N1,N2
  60 D=100*((N2-N1)/N1)
  70 PRINT # 255,0 ; D
```

****

Programme "VCCLFORM" (Utilitaire VisiCalc, voir rubrique HP-75)

```
   - VCCLFORM (c)Eric GENGOUX
  20 DISP "Effacement formules VisiCALC"
  30 OPTION BASE 1
  40 DIM W$[8]
  50 W$=ACTIVE$
  60 INPUT "Feuille traitee ",W$;W$
  70 IF INCAT(W$,'W')#2 THEN BEEP @ GOTO 60
```

```
 80 WORKSHEET W$
 90 W2$=COORD$(1,1,1)
100 INPUT "1ere case ",W2$;W2$
110 IF COL(W2$)=0 OR ROW(W2$)=0 THEN BEEP @ GOTO 100
120 R0=ROW(W2$) @ C0=COL(W2$)
130 W2$=COORD$(MAXROW,MAXCOL,1)
140 INPUT "Dern.case ",W2$;W2$
150 IF COL(W2$)=0 OR ROW(W2$)=0 THEN BEEP @ GOTO 100
160 R9=ROW(W2$) @ C9=COL(W2$)
170 DISP "Un instant..."
180 FOR R=R0 TO R9
190 FOR C=C0 TO C9
200 IF GETLABEL$(C,R)#"" THEN GOTO 240
210 PUTFORMULA C,R,GETVALUE$(C,R,0)
220 NEXT C
230 NEXT R
240 BEEP 1000,.5 @ BEEP 700,.5 @ DISP "Termine" @ END
```

****

Programme "VCLISTF" (Utilitaire VisiCalc, voir rubrique HP-75)

```
   - VCLISTF liste formules avec en-têtes par défaut
     (c)Eric GENGOUX 29\10\1986
  12 PRINT CHR$(27)&'&k2S'
  20 DIM F$[120],W$[8]
  30 INPUT "WS name? "; W$ @ W$=UPRC$(W$)
  40 IF INCAT(W$,"W")#2 THEN BEEP @ GOTO 30
  50 WORKSHEET W$
  60 PRINT "Listage formules feuille ";W$ @ PRINT
  65 PRINT "LABELS UTILISATEUR:" @ PRINT
  70 FOR C=1 TO MAXCOL
  80 F$="COL. "&COORD$(C,0,0)&" = "&GETLABEL$(C,0) @ PRINT F$
  90 NEXT C
  95 PRINT @ PRINT
 100 FOR L=1 TO MAXROW
 110 F$="ROW "&STR$(L)&" = "&GETLABEL$(0,L) @ PRINT F$
 120 NEXT L
 130 PRINT @ PRINT "FORMULES (par colonnes)" @ PRINT
 140 FOR C=1 TO MAXCOL
 150 FOR L=1 TO MAXROW
 160 F$=COORD$(C,L,0)&' = '&GETFORMULA$(C,L,0)
 161 IF LEN(F$)=POS(F$,'=')+1 THEN GOTO 170
 165 PRINT F$
 170 NEXT L
 180 PRINT
 190 NEXT C
 200 DISP "**Fin**" @ BEEP 600,.5 @ BEEP 400,.5 @ END
```

****

Programme "GENEALOGIE" (Genealogie assistee par ordinateur, premiere partie)

```
 ****************************************
 *                                      *
 *   GENEALOGIE                         *
 *                                      *
 *   NICODEME  philippe                 *
 *                                      *
 *   18/01/1986                         *
 *                                      *
 ****************************************


 ****************************************
 *            1ère partie               *
 * création des fichiers MAN et FIC     *
 * création des caractères accentués    *
 * chargement du programme principal    *
 ****************************************


 ************************
 * contenu du fichier   *
 *        M A N         *
 ************************


 ************************
 * MENU PRINCIPAL       *
 ************************
 * 101 a 105            *
 ************************

================================================================
  43 DATA Menu principal
  44 DATA FICHIER
  45 DATA NOTES
  46 DATA DATEUR
  47 DATA FIN

================================================================


 ************************
 * MENU DATEUR          *
 ************************
 * 106 a 111            *
 ************************

================================================================
  55 DATA Menu dateur
  56 DATA JOUR DE LA SEMAINE
  57 DATA Nb DE JOURS
  58 DATA CALENDRIER
  59 DATA AIDE
  60 DATA quitter DATEUR
```

