JPC

4/19/89

AVRIL 1989

NUMERO 63

Le numéro 40 F

A PROPOS DU CLUB

HP28

HP41

HP71

EDITORIAL

Chers amis

Que du sérieux ce mois ci, rien que du sérieux !

Citons en vrac un compilateur Lisp pour HP-28, un programme de calcul de résistance des matériaux pour HP-41, des fonctions électroniques et graphiques pour HP-71, et bien d'autres...

Certains le savent déjà, le 3 Juin 1989, pendant le *Consumer Electronic Show* à Chicago, aux U.S.A. aura lieu une conférence dans le but de fêter, entre autres, les 10 ans de la HP-41 et le 75ème anniversaire de Hewlett Packard. Il y a fort à parier que si un certain constructeur a des révélations à faire, ce sera ce jour là ! Malheureusement, il y à *très* peu de chances que nous puissions participer à cette fête. Mais si (par hasard!), l'un d'entre vous passait dans ce secteur, nous serions heureux d'en être tenus au courant.
Cela ne nous empêchera pas de fêter, de notre coté, les dix ans de la HP-41!

Nous aimerions aussi vous parler de la (trop rare!) production de de vos articles. Nous voudrions d'abord insister sur la forme dont nous les recevons. Trop souvent, ils nous arrivent avec une présentation telle que nous mettons surement plus de temps à les remettre en page que leur auteur à les taper ! Suivez au moins les conseils de la rubrique "ah, vous ecrivez"... Pour ceux qui nous envoient des articles tapés sur IBM PC, Essayez de nous envoyer des disquettes 3½", que nous pouvons traiter directement sur notre Integral PC. Le respect de ces quelques règles nous ferait gagner *énormément* de temps.
Plus important encore, vos articles pêchent souvent par le manque d'exemples et vos programmes de commentaires. Des exemples parce que vous êtes surement la seule personne à ne pas avoir à apprendre à vous servir de votre programme. Des commentaires, parce que les autres voudrons sûrement savoir comment vous avez fait.

Le proverbe du mois :
"`Memory Lost` n'est pas mémoire perdue!"

Le Bureau

SOS

Amis utilisateurs de HP-41, je fais appel à vos compétences!

Ma profession peu courante, gnomoniste (constructeur de cadrans solaires), m'a amené à me pencher sur l'astronomie, la topographie, la construction, les bétons, dessin en perspective, ainsi qu'évidemment sur la programmation HP-41 et du BASIC.

la HP-41 est, pour cette application, la solution "tout terrain" idéale. Prise d'orientation d'un site au soleil avec time et navigation, calcul des cadrans, des différents temps et calendriers, devis immédiats, etc... Mais la limitation de la HP-41 est, comme chacun le sait, la RAM disponible. Mes programmes dépassent 900 pas, et souvent je suis limité par la taille mémoire.

Questions :
-Comment disposer disposer d'une plus grande capacité mémoire, interieure ou extérieure, programmable en langage utilisateur, accessible et portable ?
-Un bricoleur sagace aurait-il encore à sa disposition les EPROMs MLE1H pour le boitier MLDL ? (ce qui me permettrait d'utiliser le mien)
-Des utilisateurs auraient-ils déjà programmé la HP-41 sur des problèmes liés aux techniques citées plus haut? Je serait heureux de pouvoir échanger points de vue, résultats, astuces et d'être conseillé sur ces matières.

Je suis aussi à la recherche de tous périphériques d'occasion, permettant d'étendre les possibilités de la HP-41, tels le lecteur de cassette, interface HP-IL <-> IBM PC, livrets 1 & 2 du module topographie, etc... Je suis déjà en possession d'un HP-41CV équipé d'un lecteur de cartes magnétiques, des modules d'extension fonction et mémoire, time, navigation, HP-IL, une imprimante type EPSON via une interface //, un port d'extension ainsi qu'un boitier MLDL (sans ses 2 ROM MLE1H). J'ai en plus à ma disposition pour ma gestion, un APPLE IIe et un compatible IBM XT.

Merci d'avance de vos réponses.

Claude Dupré
La Plardiére
Champgenêteux, F
53160 BAIS
tel: 43-37-08-30

COURRIER DU COEUR

Pierre DAVID
33, bd Saint Martin
75003-PARIS

Vend :

Imprimante ThinkJet HP 2225 : 2000 FF,
module Maths pour HP 71 : 500 FF.

Jean REIBEL
9 Square Victor Fleming
92350 LE PLESSIS-ROBINSON
Tél: (1) 46 31 46 11

Vend :

Pour HP-41 : module X-MEMORY 500 FF; module GAMES 150 FF; module ADVANTAGE 400 FF. Pour HP-71 : lecteur de cartes 500 FF; module FINANCE 500 FF; Périphérique HP-IL: interface vidéo 500 FF; Lecteur de cassettes 1000 FF.

Pierre Bourgot
2 sq castiglione
78150 Le CHESNAY
Tél:(1) 38 54 75 43

Vend:

Lecteur HP-81161A:1500 F
Imprimante HP-82162A:1500 F
Convertisseur 82166A:1200 F
Le tout en excellent état.

HP28

UN PEU DE SPORT

Lors des prochaines vacances, que ferez vous lorsque, allongé(e) sur la plage, la peau bien huilée, après avoir gréé votre planche à voile, bien ajusté les cambers afin de permettre un meilleur passage de la voile lors de l'empannage et que vous vous apercevrez qu'au niveau du vent c'est toujours la pétole? Relirez-vous pour la 17ème fois, le dernier *JPC*? (comment 17 seulement?)
Hé bien non! Nous vous proposons quelque chose à la saveur nouvelle. En effet nous avons découvert un livre...

- Qui vaut largement la planche-à-voile !
- Non ce n'est pas possible !!
- Mais si c'est possible !!!
  CRACK ! BOUM ! PAFF !!!

Après cet aparté entre les auteurs, revenons à nos moutons. Ce livre nous a permis de vivre une merveilleuse épopée à travers l'univers du soft et du hard, accompagnés par Hélène* et Patricia*. Les auteurs de l'ouvrage, Paul Courbis et Sébastien Lalande, nous invitent à découvrir les secrets de la HP-28 (C 1BB 1CC et S 2BB) en 272 pages. Une premiere partie de ce livre est consacrée à la structure de la mémoire : le soft. Chaque objet manipulé par la HP y est décrit avec précision; on y apprend comment est codé un objet, la façon dont il est stocké dans la RAM. Mais ce n'est pas tout! Les auteurs ce sont acharnés à décortiquer, pour vous, heureux lecteurs, la RAM et la ROM des trois modèles. Ainsi la RAM est exposée en long en large et même circulairement (buffer du clavier). On découvre alors la structure de l'écran, les adresses utiles et inutiles (pour utiliser le clavier et l'infra-rouge par exemple), comment est organisée la pile et encore beaucoup d'autre choses...

Muni de ces données, on peut alors se lancer dans la programation en assembleur en suivant les méthodes exposées dans le livre. Il devient alors possible de réaliser quelques effets surprenants tant sur le plan graphique que sonore et l'on peut donner libre cours à son imagination.

La seconde partie du bouquin est consacrée aux transformations physiques de la machine. Même si les HP-28 sont les plus belles, on peut faire mieux. En effet il peut être intéressant de démonter sa machine, d'y ajouter une alimentation externe, de l'accélérer ou de la connecter à un ordinateur ou à une autre HP. Toutes ces opérations sont rigoureusement expliquées et des photos de HP-28 écartelée aide à comprendre les opératins à effectuer.

Non contents de nous livrer des tonnes de renseignements, les auteurs nous servent tout prêts, plus d'une cinquantaine de programmes destinés à toutes les versions de la HP-28. Ces programmes sont surtout des utilitaires et des programmes mathématiques. Les utilitaires permettent de se balader dans la mémoire de la machine ou encore de provoquer quelques effets amusants. Un programme nommé SPEED acélère une HP-28S, oui cela veut bien dire que après avoir exécuté SPEED tous vos calculs et tous vos programmes s'exécuterons deux fois plus vite (attention à l'imprimante qui ne peut alors plus suivre). UNPIXEL est aussi trés utile, pour effacer un point; mais attention, en le tapant le lecteur est étreint d'un doute: s'il utilise CHK (programme de vérification des arguments dans la pile), doit-il ne taper que la première chaine de codes? La réponse est non, le faire est courir vers l'immonde Memory Lost, il faut taper le PEEK et ce qui va avec. Voici un tout petit exemple d'utilisation de SPEED et de UNPIXEL :

```
« X 1 - DUP COS R→C UNPIXEL X COS »
STEQ SPEED DRAW
```

Vous verrez se promener un ver sur l'ecran de votre HP (à condition d'être en mode radian, avec le PPAR par défaut et d'avoir tapé SPEED et UNPIXEL). Pour cet exemple SPEED est facultatif mais augmente considérablement la vitesse du petit ver.

Les programmes mathématiques, conseillés aux taupins et aux étudiants en sciences, concernent essentiellement les polynomes, leurs matrices associées et le calcul rationnel. Par exemple, le programme de calcul rationnel permet de transformer n'importe quel réel en la fraction la plus proche. Ainsi au lieu d'avoir un ignoble 1,33333333333 sur l'écran, vous aurez un merveilleux '4/3' qui ravira votre prof de maths adoré et ceci marche aussi bien avec les complexes (a/b+c/d*i), les vecteurs et matrices rééles ou complexes, et beaucoup plus fort les objets algebriques! Par exemple:

```
'TAN(X)' 'X' 5 TAYLR
```

retourne dans la pile :

```
'X+,333333333333*X^3+,133333333333*X^5'
```

en 35 s sans SPEED et en 17 avec (pour une HP-28S) au début on y croit pas mais c'est vrai! en exécutant ?→FR, on obtient en peu de temps `'X+1/3*X+2/15*X^3+2/3*X^515'`.

Enfin dans le ludique on trouve le mastermind, le carré magique (dont il faut souligner la structure de groupe commutatif) et le labyrinthe dont on ne sort

pas facilement. Nous avons failli oublier, pour les artistes, un programme de télécran est fourni, et pour les bidouilleurs un désassembleur.

Vous saurez tout lorsque nous vous aurons dit que le bouquin s'appelle *Voyage au centre de la HP-28C/S* et qu'il est édité par "La règle à calcul".

Laurent Chouraki (478)
Frederic Gobin

* Les auteurs de présent article font partie d'un groupe qui revendique le droit inaliénable aux HP-28 (toutes versions) d'avoir un prénom, au même titre que les autres êtres doués de raisonnement!

LE FACTEUR EST CONSTANT
ACTE V

Les programmes publiés récemment (page 193 et 265) dans le livre de Paul Courbis et Sébastien Lalande (*Voyage au centre de la HP 28 C/S*, les Editions de la Règle à Calcul) permettent d'étendre à la HP-28S le programme "le facteur est constant" publié en plusieurs volets dans *JPC*. Le programme a été légèrement modifié afin d'afficher les menus de façon plus condensée et afin de faciliter les changements du format d'affichage des résultats.