```
============================================================


    ***********************
    * AIDE OPTION DATEUR  *
    ***********************
    *  112 a 131          *
    ***********************

============================================================
  68 DATA DATEUR : 3 options
  69 DATA 1/JOUR DE LA SEMAINE
  70 DATA calcul du jour
  71 DATA "d'après une date"
  72 DATA 2/Nb DE JOURS
  73 DATA calcul du nombre de
  74 DATA jours entre 2 dates
  75 DATA 3/CALENDRIER
  76 DATA donne une date
  77 DATA "grégorienne d'après"
  78 DATA une date républicaine
  79 DATA PROCEDURE :
  80 DATA entrer le No du jour
  81 DATA entrer le mois
  82 DATA en toutes lettres
  83 DATA jours complémentaires
  84 DATA entrer le No du jour
  85 DATA (1 à 6)
  86 DATA 'à la question "mois?"'
  87 DATA entrer JOUR

============================================================


    ***********************
    * MENU BLOC-NOTES     *
    ***********************
    * 132 a 138           *
    ***********************

============================================================
 100 DATA Menu BLOC-NOTES
 102 DATA LIRE
 104 DATA ECRIRE
 106 DATA IMPRIMER
 108 DATA DETRUIRE
 110 DATA AIDE
 112 DATA quitter NOTES

============================================================


    ***********************
    * AIDE BLOC-NOTES     *
    ***********************
    * 139 a 154           *
    ***********************
```

```
================================================================
 128 DATA BLOC-NOTES
 130 DATA lecture et écriture
 132 DATA 100 lignes maxi
 134 DATA 20 caractères / ligne
 136 DATA OPTIONS :
 138 DATA 1/LIRE
 140 DATA lecture par ^ et v
 142 DATA quitter par ENDLINE
 144 DATA 2/ECRIRE
 146 DATA ligne par ligne
 148 DATA pour quitter :
 150 DATA entrer F
 152 DATA 3/IMPRIMER
 154 DATA 4/DETRUIRE
 156 DATA confirmer par O
 158 DATA 5/QUITTER

================================================================


    **********************
    * CALEND REPUBLICAIN *
    **********************
    * 155 a 167          *
    **********************

================================================================
 174 DATA VENDEMIAIRE
 176 DATA BRUMAIRE
 178 DATA FRIMAIRE
 180 DATA NIVOSE
 182 DATA PLUVIOSE
 184 DATA VENTOSE
 186 DATA GERMINAL
 188 DATA FLOREAL
 190 DATA PRAIRIAL
 192 DATA MESSIDOR
 194 DATA THERMIDOR
 196 DATA FRUCTIDOR
 198 DATA JOUR

================================================================
   -

    **********************
    * MENU FICHIER       *
    **********************
    * 167 a 177          *
    **********************

================================================================
 214 DATA Menu fichier
 216 DATA AJOUTER
 218 DATA MODIFIER
 220 DATA EDITER
 222 DATA EFFACER
 224 DATA SAUVER
```

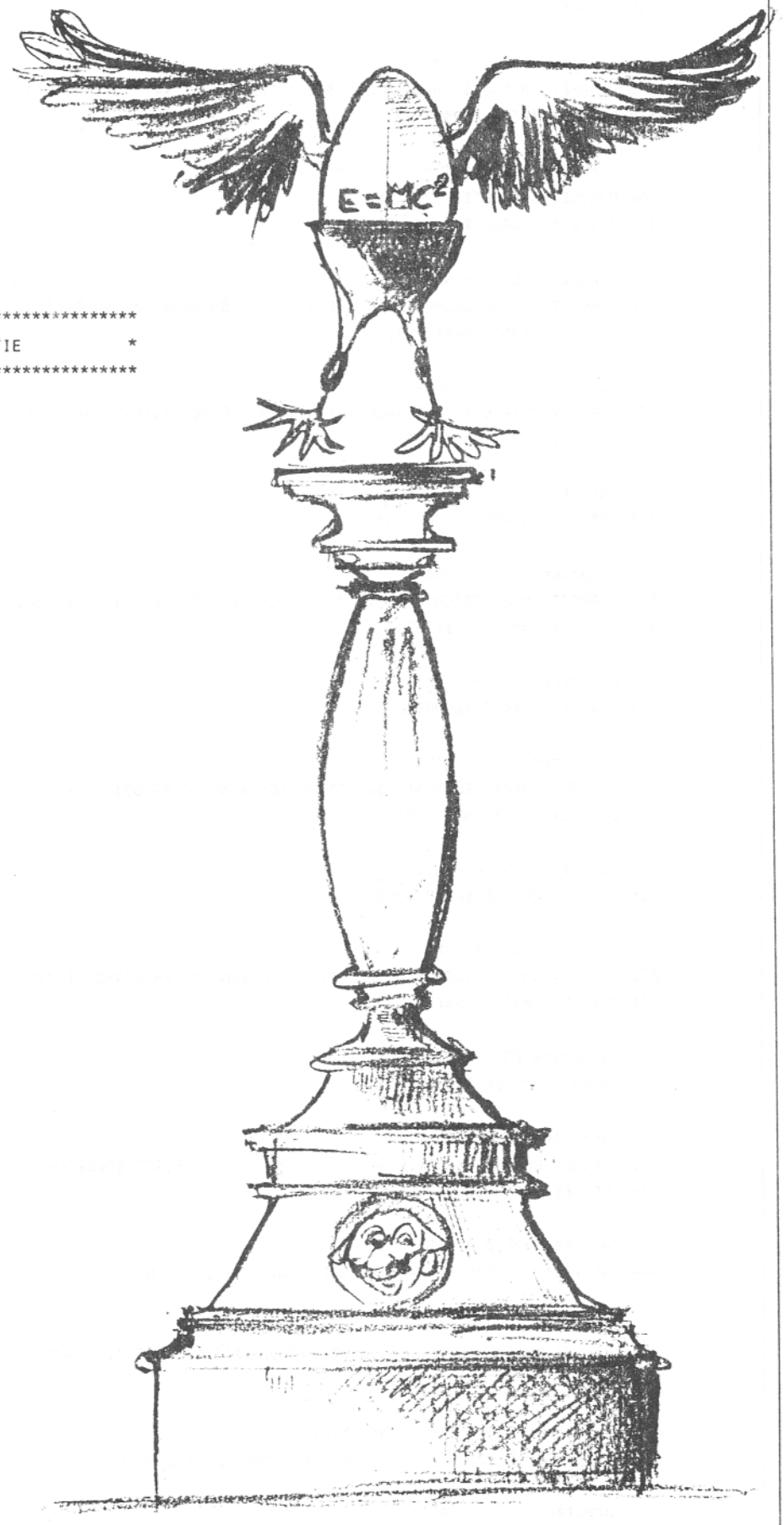
```
226 DATA CHARGER
228 DATA AIDE
230 DATA quitter FICHIER

================================================================

    **********************
    * MENU EDITER        *
    **********************
    * 178 a 184          *
    **********************

================================================================
246 DATA menu EDITER
248 DATA UNE FICHE
250 DATA LE FICHIER
252 DATA PAR CRITERES
254 DATA AVEC FAMILLE
256 DATA EN ARBRE
258 DATA quitter EDITER

================================================================

    **********************
    * CRITERES EDITER    *
    **********************
    * 186 à 199          *
    **********************

================================================================
274 DATA choix du critères
276 DATA choix par END LINE
278 DATA NOM
280 DATA prénom
282 DATA né le
284 DATA né à
286 DATA décés le
288 DATA décés à
290 DATA père (n)
292 DATA père (p)
294 DATA mère (n)
296 DATA mère (p)
298 DATA conj.(n)
300 DATA conj.(p)

================================================================

    **********************
    * AIDE FICHIER       *
    **********************
    * 200 à 220          *
    **********************

================================================================
316 DATA gestion de fichier
```

```
318 DATA 100 fiches
320 DATA 25 caract./fiche
322 DATA OPTIONS :
324 DATA AJOUTER une fiche
326 DATA MODIFIER une fiche
328 DATA EFFACER le fichier
330 DATA SAUVER le fichier
332 DATA CHARGER le fichier
334 DATA EDITER / menu
336 DATA /EDITER une fiche
338 DATA /EDITER le fichier
340 DATA /EDITER une famille
342 DATA individu + père
344 DATA + mère + conjoint
346 DATA + enfants
348 DATA /EDITER par critères
350 DATA 1 ou 2 critères
352 DATA 1 et 2 / 1 ou 2
354 DATA choix : 12 critères
356 DATA /EDITER un arbre
358 DATA parent(père / mère)
360 DATA "de l'individu"
362 DATA parent du parent etc