NOTE: Dans les lignes ci-dessous, les 'softkeys' sont désignées: {function}.

1) S'il reste assez de mémoire, l'invalidation suivante n'est pas requise: Invalider les modes COMMAND, UNDO & LAST.
Il est conseillé d'entrer les variables de droite à gauche, de bas en haut.

Cinquième ligne du menu

2) Dans HOME, taper: 'KTOP' CRDIR, puis presser {KTOP}. Ensuite:

3) Entrer le programme Z:
```
« 1 - 100 * »   Z [STO]
```

4) Entrer le programme Y:
```
« N O SWAP »   Y [STO]
```

5) Entrer le programme X:
```
« 3 DISP 1 WAIT KILL »   X [STO]
```

6) Entrer le programme W:
```
« R DUP DEPTH PICK →STR »   W [STO]
```

7) Entrer le programme V:
```
« % EXP Z ROT ROT »   V [STO]
```

8) Entrer le programme U:
```
« + SWAP →STR + q SWAP DUP INV 1 »   U [STO]
```

9) Entrer le programme R:
```
« DEPTH 3 IF >= THEN N END DUP2 »   R [STO]
```

10) Entrer le programme Q:
```
«
  DEPTH PICK SWAP DUP 'O' STO 60
  IF FS?
    THEN DEPTH PICK
  END
»
Q [STO]
```

11) Entrer le programme P:
```
«
  M ROLL DROP
  2 M
  FOR x
    DUP p →STR
    " %  " + SWAP q →STR + M ROLLD
  NEXT
  DEPTH
  ROLL →STR DEPTH ROLLD        (le cas échéant, PRSTC)
  KILL
»
P [STO]
```

12) Entrer le programme O:
```
1   O [STO]
```

13) Entrer le programme N:
```
« M ROLLD DEPTH 2 - DROPN »   N [STO]
```

13) Entrer le programme M:
```
« DEPTH 1 - »   M [STO]
```

Quatrième ligne du menu:

15) Entrer le programme q:
```
« 2 (ou 0, ou une autre valeur) FIX » ou « STD »
q [STO]
```

16) Entrer le programme p:
```
« 1 (ou 2) FIX DEPTH PICK SWAP %T »   p [STO]
```

17) Entrer le programme EFF:
```
«
  STD W "(N)/" U DEPTH PICK % * 1 +
```

```
  SWAP ^ Z 1 DEPTH PICK V
»
EFF [STO]
```

18) Entrer le programme NOM:

```
«
  q W "(E)/" STD U 1 DEPTH PICK % +
  LN * EXP Z * 1 OVER V
»
NOM [STO]
```

19) Entrer la variable B reservée au stockage de n'importe quel objet:

```
«»   B [STO]
```

20) Après avoir activé le mode hexa dans le menu BINARY (HEX), entrer l'assembleur:

```
«
  "69A20" SWAP "09F20" + + → z
  «
    PATH HOME HEX "" 1 z SIZE
    FOR I
      "#" z I DUP2
      1 + DUP SUB ROT ROT DUP
      SUB + + STR→ B→R CHR + 2
    STEP
    'J' STO # D0000h J SIZE
    2 * - SYSEVAL 1 GET 1 →LIST
    LIST→ DROP 'J' PURGE
    SWAP →STR 2 OVER SIZE 1 -
    SUB STR→
  »
»
A [STO]
```

Troisième ligne de menu:

21) Entrer le programme F:

```
«
  DUP IF
    THEN 'O' STO
    ELSE DROP O CLLCD X
  END
»
F [STO]
```

22) Entrer le programme %t:

```
«
  DEPTH 3
  IF >
    THEN PICK DUP CLLCD 2 DISP p
    →STR " %" + X
  END
  N p
»
'%t [STO]
```

23) Entrer le programme t:

```
« DEPTH PICK CLLCD X »   t [STO]
```

24) Entrer le programme o:

```
« 1 DEPTH 2 - FOR x x ROLL NEXT »   o [STO]
```

25) Entrer #, routine à double objectif: Initialisation de Σ par 0 ‹#›. Corbeille à papier (placer au niveau 1 le numéro de la ligne à supprimer, puis ‹#›).

```
«
  1 F DUP
  IF
    THEN
      M ROLL 1 - M ROLLD ROLL NEG DEPTH ROLL
      + DEPTH ROLLD
    ELSE CLEAR 0 0
  END
»
"21D201032" [ENTER] A [ENTER] [STO]
```

26) Entrer le programme Σ:

```
«
  M ROLL 1 + M ROLLD DEPTH DUP PICK - 2 MOD
  IF THEN * END
  O * DUP DEPTH
  ROLL + DEPTH ROLLD DUP
  IF THEN HALT ELSE DROP END
  P
»
Σ [STO]
```

Seconde ligne du menu:

27) Entrer le programme I%:

```
«
  DEPTH 2
  IF >=
  THEN
    R % DUP2 + ROT →STR "+" + ROT →STR + SWAP
  ELSE
    N DUP
    INV .01 - DUP INV SWAP .02 + INV
  END
»
'I% [STO]
```

28) Entrer .^. (puissance):

```
« Q ^ Y »
"21D2030E2E5E2" [ENTER] A [ENTER] [STO]
```

29) Entrer .μ. (routine Modulo et test de divisibilité):

```
« Q MOD Y »
"21D2030E2F8E2" [ENTER] A [ENTER] [STO]
```

30) Entrer le programme MU%:

```
«
  STD R SWAP →STR "-CP-" + SWAP
  →STR + q DEPTH PICK ROT DUP2
```

```
  %CH SWAP ROT %CH ABS 2 →LIST
»
'MU% [STO]
```

31) Entrer | (Bascule-Toggle pour installer K comme préfixe ou comme postfixe):

```
« 60 IF FS? THEN DEG ELSE RAD END » [ENTER]
"21D2010C7" [ENTER] A [ENTER] [STO]     C7=code pour "|"
```

32) Entrer K (la constante):

```
«
  DUP → z
  «
    IF
      THEN CLEAR z
      ELSE
      p 60
      IF FC?
        THEN SWAP
      END
    END
  »
»
K [STO]
```

Première ligne du menu:

33) Entrer ./. (division):

```
« Q / Y » [ENTER]
"21D2030E2F2E2" [ENTER] A [ENTER] [STO]
```

Entrer .*. (multiplication):

```
« Q * Y » [ENTER]
"21D2030E2A2E2" [ENTER] A [ENTER] [STO]
```

35) Entrer .-. (soustraction):

```
« Q - Y » [ENTER]
"21D2030E2D2E2" [ENTER] A [ENTER] [STO]
```

36) Entrer .+. (addition):

```
« Q + Y » [ENTER]
"21D2030E2B2E2" [ENTER] A [ENTER] [STO]
```

37) Entrer ■ (Bascule-Toggle pour installer K comme préfixe ou comme postfixe):
Taper «, puis presser la softkey {■}, puis [ENTER].

```
■ [STO]
```

38) Entrer k (la constante):

```
'K'   k [STO]
```

39) Entrer KT, le programme d'installation du menu CUSTOM, afin de ne pas afficher les routines de "service" suivantes: M,N,O,P,Q,R,U,V,W,X,Y,Z.

```
«
  { k ■ .+. .-. .*. ./. K | MU% .μ.
  .^. I% Σ # o t %t F A B NOM EFF p q }
  MENU
»
KT [STO]
```

40) Si on le désire, supprimer le contenu de A (l'assembleur).
L'assembleur peut aussi être conservé temporairement afin de créer des labels pour des fonctions qui stockeraient la constante K en haut de la pile.

MENUS

```
 Initialisations
   |==================================|
1) |  k  |  ■  | .+.   .-.   .*.   ./. | 'Functors'
   |     |     |-----|             |-----| arithmétiques
2) |  K  |  ■  | MU% | .μ.   .^.   I% |
   |----------Mark up----------Interêt
   |----------------------------------|
3) |  Σ    #    o    t    %t   |  F  | Portfolio
   |==================================|
4) |  A  |  B  | NOM   EFF |  p  |  q  |
   |-L i b r e-|----<->----|-pour FIX--|
   |==================================|
```

Exemples

Pour les exemples, se reporter aux numéros précédents de *JPC*.
Pour changer le FIX, 'VISIT'er le programme q.
Dans le programme Σ, pour changer le FIX pour pourcentages %T ou pour quantités, utiliser respectivement les programmes p et q.

Philippe Heilbronn (233)

CALCUL DE DATE ET FONCTIONS HORLOGES POUR HP28

Cet article se divise en deux parties:
- fonctions de calculs sur des dates,
- fonctions horloges.

1ERE PARTIE : CALCULS SUR LES DATES

Le but de cette partie est de fournir les équivalents des fonctions de calcul sur les dates du module Time du HP41, ou du lex DATELEX du HP71 (voir *JPC 49*).

Le format adopté pour coder une date est le format numérique *jj.mmaaaa* : par exemple, 11.021989 pour le 11 février 1989.

Fonctions implémentées :

DTADD (date add) : équivalent à DATE+ du HP-41, elle permet d'ajouter un nombre de jours à une date.
Utilisation:
Entrées : niveau 2 : date1
niveau 1 : n
Sortie : niveau 1 : date2 = date1 + n

DDAYS (difference of days) : différence de jours.
Donne le nombre de jours entre 2 dates.
Entrées : niveau 2 : date2
niveau 1 : date1
Sortie : niveau 1 : n = date2-date1

DOW (day of week) : donne le jour de la semaine correspondant à une date.
Entrée : niveau 1 : date
Sortie : niveau 1 : n = jour(date)
Le jour de la semaine est codé comme suit :
0 : dimanche, 1 : lundi, 2 : mardi, 3 : mercredi, 4 : jeudi, 5 : vendredi, 6 : samedi.

ADOW (alpha dow of week) :
Semblable à DOW, mais retourne une chaine contenant le jour en clair.
Entrée : niveau 1 : date
Sortie : niveau 1 : chaine jour
La chaine contient dimanche, lundi, ..., samedi.

Détails de réalisation:

Les calculs de date s'appuient sur la conversion d'une date en date julienne, c'est-à-dire en un nombre représentant le nombre de jours depuis une date référence.
Ces conversions sont effectués par les routines →JUL et JUL→

→JUL : conversion d'une date au format numérique *jj.mmaaaa* en jour julian.
entrée : niveau 1 : date (*jj.mmaaaa*)
sortie : niveau 1 : jour julian

JUL→ : conversion d'un jour julian en une date au format *jj.mmaaaa*
entrée : niveau 1 : jour julian
sortie : niveau 1 : date (*jj.mmaaaa*)

Les formules utilisées sont tirées du livret Utilities pour le HP-71, simplifiées et adaptées au langage RPL. Ces formules sont assez complexes, mais les calculs de date deviennent élémentaires avec ces deux procédures :

Par exemple, DTADD est défini par :

SWAP	met la date au niveau 1
→JUL	conversion en jour julian
+	addition du nombre de jours
JUL→	conversion inverse : date résultat

Traitement des erreurs :

Le domaine de validité des dates est de 4.101582 à 31.129999 . Le traitement des erreurs se fait au niveau des conversions →JUL et JUL→. JUL→ teste que l'argument fourni est bien dans l'intervalle autorisé. →JUL utilise l'astuce suivante pour tester que la date *jj.mmaaaa* est valide: →JUL convertit la date en jour julian, puis effectue la conversion inverse, si le résultat est identique à la date originale, c'est que cette date est valide. En effet, une date invalide telle que 34.011988 ou 29.021989 ne sera jamais produite par la routine JUL→ .

J'ai rencontré une difficulté pour indiquer l'erreur à l'utilisateur: j'ai tout d'abord pensé utiliser une séquence du type :

```
"ERREUR : date incorrecte" 1 DISP ABORT
```

pour afficher un message d'erreur à la ligne 1, et arreter l'éxécution. Malheureusement, le mot ABORT efface le flag système message, et l'affichage normal de la pile est restaurée, faisant disparaitre mon message. J'ai donc dû recourir à une solution moins élégante, qui consiste à mettre une chaine contenant le message d'erreur au niveau 1. J'ai fait en sorte de restaurer l'argument ayant provoqué l'erreur au niveau 2.

Les deux situations d'erreur sont :

niveau 2 : date *jj.mmaaaa*
niveau 1 : "Invalid Date"
Cette erreur est provoquée par une date incorrecte telle que 12.341988 .

niveau 2 : jour julian
niveau 1 : "Out Of Range Date"
Cette erreur est provoquée par un jour julian correspondant à une date hors de l'intervalle du 04/10/1582 au 31/12/9999.

2EME PARTIE : FONCTIONS HORLOGES

Cette partie traite de l'implémentation des fonctions horloges. Ce sujet a déjà été abordé dans *JPC* (voir *JPC* 46, 49 et 53). Je propose ici un ensemble complet de fonctions, ainsi que l'accès à la date, et à un chronomètre.

Les formats utilisés sont ceux du module time du HP-41 soit:
un nombre *hh.mmsscc* pour les heures,
un nombre *jj.mmaaaa* pour les dates.

Le chronomètre :

Commencons par définir le chronomètre. L'accès se fait par le mot SW (stopwatch : chronomètre en anglais). C'est la seule définition dépendant de la version de votre machine. SW fournit un entier binaire représentant la valeur instantanée du chronomètre en 1/8192 ème de seconde.

Pour mesurer des intervalles de temps, on utilise le mot ELAPSE :
ELAPSE prend sur la pile la valeur du chronomètre à un instant t, et retourne le temps écoulé depuis cet instant en secondes.

La mesure d'un temps se fait en mémorisant la valeur du chronomètre au début du phénomène à mesurer avec SW, puis à calculer le temps écoulé en fournissant cette valeur à ELAPSE.

Exemple : mesurer le temps d'exécution d'une boucle :

```
SW              valeur du chronomètre au niveau 1
0 100
START NEXT      n'importe quoi laissant le niveau 1 intact
ELAPSE          donne le temps écoulé
```

Sur ma machine (HP-28S), j'obtiens 0,52 seconde.

Fonctions date et heure.

DATE retourne la date actuelle au format *jj.mmaaaa*.
Sortie : niveau 1 : date *jj.mmaaaa*

TIME retourne l'heure courante au format *hh.mmss*cc.
sortie : niveau 1 : heure *hh.mmsscc*

SETDATE permet de mettre l'horloge à la date.
entrée : niveau 1 : date *jj.mmaaaa*

SETIME permet de mettre l'horloge à l'heure.
entrée : niveau 1 : heure *hh.mmss*

ADATE convertit une date au format numérique *jj.mmaaaa* en une chaine au format jj/mm/aaaa
entrée : niveau 1 : date *jj.mmaaaa*
sortie : niveau 1 : chaine jj/mm/aaaa

ATIME convertit une heure au format numérique *hh.mmss* en une chaine au format hh:mm:ss
entrée : niveau 1 : heure *hh.mmss*
sortie : niveau 1 : chaine hh:mm:ss

Détails de réalisation :

Pour implémenter la date, il est nécessaire d'avoir entré les routines →JUL et JUL→ définies dans la 1ere partie.

Il faut définir une variable EXACT :

```
0 'EXACT' STO
```

Les fonctions TIME et DATE font appel à une fonction DTTIM qui utilise SW et la valeur de EXACT pour retourner:
niveau 2 : secondes depuis minuit
niveau 1 : jour julian

Réciproquement, SETDATE et SETIME font appel à SETDT qui demande :
niveau 2 : heure au format *hh.mmss*
niveau 1 : date au format *jj.mmaaaa*
et utilise ces informations pour ajuster la valeur de EXACT.

Exemple :
Remise à l'heure et à la date le 21/12/88 à 22:52:25

```
22.5225 21.121988 SETDT
```

Application : affichage d'une horloge.

Le programme CLOCK affiche la date et l'heure. L'appui sur une touche quelconque arrête le programme.

Références :
- *Manuel du module Time HP41*
- *HP71 Users' Library Solutions : Utilities*
- *JPC 46* p.7, *JPC 49* p.6, *JPC 53* p.6 : horloge pour HP-28
- *JPC 49* p.24 : Lex DATELEX de calcul sur dates pour HP-71.

Listings 1ère partie :

```
JUL→
« → x
  « x
    IF DUP 2299150 <
 SWAP 5373484 > OR
    THEN x
 "Out Of Range Date"
```

```
   ABORT
      END x 68569 +
   36524.25 MOD IP DUP
   1 + 365.25025 / IP
   SWAP OVER 365.25 *
   IP - DUP 31 + 30.59
   / IP SWAP 31 + OVER
   30.59 * IP - ROT 3
   PICK 11 / IP + x
   68569 + 36524.25 /
   IP 49 - 100 * + ROT
   DUP
      IF 11 <
      THEN 2 +
      ELSE 10 -
      END 100 / SWAP
   1000000 / + +
   »
»



→JUL
«  → x
   «  x 100 * FP 10000
   * x FP 100 * IP
      IF DUP 3 <
      THEN 12 + SWAP 1
   - SWAP
      END OVER 365.25
   * IP SWAP 1 +
   30.6001 * IP + x IP
   + 1720995 + SWAP DUP
   400 / IP SWAP 100 /
   IP - 2 + + DUP JUL→
   x
      IF <>
      THEN DROP x
   "Invalid Date" ABORT
      END
   »
»
```

Note : le signe < > représente le symbole 'différent de' (touche [shift] [=]) du HP-28.

```
DOW
«  →JUL 1 + 7 MOD »


ADOW
«  DOW 1 + {
"dimanche" "lundi"
"mardi" "mercredi"
"jeudi" "vendredi"
"samedi" } SWAP GET »
```

```
DDAYS
«  →JUL SWAP →JUL SWAP - »

DTADD
«  SWAP →JUL + JUL→ »
```

Listings 2ème partie :

```
SW
```

Selon la version de la machine :
HP28C 1BB : « #123E SYSEVAL » (hex)
HP28C 1CC : « #1266 SYSEVAL » (hex)
HP28S 2BB : « #11CAh SYSEVAL »

```
ELAPSE
«  SW SWAP - B→R 8192 / »
```

Création de la variable EXACT

```
0 'EXACT' STO

DTTIM
«  SW EXACT -
 # 2A300000h DUP2 /
 DUP ROT * ROT SWAP -
 B→R 81.92 / IP 100 /
 SWAP B→R
»

SETDT
«  SW ROT ROT →JUL 24
 * SWAP HMS→ + 3600 *
 8192 * - 'EXACT' STO
»

SETDATE
«  TIME SWAP SETDT »

SETIME
«  DATE SETDT »

DATE
«  DTTIM SWAP DROP JUL→ »

TIME
«  DTTIM DROP 3600 / →HMS »

ADATE
«  → x
   «  x IP →STR "/" +
      x FP 100 * IP →STR +
      "/" + x 100 * FP
      10000 * IP →STR +
   »
»

ATIME
«  → x
   «  x IP →STR 58 CHR
```

```
      + x FP 100 * IP →STR
      + 58 CHR + x 100 *
      FP 100 * IP →STR +
    »
»

CLOCK
«  RCLF STD CLLCD
  DATE DUP ADOW " " +
  SWAP ADATE + 1 DISP
  DO TIME ATIME 2
  DISP .65 WAIT
  UNTIL KEY
  END DROP STOF
»
```

Jean François Garnier (242)

CRWTH-LISP 3.2

NDLR: Cet article nous est parvenu par le réseau Unix (merci Pierre!). Comme désiré par l'auteur, nous l'avons traduit tel quel, sans rien ajouter ni retrancher. Vous pouvez cependant relire l'article de Serge Vaudenay dans JPC 54.

Cwrth-Lisp 3.2

Un compilateur Lisp pour HP-28S

par Olle Gallmo (Crwth)

Septembre 1988

Préface (Changements depuis la version 3.1)

DEFMAC a été ajouté. Vous pouvez maintenant définir des macros en Crwth-Lisp

La bogue dans COND a été corrigée. En Cwrth-Lisp 3.1, on ne pouvais pas avoir une séquence d'appels de fonctions dans la partie conséquence d'une COND.

COND ne retourne rien si aucun des tests est vrai. Crwth-Lisp 3.1 renvoyait NIL. La raison pour laquelle Crwth-Lisp 3.1 faisait ceci était que je voulais que COND retourne toujours quelque chose, mais je ne peut pas être sûr de ceci quand ça peut appeler une fonction HP-28 qui ne retourne rien.

La partie HP-28C a été retirée du manuel, depuis que le compilateur occupe environ 1.4Kb, il n'aurait plus aucun interet sur cette version.

1. Introduction

Le but que je m'était fixé était de créer un langage Lisp pour le HP-28. Je le voulais aussi compact et efficace que possible pour qu'il soit plus qu'un simple jouet.

Premièrement j'ai fait un petit interpréteur avec lequel je peux appeler la plupart des fonctions internes du HP-28 comme des expressions Lisp (L-expr), mais j'ai vite réalisé qu'un compilateur serait plus facile à écrire, et bien sûr plus efficace !

J'en suis arrivé à un compilateur qui me donne un contrôle *à peu près* total du HP-28 et qui produit un code HP-28 *à peu près* optimal.

2. Description

Crwth-Lisp diffère un peu des Lisps conventionnels. Les différences principales sont la syntaxe, la représentation des listes et les "truthvalues".