================================================================


    **********************
    * crée le fichier    *
    *    M --> menu      *
    *    A --> aide      *
    *    N --> notes     *
    **********************

    message d'attente
384 DELAY 0,0 @ DISP "un peu de patience..."

    création et ouverture du fichier
390 CREATE DATA MAN,221,28 @ ASSIGN #1 TO MAN

    les enregistrements 1 à 100 serviront à l'écriture des notes
396 FOR I=1 TO 100 @ RESTORE #1,I @ PRINT #1;" " @ NEXT I

    chargement des menus et des aides
402 FOR I=101 TO 220 @ READ A$ @ RESTORE #1,I @ PRINT #1;A$ @ NEXT I

    l'enregistrement 0 (taille du bloc-notes) est initialisé
408 RESTORE #1,0 @ PRINT #1;1

    fermeture du fichier
414 ASSIGN #1 TO *


    **********************
    * crée le fichier    *
    *      F I C         *
    *     (fiches)       *
    **********************
```

```
    création et ouverture du fichier
438 CREATE DATA FIC,840,18 @ ASSIGN #1 TO FIC

    initialisation des enregistrements
444 FOR I=1 TO 839 @ RESTORE #1,I @ PRINT #1;" " @ NEXT I

    initialisation enregistrement 0 (taille du fichier)
450 RESTORE #1,0 @ PRINT #1;1
  -
    fermeture du fichier
456 ASSIGN #1 TO *


    **********************
    *  création et       *
    *  affectation des   *
    *  caractères        *
    *  accentués         *
    **********************


    caractère v  non affecté
480 CHARSET CHR$(32)&CHR$(96)&CHR$(192)&CHR$(96)&CHR$(32)

    caractère ^  non affecté
486 CHARSET CHARSET$&CHR$(4)&CHR$(6)&CHR$(3)&CHR$(6)&CHR$(4)

    caractère ^ et v
492 CHARSET CHARSET$&CHR$(36)&CHR$(102)&CHR$(195)&CHR$(102)&CHR$(36)

    caractères 133 à 180 : non utilisés
498 FOR I=131 TO 180 @ CHARSET CHARSET$&CHR$(255) @ NEXT I

    caractère ç  --> ASIN
504 CHARSET CHARSET$&CHR$(56)&CHR$(68)&CHR$(196)&CHR$(68)&CHR$(68)
506 DEF KEY '#95',CHR$(181);

    caractères 182 á 191 : non utilisés
512 FOR I=182 TO 191 @ CHARSET CHARSET$&CHR$(255) @ NEXT I

    caractére â  --> ATAN
518 CHARSET CHARSET$&CHR$(32)&CHR$(85)&CHR$(85)&CHR$(85)&CHR$(121)
520 DEF KEY '#97',CHR$(192);

    caractère ê  --> SDEV
526 CHARSET CHARSET$&CHR$(56)&CHR$(85)&CHR$(85)&CHR$(85)&CHR$(24)
528 DEF KEY '#69',CHR$(193);

    caractère ô  --> LOG
534 CHARSET CHARSET$&CHR$(56)&CHR$(69)&CHR$(69)&CHR$(69)&CHR$(56)
536 DEF KEY '#98',CHR$(194);

    caractère û  --> TAN
542 CHARSET CHARSET$&CHR$(60)&CHR$(65)&CHR$(65)&CHR$(65)&CHR$(124)
544 DEF KEY '#83',CHR$(195);
```

```
    caractère 196 non utilisé
550 CHARSET CHARSET$&CHR$(255)

    caractère é  --> PREDV
556 CHARSET CHARSET$&CHR$(56)&CHR$(84)&CHR$(86)&CHR$(85)&CHR$(24)
558 DEF KEY '#67',CHR$(197);

    caractère 198 et 199 : non utilisés
564 CHARSET CHARSET$&CHR$(255)
566 CHARSET CHARSET$&CHR$(255)

    caractère à  --> ACOS
572 CHARSET CHARSET$&CHR$(32)&CHR$(85)&CHR$(86)&CHR$(84)&CHR$(120)
574 DEF KEY '#96',CHR$(200);