Une liste est un équivalent des listes HP-28, c'est à dire une liste est notée par des accolades et il n'y a ni guillemets ni délimiteurs.

NIL et {} ne sont *pas* équivalents !

{} est une liste vide mais *non* une "truthvalue".

NIL est une "truthvalue", équivalente au 0 (zéro) du HP-28, et *non* la liste vide.

l'atome T est compilé en 1 (un) du HP-28.

J'ai choisi cette représentation de "truthvalue" parce que je veux rendre facile l'utilisation des prédicats du HP-28 à partir du Lisp, et vice versa, sans avoir à effectuer de conversion entre les differentes des "truthvalues".

Le HP-28 crie Syntax Error si vous tentez de placer sur la pile une liste contenant un atome dans le menu BRANCH. Il n'est pas possible d'appeler cette fonction HP-28 à partir du Crwth-Lisp.

Le compilateur est très petit. Il doit seulement connaitre quelques formes spéciales. Il n'est pas nécessaire de définir FIRST, REST, CONCAT, et tout ce qui peut fait en Crwth-Lisp par l'appel des fonctions internes du HP-28.

les formes spéciales actuellement définies sont: QUOTE, DEFUN, DEFMAC, LET et COND.

3. Le compilateur

LISP - Compile une L-expr en un bloc programme, et l'évalue.
COMP - Compile une L-expr en une chaine d'instructions HP-28.
CLIST - Compile une liste.
CATOM - Compile un atome.
CFUN - Compile un appel de fonction.
CSEQ - Compile une séquence d'appel de fonctions.
SPQ - Compile une forme spéciale QUOTE.
SPDEF - Compile une forme spéciale DEFUN.
SPMAC - Compile une forme spéciale DEFMAC.
CDEF - Compile DEFUN ou DEFMAC sans stocker le résultat.
SPLET - Compile une forme spéciale LET.
SPCOND - Compile une forme spéciale COND.
SPA - Une liste contenant les noms Lisp des formes spéciales.
SPB - Une liste contenant la fonction compilateur qui compile une forme spéciale.
MACA - Une liste contenant les noms des macros.
MACB - Une liste contenant les définitions des macros.
->PROG - Place les délimiteurs de blocs programme autour d'une chaine de code HP-28.
REST - Retourne la queue d'une liste.

LISP est la seule fonction que vous avez besoin de connaitre. Elle prend une L-expr comme argument. Si la L-expr est une DEFUN, la fonction sera compilée en code HP-28 et stockée dans une variable. Si la L-expr est une DEFMAC, la fonction sera compilée et stockée dans les variables MACA et MACB (voir section 6). Si la L-expr est autre chose, Crwth-Lisp agira comme un interpréteur, c'est à dire qu'il compilera la L-expr en code HP-28 et l'évaluera immédiatement.

4. Factorielle -- Un exemple

Placer la définition d'une factorielle sur la pile.

```
{DEFUN FAC {X}
  {COND {{== X 0} 1}
    {T {* X {FAC {- X 1}}}}}}
```

Maintenant appeler Lisp. Après un moment, FAC apparait comme une nouvelle variable dans le menu, et Lisp retourne 1 (qui est identique à T, vous vous rapellez ?).

Visitons FAC pour voir à quoi il ressemble.

```
« -> X
  «
    IF X 0 == THEN
      1
    ELSE
      X X 1 - FAC *
    END
  »
»
```

Si vous écrivez une factorielle en code HP-28, vous ne voudrez probablement pas nommer l'argument. Vous préférerez opérer directement sur la pile (au moins je le pense).
Une chose en commun dans toutes les fonctions Crwth-Lisp est qu'elles prennent toujours les arguments dont elles ont besoin sur la pile, et les stockent dans des variables locales, au lieu d'opérer directement sur la pile (bien qu'il soit possible d'opérer sur la pile en Crwth-lisp, c'est ce que j'appelle de la programation *sale*).

Maintenant, si vous voulez calculer 5!, vous pouvez le faire de deux façons:

-Comme fonction HP-28, en plaçant 5 sur la pile et presser FAC.
-Comme fonction Crwth-Lisp, en placant {FAC 5} sur la pile et presser Lisp.

la deuxième méthode prendra un peu plus de temps, {FAC 5} étant compilé en code HP-28 avant exécution.

5. Variables locales

Vous povez créer des variables locales en Crwth-Lisp avec la forme spéciale LET. Elles agissent comme avec le Common-Lisp LET.

Ex:

```
{DEFUN FOO {X}
    {LET {{A 5}  ; A et B sont des variables locales.
      {B 42}}
     {* {+ X B} A}}}
```

Le code compilé utilise les moyens de la HP-28 pour créer les variables locales:

```
« -> X
  « 5 42 -> A B
    « X B + A *
    »
  »
»
```

6. Macros

L'égalité est notée '==' en code HP-28, mais en Crwth-Lisp vous voudrez peut être l'appeler 'EQ'. Vous pouvez bien sûr définir EQ avec DEFUN comme ceci:

```
{DEFUN EQ {X Y} {== X Y}}
```

mais ceci fera un code HP-28 plus lent si vous utilisez beaucoup EQ et quand vous aurez à déplacer votre super-programme vous aurez à emmener votre *petite* définition avec vous, ou il la ne trouvera pas quand EQ sera appelée.

Définissons plutot EQ comme une macro (avec DEFMAC):

```
{DEFMAC EQ {X Y} {== X Y}}
```

EQ ne sera pas stockée comme une fonction, mais comme une macro. Le nom EQ sera ajouté à la liste MACA, et le code compilé sera ajouté comme une chaine à la liste MACB.

Maintenant, quand vous compilerez un programme qui appele EQ, l'appel de fonction sera substitué par le code présent dans la liste MACB.

Mauvais conseil : vous pouvez duper le compilateur et définir EQ comme {DEFMAC EQ {} {==}}, ce qui inhibe le nom des paramètres.

De trés courtes fonctions, ou du code HP-28 que vous voudrez appeler quelque part dans Crwth-Lisp, devraient plutot être définies comme macros.

Si vous définissez une macro dans le même répertoire que le compilateur, les listes MACA et MACB du compilateur seront élargies. Le seul moyen de retirer une définition de macro serait d'éditer les listes MACA et MACB. Mais si vous définissez les macros dans un sous-répertoire, les nouvelles listes MACA et MACB seront stockées dans le sous répertoire. Alors vous pourrez les purger quand vous voudrez retirer les macros, les listes globales MACA et MACB ne seront pas touchées dans le répertoire du compilateur.

Attention: Ne purgez pas les listes MACA et MACB du répertoire du compilateur, il en a besoin!

7. Commentaires

C'est une bonne idée de créer vos programmes Crwth-Lisp dans un sous-répertoire du compilateur, vous n'endomagerez pas accidentellement votre compilateur. J'ai créé un répertoire Lisp pour le compilateur qui a un sous-répertoire SRC et un sous-répertoire BIN.

Si une fonction Crwth-Lisp appelle une procédure HP-28 qui ne retourne rien (DISP par exemple), le Crwth-Lisp ne retournera rien.

Souvenez vous de traiter NIL comme une "truthvalue" et une "truthvalue" *seulement* !

Ceci est un avant-propos, non un manuel complet.

Crwth (prononcez 'croth') est une ancienne lyre celtique

8. Programme

Bon, et maintenant : Crwth-Lisp 3.2 !

L'impact que Crwth-Lisp 3.1 a produit sur le réseau est allé au delà de mes rêves les plus fous !

Suggestions, "bugs-reports", insultes, lettres d'amour sont les bienvenues, (bien que j'espere ne pas recevoir de "bug-reports" !), mais doivent être écrites en Anglais (ma connaissance du français est NIL (la "truthvalue" *and* la liste vide !)).

Les changements effectués depuis Crwth-Lisp 3.1 sont listés dans la préface du manuel.

Crwth

```
                    ; Crwth-Lisp 3.2
                    ; A Lisp Compiler for the HP28S
                    ; by Olle Gallmo (Crwth) 1988
                    ; -----
LISP ( L-expr --- Value )     ; Compile une L-expr
                                en un bloc programme
« COMP →PROG STR→ EVAL    et l'évalue
»
                              ; -----

COMP ( L-expr --- String )    ; Compile une L-expr
                                en une chaine
« IF DUP TYPE 5 SAME THEN       d'instructions HP-28
    CLIST                     ; Listes
  ELSE
    IF DUP TYPE 6 SAME THEN ; Atomes
      CATOM
    ELSE
      IF DUP TYPE 2 SAME THEN ; Chaines
        34 CHR SWAP OVER + +  ; Une chaine dans une
      ELSE                      chaine!
        →STR                ; Autres
      END
    END
  END
  " " +
»
                              ; -----

CLIST ( List --- String )     ; Compile une liste
« IF DUP {} SAME THEN         ;Une liste vide compilée
    DROP "{}"                   dans elle même
```

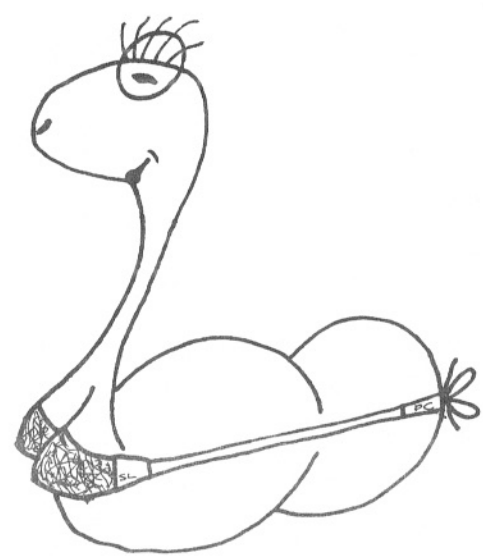
```
   ELSE
     DUP REST SWAP 1 GET     ; Prend la tête et la
                               queue
     IF SPA OVER POS THEN    ; Si la fonction est
                               une forme spéciale
       SPB SPA ROT POS GET EVAL ; compile une forme
                               spéciale
     ELSE
       CFUN                  ; Sinon compile un
     END                       appel de fonction
   END                         normal
»
                             ; -----


CATOM ( Atom --- String )    ; Compile un atome
« IF MACA OVER POS DUP THEN  ; Si l'atome est défini
                               comme une macro
     SWAP DROP MACB SWAP GET ; prend la chaine
   ELSE                      ; correspondante
     DROP →STR 2 OVER SIZE 1 - SUB   ; Sinon
   END                         convertit en chaine
»                              et efface les " ' "
                             ; -----


CFUN ( Arglist Functor --- String ) ; Compile
                               un appel de fonctions
« COMP SWAP                  ; Compile la fonction
   IF DUP SIZE THEN          ; Si la fonction a
                               des arguments
     CSEQ SWAP +             ; les compiler en premier
   ELSE                        et ajoute le fonctor
     DROP                    ; Sinon retourne juste
   END                         le fonctor
»
                             ; -----


CSEQ ( Arglist --- String )  ; Compile une séquence
« "" 1 3 PICK SIZE FOR i       d'appels de fonctions
     OVER i GET COMP +         (pour l'instant le
   NEXT                        corps d'un DEFUN, LET
   SWAP DROP                   ou COND)
»
                             ; -----


SPQ ( Arglist --- String )   ; Compile la forme
« 1 GET →STR                   spéciale QUOTE
»
                             ; -----


SPDEF ( Arglist --- String ) ; Compile la forme
                               spéciale DEFUN
« DUP CDEF →PROG STR→        ; Compile la définition
   SWAP 1 GET STO "1"        ; Stocke dans une
»                              variable
                             ; -----
```

```
SPMAC ( Arglist --- String ) ; Compile une forme
                               spéciale DEFMAC
« DUP CDEF MACB + 'MACB' STO ; Compile la
                               définition dans MACB
   1 GET MACA + 'MACA' STO "1" ; et stocke le nom de
»                              la macro dans MACA

CDEF (Arglist --- String )   ; Compile une définition
                               (DEFUN or DEFMAC)
« DUP 3 OVER SIZE SUB CSEQ   ; Compile le corps
   SWAP 2 GET                ; Prends la liste de
                               paramètres
   IF DUP SIZE THEN          ; Si la fonction prend
                               des arguments
     CSEQ "→ " SWAP +        ; crée leurs variables
                               locales
     SWAP →PROG +            ; et ajoute le
                               corps compilé
   ELSE                      ; Sinon
     DROP                    ; laisse juste le
   END                         corps compilé
»
                             ; -----


SPLET ( Arglist --- String ) ; Compile la forme
                               spéciale
« DUP REST CSEQ →PROG        ; Compile le corps
   SWAP 1 GET                ; Prend les déclarations
   "" 1 3 PICK SIZE FOR i    ; Compile les valeurs
     OVER i GET 2 GET COMP +   des variables locales
   NEXT
   "→ " +
   1 3 PICK SIZE FOR i       ; Crée le code HP-28
                               correspondant
     OVER i GET 1 GET COMP + ; variables locales
   NEXT
   SWAP DROP SWAP +          ; et ajoute le corps
»                              compilé
                             ; -----


SPCOND ( Arglist --- String ); Compile la forme
                               spéciale COND
« IF DUP SIZE 1 SAME THEN    ; If only one test left
     1 GET DUP 1 GET COMP
     IF DUP "1 " SAME THEN   ;   SI c'est un 'else'
      DROP REST CSEQ         ;     compile seulement
     ELSE                    ;   Sinon
      "IF " SWAP + "THEN " + ;     Crée un IF-THEN-END
      SWAP REST CSEQ + "END " +
     END
   ELSE                      ; Sinon
     "IF " OVER 1 GET 1 GET COMP +  ;   Crée un
                                  IF-THEN-ELSE-SPCOND-END
     "THEN " +
     OVER 1 GET REST CSEQ +
     "ELSE " +
     SWAP REST SPCOND +
     "END " +
```

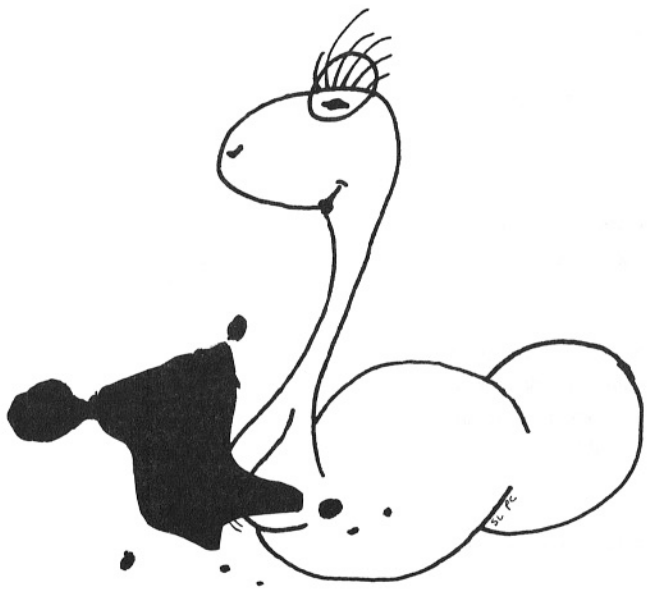
```
   END
»
                              ; -----

SPA                           ; Liste des formes
{ QUOTE DEFUN DEFMAC LET COND }   spéciales
                              ; -----

SPB                           ; Liste des mots
                                compilants correspondants
{ SPQ SPDEF SPMAC SPLET SPCOND }
                              ; -----

MACA                          ; Liste des macros
{ T NIL }
                              ; -----

MACB                          ; Liste des chaines
{ "1" "0"}                    correspondantes
                              ; -----

→PROG ( String --- String )   ; Place les delimiteurs
« "« " SWAP + "» " +            de programme des deux
»                               cotés d'une chaine de
                                code HP-28
                              ; -----

REST                          ; Prend la queue d'une
« 2 OVER SIZE SUB               liste
»
                              ; -----
```

Olle Gallmo (Crwth)
Skyttevagen 68
181 46 LINDINGO
Suède

Email: crwth@kuling.UU.SE



En page 16 j'enlève le haut...



JPC le journal qui tient ses promesses...

Note de l'imprimeur: Veuillez nous excuser pour cette tache totalement independante de notre volonté !

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). Vous pouvez taper vos articles sur IBM PC, mais dans ce cas transmettez les nous sur disquette 3½". 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.

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)
```

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 mai 1989
Samedi 3 juin 1989

Pierre David (37)

NOUS EN AVONS

La coopérative du Club vous propose :

- des **anciens numéros** de JPC, au prix de 40 F + 7,40 F de frais d'affranchissement,

- d'une **année complète** de numéros de JPC (février à janvier) pour 300 F (offre spéciale) port compris,

- de **manuels de service** du HP-41 au prix de 75 F (port compris),

- de **manuels de service** du HP-75 au prix de 75 F (port compris).

Vous pouvez aussi bénéficier de la **Programmathèque HP-71**, regroupant tous les Lex et programmes pour HP-71 et HP-75 parus à ce jour dans *JPC*. Joindre 3 disquettes 3½" à votre reglement.

Si vous souhaitez des renseignements complémentaires, n'hésitez pas à nous contacter.

VOUS EN VOULEZ

Nom :
Prénom :
No de membre :
Adresse :

Commande :

	Qté	Prix Unitaire	Prix Total
anciens numéros de JPC	x	47,40 FF	
année complète de JPC	x	300 FF	
Programmathèque HP-71 (joindre 3 disquettes)	x	75 FF	
Manuel de service pour HP-41	x	75 FF	
Manuel de service pour HP-75	x	75 FF	
Actualisation Eprom	x	150 FF	
		Total	FF

Préciser éventuellement les
numéros de JPC commandés :

PPC PARIS

Association régie par la loi de 1901, enregistrée
à Paris le 2 décembre 1982 sous le numéro 82/3240

BULLETIN D'ADHESION