    caractère è  --> MEAN
580 CHARSET CHARSET$&CHR$(56)&CHR$(85)&CHR$(86)&CHR$(84)&CHR$(24)
582 DEF KEY '#68',CHR$(201);

    caractère 202 -->  non utilisés
588 CHARSET CHARSET$&CHR$(255)

    caractère ù -->  COS
594 CHARSET CHARSET$&CHR$(60)&CHR$(65)&CHR$(66)&CHR$(64)&CHR$(124)
596 DEF KEY '#82',CHR$(203);

    caractère 204 non utilisé
602 CHARSET CHARSET$&CHR$(255)

    caractère ë -->  SQR
608 CHARSET CHARSET$&CHR$(56)&CHR$(85)&CHR$(84)&CHR$(85)&CHR$(24)
610 DEF KEY '#70',CHR$(205);

    caractère 206 non utilisé
616 CHARSET CHARSET$&CHR$(255)

    caractère ü -->  EXP
622 CHARSET CHARSET$&CHR$(60)&CHR$(65)&CHR$(64)&CHR$(65)&CHR$(124)
624 DEF KEY '#84',CHR$(207);

    caractère 208 -->  non utilisé
630 CHARSET CHARSET$&CHR$(255)

    caractère î -->  LR
636 CHARSET CHARSET$&CHR$(0)&CHR$(0)&CHR$(69)&CHR$(125)&CHR$(65)
638 DEF KEY '#66',CHR$(209);

    caractères 210 à 214 : non utilisès
644 FOR I=210 TO 214 @ CHARSET CHARSET$&CHR$(255) @ NEXT I

    caractère æ  --> SIN
650 CHARSET CHARSET$&CHR$(120)&CHR$(20)&CHR$(124)&CHR$(84)&CHR$(84)
652 DEF KEY '#81',CHR$(215);

    caractères 216 à 220 : non utilisés
658 FOR I=216 TO 220 @ CHARSET CHARSET$&CHR$(255) @ NEXT I

    caractère ï -->  FACT
```

```
664 CHARSET CHARSET$&CHR$(0)&CHR$(0)&CHR$(69)&CHR$(124)&CHR$(65)
666 DEF KEY '#80',CHR$(221);


    **********************
    * chargement du      *
    * programme          *
    * principal          *
    **********************

684 CHAIN GAO:TAPE


    ************************************
    * FIN DE LA PREMIERE PARTIE        *
    ************************************
```



LE COIN DES LHEX

Comme de coutume, cette rubrique contient la liste des codes hexadécimaux des fichiers Lex parus ce mois-ci.

Rappelons ce qu'est un fichier Lex : c'est un programme pour le HP-71, en assembleur, qui apporte de nouvelles fonctions. Celles-ci sont utilisables directement, ou dans des programmes Basic.

Pour bénéficier de ces nouvelles fonctions, vous n'avez pas besoin de programmer vous-même en assembleur, ni de posséder un module Forth/Assembleur.

Il suffit de recopier le petit programme basic "MAKELEX" ci-dessous, de le lancer et de recopier les codes du fichier Lex désiré. Quand vous avez fini, les nouvelles fonctions sont accessibles, après avoir éteint et rallumé votre HP-71.

Si l'erreur "Erreur de somme" apparaît, vérifiez la ligne que vous avez introduite.

Vous trouverz donc le Lex CHARLEX nécessaire à la rédaction de votre article (voir "Ah ! Vous écrivez !") et le Lex MAPLEX de Tapani Tarvaïnen.