```
Nom        |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
Prénom     |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_| Date de naissance |_|_|/|_|_|/|_|_|
Adresse    |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
           |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
           |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
           |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
Code Postal |_|_|_|_|_| Ville |_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|
Téléphone  Domicile |_|_|_|_|_|_|_|_|_|_|_|_|_|  Bureau |_|_|_|_|_|_|_|_|_|_|_|_|_|
```

Profession ____

Intérêts ____

____

Matériel HP en votre possession ______________________________

____

Autre matériel informatique ______________________________

____

Comment avez-vous connu PPC Paris ? ______________________________

____

Que recherchez-vous au sein de PPC Paris ? ______________________________

____

____

La Loi No 78-17 du 6 janvier 1978 relative à l'informatique, aux fichiers et aux libertés, garantit à toute personne justifiant de son identité un droit d'accès et de rectification auprès des services ou organismes chargés de mettre en oeuvre des traitements informatiques comportant des informations nominatives le concernant.

Je souhaite adhérer au club PPC Paris conformément aux statuts de l'Association. Au mieux de ma connaissance, je déclare avoir le droit de fournir tous les programmes et informations que je vous enverrai (sans enfreindre des obligations de secret à l'égard d'autres personnes ou organismes) pour publication dans le Journal de liaison, sans obligations ni responsabilité d'aucune sorte (en cas d'utilisation frauduleuse) de la part des dirigeants de PPC-Paris.

Date |_|_|/|_|_|/19|_|_|

Signature, précédée de la mention "Lu et Approuvé" ____________________

Le montant de la cotisation s'élève à 350.00 F pour un an.
Etudiants: 300.00 F (justificatif indispensable)

Paiement à l'ordre de "PPC Paris" par chèque bancaire ou virement postal (CCP No 18 823 40 C à Paris). Ne pas utiliser d'Eurochèques.

Veuillez envoyer toute correspondance à :
PPC Paris, BP 604, 75028 Paris Cedex 01, France

HP41

LA METHODE DE HAHN

En 1946, M. L. Hahn a publié la circulaire Z, No 16 (Institut Technique du Bâtiment et des Travaux Publics) ayant le titre *Détermination des contraintes dans un massif de fondation rectangulaire soumis à des charges excentrées*.

Avant de présenter le programme HP-41, pour utiliser la méthode de Hahn, on commence par une courte introduction concernant la procédure (fig.1 a,b,c). Quand un massif de fondation de type dalle est soumis a des charges excentrées on distingue trois cas définis par la position de l'axe neutre (DE sur la fig.1) qui sépare la zone tendue de celle comprimée (hachurée sur la fig.1). Suivant que la charge est localisée dans la zone I ou II ou III , l'axe neutre découpe une zone comprimée de contour triangulaire, trapézoîdale ou pentagonale (fig. 2 et 1). Dans la zone comprimée la dalle développe des pressions spécifiques variables d'un point à un autre. Ces pressions qui s'annulent sur l'axe neutre forment, si elles sont représentées graphiquement perpendiculairement au plan de la section, une pyramide des sollicitations qui doivent équilibrer la force appliquée.

Pour obtenir les relations capables de nous fournir la solution on utilise, pour chacun des trois cas suscités, l'equivalece de l'effort appliqué avec la résultante des pressions spécifiques et la coincidence du centre de gravité des pressions spécifiques avec le point d'application de l'effort appliqué.

Considérons l'exemple de l'article de Hahn: une dalle 700x350 cmxcm et une force appliquée P=1482 t. Le programme présenté ne traite pas la zone 1, parce que pour cette zone la solution est immédiate. Si Rb est la pression maximale qui correspond au sommet B de la pyramide et b' et h' sont les coordonnées du point d'application de la force par rapport aux cotés de la section, alors on a Rb=3P/(8b'h'). Avec b'=66 cm et h'=87 cm, alors Rb=96.8 kg/cm2. Le deuxième cas ou l'axe neutre découpe un quadrilatère est plus compliqué. La position de la force appliqué est donnée par une abscisse x0, par rapport à l'axe de symétrie verticale de la dalle, et une ordonnée h', par rapport au coté supérieur. On a pour l'exemple de Hahn x0=23 cm et h'=150 cm. La position de l'axe neutre est déterminée par α et β, (deux parametres) et la pression maximale au coin B de la section Rb=λP/(bh') (h"=αh' et h"'=βh'). L'étiquette A du programme calcule et affiche α, β et λ en fonction de ξ=x0/b=23/350=.0657 (STO 01), c'est à dire d'abord α=2.339; β=3.507; λ=.81 et ensuite, avec P en (STO 9), h' en (STO 21) et b en (STO 15) on détermine Rb. Les valeurs correspondantes de l'exemple de Hahn sont respectivement 2.3392; 3.5066; 0.8101 et 229 t/m2.

On a h"=2.339*150=351 cm et h"'=3.506*150=526 cm.

On passe maintenant au troisième cas qui est le plus compliqué. Hahn a résolu ce cas en utilisant des abbaques. Le programme qui suit est basé sur un autre algorithme. Considérons comme point de départ les relations utilisées par Hahn (13,18,19) qui sont les suivantes:

$$\xi = .25(2\delta - 2\alpha^2\delta^2 + \alpha^3\delta^2)/n$$
$$\eta = .25(2\alpha - 2\alpha^2\delta^2 + \alpha^2\delta^3)/n$$

où $n = \alpha^2\delta^2 - 6\alpha\delta + 3(\alpha+\delta)$

et $\mu = 6(\alpha+\delta-\alpha\delta)/n$

(On a noté avec ξ et η les valeurs denomées 'csi' et 'éta')

Les valeurs données sont 'csi' et 'éta' et les valeurs recherchées sont α et δ. Une fois ces valeurs connues on obtient directement μ. On considère qu'on a deux relations de la forme: $\xi = f(\alpha,\delta)$ et $\eta = \phi(\alpha,\delta)$. On peut écrire $f(\alpha+k,\delta+h) = f(\alpha,\delta) + kf'/\alpha + hf'/\delta$ (ou $f'/\alpha = \partial f/\partial\alpha$) et $\phi(\alpha+k,\delta+h) = \phi(\alpha,\delta) + k\phi'/\alpha + h\phi'/\delta$.
En considérant connus α et δ on peut déterminer par itérations 'csi'(i), 'eta'(i) et ensuite k et h qui constituent de valeurs permettant d'obtenir un α et δ améliorés. La procédure itérative s'arrête quand 'csi'(i) et 'eta'(i) coincident pratiquement avec les valeurs données 'csi' et 'eta'.

L'organisation du programme est la suivante:

L'étiquette C calcule 'csi'(i) et 'eta'(i) pour α en (STO 2) et δ en (STO 3).
A l'étiquette D on calcule k et h; le programme place 'csi' en (STO 20) ,'eta' en (STO 21), 'csi'(i) en (STO 10), 'eta'(i) en (STO 11). Les derivées partielles de f et ϕ sont en (STO 23,24,25,26) et k et h en (STO 18,19). Comme l'abbaque IIIbis de Hahn le montre α et δ sont des valeurs plus petites que 1; en conséquence toute valeur plus petite que 1 peut être utilisée. L'appendice montre les resultats obtenus de l'étiquette C ,qui s'exécute en même temps que D; d'abord avec les valeurs initiales α=.2 et δ=.8 et après avec α=.2 et δ=.2 . Les resultats sont les mêmes dans les deux cas et coincident avec ceux de Hahn;

On a:

dans les registres 2 et 3 respectivement: α=.2527 (Hahn .253) et δ=.6938 (Hahn .694) et avec ces valeurs l'étiquette B permet de calculer μ=2.5447

(Hahn 2.53). Si on exécute maintenant l'étiquette E on obtient les différences (erreurs d'approximations).

Enfin si on introduit dans les registres 9,15,16-P,b,h, alors l'étiquette F imprime succesivement la pression maximale, les pressions aux sommets C et F le volume de la pyramide des sollicitations (qui égale l'effort appliqué). La dernière valeur imprimée est la pression moyenne.

APPENDICE

Zone 2
(I) $A\alpha^2+B\alpha+C=0$
où $A=\xi(1+4\xi^2)$ (STO 02)
$B=12\xi^2-4\xi+1$ (STO 03)
$C=12\xi-3$ (STO 04)
donc $\alpha=[-B+SQRT(B^2-4AC)]/2A$ (STO 05)
et
(II) $\beta=(2\alpha+1/\xi-4)\alpha/[1/\xi-4(1+\alpha\xi)]$ (STO 06)
(III) $\lambda=6\beta/(\beta^2+\alpha\beta+\alpha^2)$ (STO 07)
Les formules I et II sont celles de HAHN (page 3) la formule III est equivalente a (8).

Zone 3
Les resultats des itérations quand on met .2 et .8 en STO 2 et 3 on a les valeurs initiales 'csi'=.05604 'eta'=.0560 et les valeurs finales sont 'csi'=.18341 et 'eta'=.06397 et $\alpha=.2527$ et $\delta=.6939$; B donne $\mu=2.5448$ et quand $\alpha=\delta=.2$ initialement on a 'csi'='eta'=.1032 et en final 'csi'=0.1834 et 'eta'=.06396 il résulte $\alpha=.2527$, $\beta=.6938$ et $\mu=2.5447$.

L'exécution à l'étiquette F donne Rmax=15.3 kg/cm2, Rmin=1.54 et Rf=10.24; V=1473 t et Rmoy=6.63 kg/cm2

Aurel Rottman (289)

```
Programme "HAHN41"

 01*LBL 00

 02*LBL "HAHN41"

 03*LBL A
"POUR MIU=" ARCL 01 AVIEW CLA RCL 01 X^2 4 * 1 + RCL
01 * STO 02 RCL 01 X^2 12 * RCL 01 4 * - 1 + STO 03
RCL 01 12 * 3 - STO 04 RCL 03 X^2 RCL 02 RCL 04 * 4
* - SQRT RCL 03 - 2 / RCL 02 / STO 05 RCL 05
"ALPHA=" ARCL 05 AVIEW RCL 05 2 * RCL 01 1/X + 4 -
RCL 05 RCL 01 * 1 + 4 * CHS RCL 01 1/X + / RCL 05 *
STO 06 "BETA=" ARCL 06 AVIEW RCL 06 6 * RCL 06 X^2
RCL 05 X^2 + RCL 06 RCL 05 * + / STO 07 "LAMDA="
ARCL 07 AVIEW CLA RCL 09 RCL 07 * RCL 15 / RCL 21 /
"Rb=" ARCL X AVIEW STOP


 109*LBL B

SF 12 RCL 02 RCL 03 + RCL 02 RCL 03 * - 6 * RCL 02
RCL 03 * X^2 RCL 02 RCL 03 * 6 * - RCL 02 RCL 03 + 3
* + / STO 08 "MU=" ARCL 08 AVIEW CLA STOP


 143*LBL C

RCL 02 RCL 03 * ENTER^ X^2 X<>Y 2 * CHS RCL 03 + RCL
02 + 3 * + STO 04 3 STO 06 2 STO 07 SF 01


 166*LBL "SIMU"

RCL IND 06 2 * RCL 03 RCL 02 * X^2 2 * - RCL 02 RCL
03 * X^2 RCL IND 07 * + ,25 * RCL 04 / FC? 01 RTN
STO 10 "CSI=" ARCL X AVIEW 2 STO 06 3 STO 07 CF 01
XEQ "SIMU" STO 11 "ETA=" ARCL X AVIEW


 204*LBL D
3 RCL 02 X^2 * RCL 03 X^2 * 4 RCL 02 * RCL 03 X^2 *
- 2 RCL 02 * RCL 03 X^2 * 6 RCL 03 * - 3 + STO 28
RCL 10 * - RCL 04 / ,25 * STO 23 2 RCL 02 3 Y^X *
RCL 03 * 4 RCL 03 * RCL 02 X^2 * - 2 + 2 RCL 03 *
RCL 02 X^2 * 6 RCL 02 * - 3 + STO 29 RCL 10 * - RCL
04 / ,25 * STO 24 2 RCL 02 * RCL 03 3 Y^X * 2 + 4
RCL 02 * RCL 03 X^2 * - RCL 28 RCL 11 * - RCL 04
/ ,25 * STO 25 RCL 03 X^2 RCL 02 X^2 * 3 * 4 RCL 02
X^2 * RCL 03 * - RCL 29 RCL 11 * - RCL 04 / ,25 *
STO 26 RCL 23 * RCL 25 RCL 24 * - STO 27 RCL 20 RCL
10 - RCL 26 * RCL 21 RCL 11 - RCL 24 * - RCL 27 /
STO 18 PRX RCL 20 RCL 10 - RCL 25 * RCL 21 RCL 11 -
RCL 23 * - RCL 27 CHS / STO 19 PRX RCL 20 RCL 10 -
ABS ,0001 X<=Y? GTO 05 RCL 21 RCL 11 - ABS ,0001
X<=Y? GTO 05 STOP


 378*LBL 05
RCL 18 ST+ 02 RCL 19 ST+ 03 GTO C


 384*LBL E
RCL 18 RCL 23 * RCL 19 RCL 24 * + PRX RCL 20 RCL 10
- PRX RCL 18 RCL 25 * RCL 19 RCL 26 * + PRX RCL 21
RCL 11 - PRX STOP


 410*LBL F
RCL 09 RCL 15 / RCL 16 / RCL 11 * STO 04 "Rb=" ARCL
04 AVIEW RCL 02 RCL 03 + RCL 02 RCL 03 * - STO 29 1
RCL 03 - RCL 02 * STO 17 RCL 29 / RCL 04 * STO 10
"Rc=" ARCL 10 AVIEW 1 RCL 02 - RCL 03 * STO 18 RCL
29 / RCL 04 * STO 13 "Rf=" ARCL 13 AVIEW RCL 15 RCL
03 / ST* 17 RCL 16 RCL 02 / ST* 18 RCL 17 RCL 15 +
RCL 18 RCL 16 + * RCL 04 * STO 19 1 RCL 03 - RCL 16
```

```
* RCL 17 * RCL 10 * ST- 19 1 RCL 02 - RCL 15 * RCL
18 * RCL 13 * ST- 19 ,166666 ST* 19 1000 ST/ 19 FIX
0 "V=" ARCL 19 AVIEW RCL 02 RCL 03 * 2 / 1 - CHS RCL
15 * RCL 16 * RCL 09 X<>Y FIX 2 / "Rm=" ARCL X AVIEW
STOP
END
```

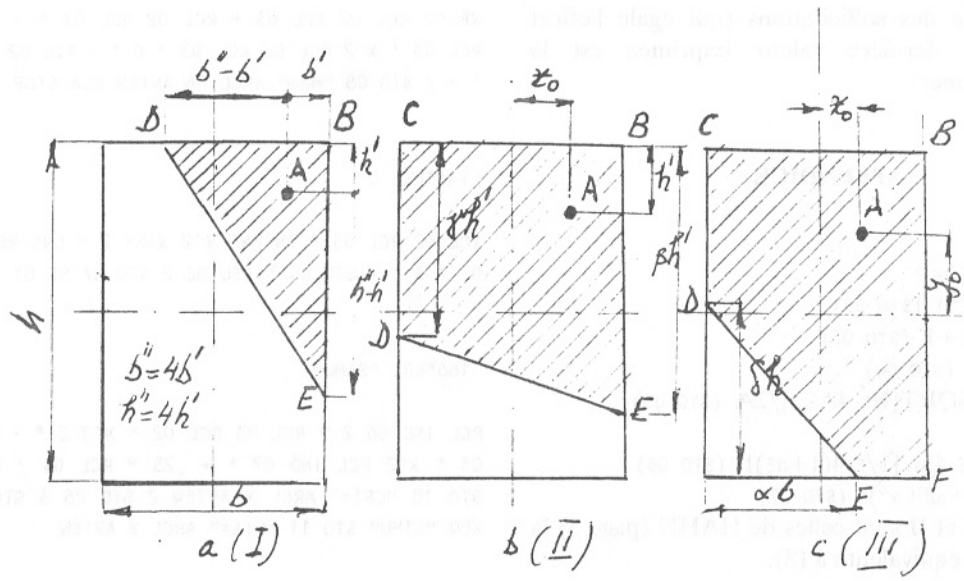


Fig. 1

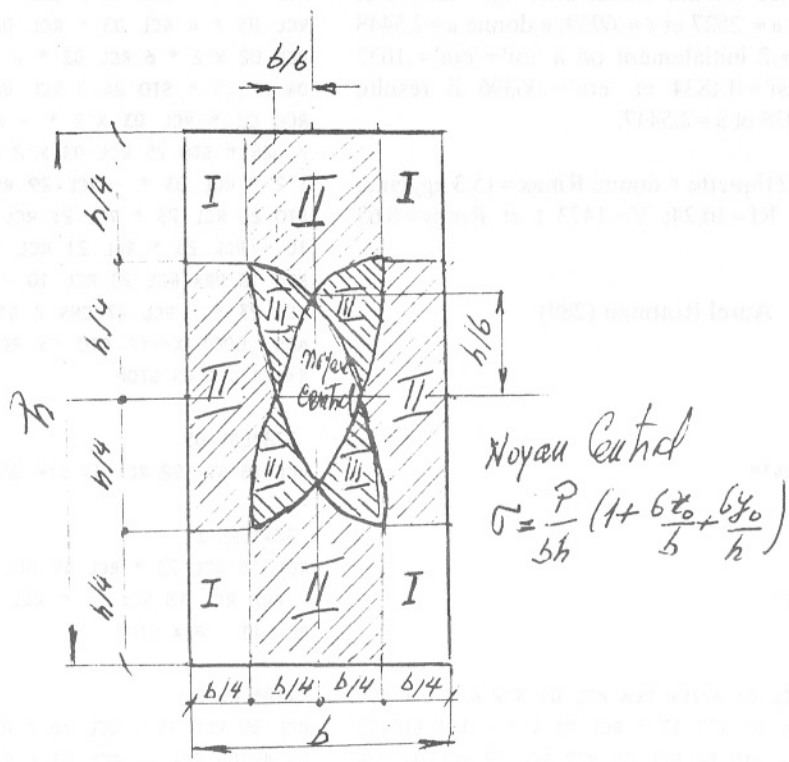


Fig. 2

HP71

MISE EN PARALLELE DE RESISTANCES ET DE REACTANCES

La fonction décrite ci-aprés permet le calcul de 2 éléments en parallèle.

La première version écrite l'était pour deux résistances ou self (ou condensateurs en série). La formule utilisée est :

$$1/R = (1/R1) + (1/R2)$$

Cette partie est implémentée par la routine PARRR ci-dessous;

Par la suite j'ai également écrit la routine pour traiter des valeurs complexes, il s'agit de la routine CMPLX. Cette routine est appelée si le HP-71 détecte que l'un des 2 arguments est complexe.

En effet, dans un environnement, tel le HP-71 avec la mathrom, ou la plupart des opérations peuvent être effectuées aussi bien en réel qu'en complexe, il est intéressant d'étendre cette possibilité aux nouvelles fonctions.

La formule est la même que ci-dessus, avec en plus la formule suivante pour l'inversion d'un nombre complexe :

$$1/(a+jb) = a/(a^2 + b^2) - jb/(a^2 + b^2)$$

Le listing est reproduit ci-aprés; il est bien évident que ce listing est un extrait de celui d'un lex plus long comportant d'autres mots et fonctions; ces derniers ayant tous été décrits par le passé dans le PPC Lausanne (dec87). (NDLR: En effet, ce Lex nous étant parvenu sous forme d'une partie de listing "objet", ne comprenant ni pseudo-op LEX, ni numéro d'ID, nous l'avons représenté dans le format habituel à nos colonnes. D'autre part, ne connaissant pas l'ID attribué au club de Lausanne, nous lui avons attribué le numéro 5E, les lecteurs suisses corrigeront. Nous n'avons pas changé le TOKEN.)

Il est possible et même probable que la même fonction puisse être écrite en moins d'instructions, et je serais heureux de prendre connaissances de remarques en vue de l'amélioration du programme, n'étant moi-même pas particulierement versé dans la programmation en langage machine du HP-71.

Il est malheureusement indispensable de posséder le math pac (mathrom) pour la reconnaissance des variables complexes, et donc l'utilisation de cette fonction. Je me permet d'ailleurs de signaler que cette ROM est trés utile, sinon indispensable a tout possesseur de HP-71 étudiant ou travaillant dans une profession scientifique.

Note : tous les calculs sont effectués sur 15 chiffres, selon les conventions internes du HP-71.

Olivier Pilloud

```
        LEX     'PARRLEX'
        ID      #5E
        MSG     0
        POLL    0

        ENTRY   PARRR
        CHAR    #F
        KEY     'PARR'
        TOKEN   23
        ENDTXT

*
* Cross ref
*
AD2-15  EQU     #0C363
BSERR   EQU     #0939A
DV2-15  EQU     #0C4AC
EXAB1   EQU     #0D3E7  AB <-> scr1
EXAB2   EQU     #0D40E  AB <-> scr2
EXPR    EQU     #0F23C  go on
F1/X15  EQU     #0C33E
MEMCHK  EQU     #012C7
MP2-15  EQU     #0C43A
MPOP2N  EQU     #0BD54  Pop 2 nbs
OUTRES  EQU     #0BC84  Num functs exit
RCCD1   EQU     #0D3F5
RCCD2   EQU     #0D41C  CD <- scratch 2
RCLW1   EQU     #0E981
RCLW2   EQU     #0E9BE
RCLW3   EQU     #0E9C4
RCSCR   EQU     #0E954  pull
SPLITA  EQU     #0C6BF  Split nb in A
SPLTAC  EQU     #0C934  Split  A and C
SQR15   EQU     #0C534
STSCR   EQU     #0E92C  push
STAB1   EQU     #0D3D9
STAB2   EQU     #0D400
STCD2   EQU     #0D427  CD -> scratch 2
XYEX    EQU     #0C697  AB <-> CD
uRES12  EQU     #0C994  15->12
*
*
* PARR function
*
        NIBHEX 8822     Min & max 2 numeric args
```

```
PARRR   GOSBVL MPOP2N
        GOC    CMPLX   Complex PARR
*
        GOSBVL SPLTAC  Do 12->15 on A & C
        GOSBVL STCD2   CD => scratch2
        GOSUB  INV
        GOSUB  exab2   Exch AB with scratch2
        GOSUB  INV
        GOSBVL RCCD2   Recall first arg
        GOSUB  ad2-15  Add
        GOSUB  INV
*
pret    GOVLNG OUTRES  Done
*
INV     GOVLNG F1/X15  Common gates
push    GOVLNG STSCR
stab1   GOVLNG STAB1
rccd1   GOVLNG RCCD1
ad2-15  GOVLNG AD2-15
dv2-15  GOVLNG DV2-15
pull    GOVLNG RCSCR
xyex    GOVLNG XYEX
exab2   GOVLNG EXAB2
*
push15  GOSBVL SPLITA  12->15
        GOTO   push    push AB
*
* CMPLX - Complex PARR
*
CMPLX   R1=C           temp save Re1
*
        CDOEX          Save D0
        CSLC   W       .
        CSLC   W       .
        CSLC   W       .
        CSLC   W       .
        CSLC   W       .
        R4=C           .
*
        GOSUB  push15  push Re2
        A=R1           recall Re1
        GOSUB  push15  push Re1
        A=R0           get Im2
        GOSUB  push15  push Im2
        A=R2           get Im1
        GOSUB  push15  push Im1
*
        GOSBVL RCLW3   get Re1
        GOSUB  AXA
        GOSUB  stab1   Re1^2 is scrtch1
        GOSBVL RCLW1   get Im1
        GOSUB  AXA
        GOSUB  rccd1   recall Re1^2
        GOSUB  ad2-15
        GOSUB  stab1   save sum
*
        GOSBVL RCLW3   get Re1
        GOSUB  dv2-15
        GOSBVL STAB2   Re1/sum in scrtch2
        GOSUB  pull    pop Im1
        GOSUB  xyex
        A=-A-1 S       CHS
        GOSUB  rccd1
        GOSUB  dv2-15
        GOSUB  stab1   Im1/sum in scrtch1
*
        GOSBVL RCLW3   get Re2
        GOSUB  AXA
        GOSUB  push    Re2^2 in top stk
        GOSBVL RCLW2   get Im2
        GOSUB  AXA
        GOSUB  pull    pull Re2^2
        GOSUB  ad2-15
        GOSUB  push    push sum
*
        GOSBVL RCLW2   get Im2
        A=-A-1 S       CHS
        GOSUB  dv2-15
        GOSUB  rccd1   get 1/Im1
        GOSUB  ad2-15
        GOSUB  stab1   save Im
        GOSUB  pull    pull D2
        GOSUB  xyex    temp save sum in AB
        GOSUB  pull    drop Im2
        GOSUB  pull    drop Re1
        GOSUB  pull    drop Re2
        GOSUB  xyex
        GOSUB  dv2-15
        GOSBVL RCCD2   get 1/Re1
        GOSUB  ad2-15
*
        GOSUB  push    Push Re
        GOSUB  AXA     do Re^2
        GOSBVL EXAB1   AB <-> scrtch1
        GOSBVL STAB2   save Im
        GOSUB  AXA
        GOSUB  rccd1
        GOSUB  ad2-15
        GOSUB  stab1
        GOSUB  pull    pull Re
        GOSUB  xyex
        GOSUB  dv2-15
        GOSUB  push
        GOSBVL EXAB2   get Im save Re
        GOSUB  rccd1   get sum
        A=-A-1 S       CHS
        GOSUB  dv2-15
*
cexit   P=     0
        GOSBVL uRES12  15->12
        R0=C
        GOSUB  pull
        GOSUB  xyex
        GOSBVL uRES12  15->12
        R1=C           save Im12
*
```