```
CHARLEX

MAPLEX    MAP$        XFN 225076    MAP        XWORD 225077
```

```
 10 CALL MLEX @ SUB MLEX @ SFLAG -1 @ PURGE AH @ INPUT "Nb. d'octets: ";N @ LC OFF
 20 CREATE DATA AH,1,N-4 @ A=HTD(ADDR$("AH")) @ B=A @ GOSUB 130
 30 Q=1 @ X=0 @ INPUT "000: ",P$;A$ @ C$=A$ @ S=0 @ GOSUB 90
 40 Q=2 @ X=1 @ GOSUB 80 @ A$=A$&C$ @ A=A+37 @ N=N*2+37 @ Q=3 @ SFLAG 5 @ FOR X=2 TO N DIV 16-1
 50 GOSUB 80 @ C$=C$[5*FLAG(5)+1] @ POKE DTH$(A),C$ @ A=A+16-5*FLAG(5,0) @ NEXT X @ Q=4
 60 DISP DTH$(X)[3]; @ INPUT ": ",P$[1,MOD(N,16)];C$ @ GOSUB 90
 70 POKE DTH$(A),C$ @ POKE DTH$(B),A$ @ CFLAG -1 @ END
 80 DISP DTH$(X)[3]; @ INPUT ": ",P$;C$
 90 DISP DTH$(X)[3]; @ INPUT " sm ","---";D$
100 M=S @ FOR Z=1 TO LEN(C$) @ M=NUM(C$[Z])+M+1 @ NEXT Z
110 IF D$=DTH$(MOD(M,4096))[3] THEN GOSUB 130 @ S=M @ RETURN
120 DISP "Erreur de somme" @ BEEP @ P$=C$ @ POP @ ON Q GOTO 30,40,50,60
130 P$="----------------" @ RETURN
```

```
CHARLEX  ID#E1  624 octets

     0123456789ABCDEF sm

000: 34841425C4548502 35E
001: 802E006031326078 6B1
002: 5E4001E000000000 9F5
003: FE0000000800001F D4F
004: F31BF961400032BF 0E2
005: 38F14A11DB10AD23 47C
006: 07D532BFB8FD7911 82F
007: 11AD754D7A101743 BB2
008: 11014D1CB15D0000 F1D
009: 71450375FF864834 29A
00A: 5655581008355654 5F1
00B: 5810002455565870 93E
00C: 0026555658700836 C90
00D: 5556581008364545 FE6
00E: 4A30000A49724000 339
00F: 0808094A2C180814 6A2
010: A464242008355455 9FC
011: 581000054C714000 D42
012: 0C3142404C700832 09E
013: 41414A70002078A0 3F6
014: 2F30000000000000 721
015: 0000000000000000 A31
016: 0000000000000000 D41
017: 0000000000000000 051
018: 0000000000000000 361
019: 0000000000000000 671
01A: 0000000000000000 981
01B: 0000000000000000 C91
01C: 0000000000000000 FA1
01D: 0000000000000000 2B1
01E: 0000000000000000 5C1
01F: 0000000000000000 8D1
020: 0000000000000000 BE1
021: 000000000000080C F0C
022: 1A28080008080A2C 276
023: 180008040E340800 5BF
024: 08001E3018000000 8F9
025: 0000000000000000 C09
026: 0000000000000000 F19
027: 0000000000000000 229
028: 0201000000010200 53F
029: 0000000201020000 854
02A: 0001000100000002 B68
02B: 0102010000000000 E7C
02C: 0000000000000000 18C
02D: 045E755142400101 4D8
02E: 0101010000000000 7EB
02F: 0000000000000000 AFB
030: 0000070507000000 E1E
031: 000000000083444C4 15C
032: 44400D7901112D70 4BC
033: 050D750509700000 806
034: 0D70000000384540 B49
035: 4020014E322E3140 E9D
036: 084E794142400000 1ED
037: 00000000002E4559 52B
038: 3200000000000000 840
039: 0000000000000026 B58
03A: 5556587008365556 EB7
03B: 5810083645464830 208
03C: 0832414248700024 549
03D: 5655587008345655 8A6
03E: 5810083446454830 BF5
03F: 0C3042414C700024 F4A
040: 5556587008355654 2A7
041: 5810083546444830 5F6
042: 0C3142404C700025 94C
043: 5455587008355455 CA6
044: 5810083544454830 FF4
045: 0C3140414C700875 356
046: 14141870000A4972 6A7
047: 40000E3159454E30 A07
048: 0C7A0F7949400024 D7F
049: 5554587000084A71 0DB
04A: 40000C523A262D10 43C
04B: 0424587458400875 793
04C: 1415187000094A70 AE3
04D: 4000083544454830 E27
04E: 0C3140414C300C74 18F
04F: 5655545000054C71 4E6
050: 40000            5DF

MAPLEX   ID#E1  484 octets

     0123456789ABCDEF sm

000: D41405C454850202 363
001: 802E007031326078 6B7
002: DC3001EC4D400000 A36
003: F02000000E200000 D75
004: 0CF000FB00A0100D 0FC
005: 7D4140542C45D414 474
006: 05D41FF969400011 7EC
007: B1351CB137112886 B6C
008: A113510B3B24A305 EDE
009: 14D40215DB003132 242
00A: 962908F957508D07 5C7
00B: 4508F653304018FF 948
00C: DC207B505318F5CC CF1
00D: 304758FB39408FEA 098
00E: 2305D48FD97307B2 42A
00F: 05F38FD97307D105 7BE
010: 517D107B007E0056 B30
011: 07E008D271308DDC EC8
012: 6308DD96308D20F2 259
013: 08DB2E2044433853 5D2
014: 1361086C4125FFF4 960
015: 6FFF14A313296695 CFA
016: 1618F724118F7541 06E
017: 11618F871F013713 3DA
018: 41F698F214713517 751
019: 4D215F3173153417 ABA
01A: 21574AC717814313 E24
01B: 11C45C68F67B9049 1BE
01C: 031A35661F178F21 537
01D: 5171D18AFB155716 8C4
01E: 18F871F01371341F C43
01F: 178F215371D18157 FB6
020: 7AF78F36F9040217 34F
021: FD215F3173153417 6CA
022: BCE8AAD031F38D39 A97
023: 3908F7F5A031C35D E37
024: E8F0837143E13613 1B7
025: 7108843D2109CE10 52F
026: AAC3309153798652 8B1
027: 17F8F322B1583121 C2B
028: A4F40E119EA540D0 FCD
029: 1028F83DB0137134 342
02A: D8CA1318F83DB08A 6FF
02B: 0908D91FB0DEAD3D ACB
02C: 781F137D53400200 E2E
02D: 8F7C210436131AE2 1B3
02E: 14D171B6656FDC13 54F
02F: 3CF432D014BA34C0 8EA
030: 13314E14D1331711 C45
031: 61CF51E863521371 FC4
032: 3506AD017114381C 334
033: 17D6280316360FE1 6AF
034: 1013117F14317413 9FB
035: 7135C2D7AC3119CE DA5
036: 54194B908D84A80B 13A
037: 4711A109AA014B17 4B3
038: 1F0F014B17123B14 82C
039: 2045D30E0E061371 B9A
03A: 35C2C28B7E894F80 F45
03B: 13557ACC4D1D214F 2EB
03C: A36C913414E14D17 674
03D: 1CC57E8632807135 9F7
03E: 1181348DC32F0    CD0
```

Le Journal JPC est le bulletin de liaison entre les membres de l'Association "PPC-PC", régie par la loi de 1901. Le Club est éditeur du JPC, et son siège social est au 56, rue Jean-Jacques Rouseau, 75001 Paris.

La maquette de ce numéro a été préparée et réalisée par Pierre David, Jean-Jacques Dhénin et Janick Taillandier, grâce à un système comprenant un HP71B, deux lecteurs de disquettes HP9114A, une imprimante HP2225B et une imprimante LaserJet.

Numéro ISSN : 0762 - 381X

COPY-EXPRESS 42.86.91.94

ENGLISH SUMMARY
JPC 46 - JULY / AUGUST 1987

This double issue, corresponding to holidays, allows us to take some rest.

A member letter we publish is one the first reactions we receive about the JPCLEX Eprom. The user's manual was very well received. The english translation is almost completed, and we expect a larger distribution in september.

This issue contains a lot of articles about the HP-28C. The first one gives a few programs to compute mathematical series and differential operators. The second one gives a nice clock mode. HP has exceptionally allowed us to publish it in french in JPC. The last article is a follow-up of last month one about stack manipulations.

Le compte est bon is one of the most popular TV games in France. Its purpose is to find a way to compute a number, given 6 numbers and addition, subtraction, multiplication and integer division operators. The program for HP-41 (and HP-71) by Tony Guilloux solves this puzzle.

The next HP-41 program helps you manage competition results.

Eric Gengoux presents VisiCalc extension functions he is regularly using. They consist of HP-12 type functions applied to a worksheet.

The first article in HP-71 column is the sequel of the assembly language introductory serie. Jacques Baudier explains us the microprocessor architecture, Forth / Assembler module usage, and gives us the first code example.

Next, Tapani Tarvainen presents us its character mapping Lex file. This Lex and article were initially published in *STaK* 5, the Finnish Users Club Journal. The article describes very well the various development stages of this Lex.

The last article for this month is a CAG (Computer Aided Genealogy) program ! The second program will be published in the September issue.

Until next time,

Happy Programming and JPC reading...