```
        C=R4            Restore D0
        CSRC    W       .
        CSRC    W       .
        CSRC    W       .
        CSRC    W       .
        CSRC    W       .
        CDOEX           .
*
        CD1EX           Chk if enough stack
        D=C     A       space available
        A=C     A
        C=0     A
        P=      0
        LCHEX   12
        GOSBVL  MEMCHK
        CDEX    A
        D1=C
        GONC    noprob
        C=D     A
        GOVLNG  BSERR
noprob  A=R1
        DAT1=A  W       put Im12
        D1=D1-  16
        A=R0
        DAT1=A  W       put re12
        D1=D1-  2
        LCHEX   0E
        DAT1=C  B       put header
        GOVLNG  EXPR    all done (ouf)
*
*
AXA     C=B     W       square
        CDEX    W
        C=A     W
        GOVLNG  MP2-15
        END
```

INCRUSTATION VIDEO SUR HP-71

Certaines fonctions ou Lex permettent créer des graphiques sur l'affichage du HP-71. Mais si certaines permettent d'agir sur un seul point, alors que d'autres peuvent agir sur la totalité de l'affichage, il devient très laborieux (et très lent!) d'incruster *point par point* une image dans l'affichage. Ceci serait pourtant interressant, si on voulait, par exemple dans un jeu d'action, placer des petits personnages devant un décor.

J'ai donc créé une fonction permettant cette incrustation (ou plutôt *superposition*) en espérant que vous lui trouverez bon usage.

La syntaxe est la suivante :

`GINS` *chaine1* `[,`*position* `[,`*chaine2* `]]`

Les chaines alpha sont du même type que celles necessaires pour l'ordre `GDISP`.

chaine1 est la chaine à incruster (par exemple les personnages cités plus haut).

position est le numéro de la colonne ou incruster *chaine1*. Il doit être compris entre 0 et 131. Si il n'est pas spécifié, il est égal à zéro.

chaine2 est l'autre chaine à incruster (par exemple le décor). par défaut, c'est l'affichage courant.

Par exemple :

```
GINS GDISP$[1,24],12
```

incruste les quatre premiers caracteres de la ligne de commande (l'affichage courant lors du lancement de la fonction) à partir de la 12ème colonne.

Il est à noter que les couples de chaines suivantes sont équivalents :

```
GINS A$,32
GINS A$,32,GDISP$
```

ou

```
GINS A$,0,B$
GINS B$,0,A$
```

Pour accompagner ce lex, je vous propose aussi un petit programme qui, je l'espère, vous donnera quelques idées.

Bon, maintenant, qui sera le premier à nous sortir un *SPACE INVADERS* sur HP-71 ?

Jacques Belin (123)

```
        LEX     'GINSLEX'
        ID      #E1
token   EQU     123
IDBUF   EQU     #E10

        MSG     0
        POLL    0
```

```
       ENTRY  GINSe
       CHAR   #D
       KEY    'GINS'
       TOKEN  token

       ENDTXT

tCOMMA EQU    #F1
BitsOK EQU    1
ANNAD1 EQU    #2E100
ANNAD3 EQU    #2E34C
CPYDD- EQU    #1C43F |
CPY-DD EQU    #1C4C0 | Attention: points d'entrées
CPY-D1 EQU    #1C4C2 | non suportés
DD1ST  EQU    #2E300
DD1END EQU    #2E34C
DD2ST  EQU    #2E200
DD2END EQU    #2E260
DD3ST  EQU    #2E104
DD3END EQU    #2E160
EXPEX- EQU    #0F178
FIXDC  EQU    #05493
FLTDH  EQU    #1B223
FORSTK EQU    #2F59E
I/OALL EQU    #1197D
MFERR* EQU    #093F1
MOVEU3 EQU    #1B177
NEEDSC EQU    #2F94A
NUMCK  EQU    #0369D
NXTSTM EQU    #08A48
POP1N  EQU    #0BD1C
POP1S  EQU    #0BD38
RCLSTA EQU    #01BA0
RESPTR EQU    #03172
SNAPBF EQU    #2F7F0
STOSTA EQU    #01BAB
STRGCK EQU    #036BA
STRNGP EQU    #0379D

GINSd  GOVLNG FIXDC

*Syntaxe : GINS A$ [,N [,B$]]

GINSp  GOSBVL STRGCK   Une chaine obligatoire
       LC(2)  tCOMMA |
       ?A#C   B      |
       GOYES  resptr |  Un nombre optionnel
       GOSBVL NUMCK  |
       LC(2)  tCOMMA |
       ?A=C   B      | Une autre chaine
       GOYES  strngp | optionnelle
resptr GOVLNG RESPTR |
strngp GOVLNG STRNGP

* Registres utilisés : R0: pointeur Math Stack
*                      R1: Longueur chaine Alpha
*                      R2: N=position chaine A$
*                      R3: Addresse buffer

       REL(5) GINSd
       REL(5) GINSp
GINSe  P=     0      |
       AD1EX         |
       R0=A          | Sauvegarde de D0 et D1
       AD0EX         |
       R1=A          |
       LCHEX  00110  |
       B=C    A      |
       LC(3)  IDBUF  |
       P=     0      | Création du buffer
       GOSBVL I/OALL |
       GOC    s1     |
       P=     2      |
       GOVLNG MFERR* |
s1     AD1EX         |  Stockage addresse buffer
       D0=(5) SNAPBF |  dans SNAPBF
       DAT0=A A      |
       A=R0          |
       D1=A          | Recupération D0 et D1
       A=R1          | pour analyse
       D0=A          |
       GOSBVL EXPEX-   Analyse des parametres
       D0=(5) SNAPBF |
       C=DAT0 A      | stockage addresse buffer
       R3=C          |
       C=0    A        | Initialisation de N à 0
       R2=C            |
       LCHEX  0F       | Le dernier parametre
       ?A=C   B        | est il une chaine ?
       GOYES  cas13      oui, on va au cas13

* on est dans le cas  < GINS A$,N >
       GOSBVL POP1N  |
       A=0    S      | on dépile N
       GOSBVL FLTDH  | et on le stocke sous forme
       A=A+A  A      | Hexa apres l'avoir
       R2=A          | multiplié par 2
       D1=D1+ 16       | on passe le paramètre
       AD1EX           | numérique et on stocke
       R0=A            | le pointeur
       C=R3          |
       D1=C          | stockage de l'affichage
       GOSUB  gdisp$ | courant dans le buffer
       A=R0            |
       D1=A            | On dépile A$
       GOSBVL POP1S    |
       R1=A              On stocke sa longueur
       AD1EX         |
       R0=A          | on sauve le pointeur..
       GOTO   disp     et on va plus bas !

* on peut être dans deux cas différents :
* soit GINS A$  soit GINS A$,N,B$
cas13  GOSBVL POP1S    On dépile la chaine
       R1=A            On stocke sa longueur
       CD1EX         | on stocke le pointeur
       R0=C          |
```

```
       A=A+C  A       |
       D1=(5) FORSTK  | ce paramètre touche t-il
       C=DAT1 A       | le fond de la math-stack ?
       ?A#C   A       |
       GOYES  cas3    non, on est dans le cas A$,N,B$

*sinon on est dans le cas GINS A$
       C=R3           |
       D1=C           | on sauve l'affichage courant
       GOSUB  gdisp$  |
       GOTO   disp    et on va plus bas...

* on est dans le cas GINS A$,N,B$
cas3   A=R1           |
       LCHEX  00108   | on force la longueur de
       ?A<=C  A       | la chaine B$ à être
       GOYES  s2      | inferieure à 264 nibbles
       ACEX   A       | (108 hexa)
s2     R1=A           |
       LCHEX  00108   |
       C=C-A  A       | nombre de blancs à droite
       A=R3           | de la chaine à l'affichage
       D1=A           |
       A=0    P       | on remplit le buffer de
loop   DAT1=A 1       | blancs (pour éviter d'avoir
       D1=D1+ 1       | des parasites à droite de
       C=C-1  A       | la chaine B$ à l'affichage)
       GONC   loop    |
       D1=D1- 1       |
       A=R0           |
       D0=A           |
       C=R1           | on place la chaine B$ dans
       A=A+C  A       | le buffer
       R0=A           |
       GOSBVL MOVEU3  |
       A=R0           | on récupere le pointeur
       D1=A           | de la math stack
       GOSBVL POP1N   |
       A=0    S       |
       GOSBVL FLTDH   | on dépile N et on le stocke
       A=A+A  A       | sous forme Hexa
       R2=A           |
       D1=D1+ 16      |
       GOSBVL POP1S   | on dépile A$ et on stocke
       R1=A           | la longueur
       AD1EX          |
       R0=A           | on stocke le pointeur

* tous les cas aboutissent ici.
* on a donc:
* -dans le buffer: soit la chaine B$
*                  soit l'affichage courant
* -dans les registres : R0: l'addr. de la chaine A$
*                           dans la math stack
*                       R1: la longueur de A$
*                       R2: la valeur de N en nibbles
*                       R3: l'adresse du buffer
disp   A=R2           |
       LCHEX  00108   |
       ?A<=C  A       | on force N à être inférieur
       GOYES  s3      | à 264 nibbles
       ACEX   A       |
       R2=A           |
s3     C=R1           |
       A=A+C  A       |
       LCHEX  00108   | on teste si la chaine A$
       C=C-A  A       | ne déborde pas de l'affichage
       GOC    s4      |
       R2=C           |
       GOTO   s5      | non, tout va bien
s4     LCHEX  00108   |
       A=A-C  A       |
       C=R0           | sinon on décale le pointeur
       C=C+A  A       |
       R0=C           |
       C=R1           |
       C=C-A  A       | ..et on réduit la longueur
       R1=C           | de A$
       A=0    A       | R2= offset par rapport à
       R2=A           | l'extremité gauche de l'ecran

s5     A=R0           |
       D1=A           | D1 ^ sur la math stack
       A=R3           |
       C=R2           | D0 ^ sur la position
       A=A+C  A       | desirée de la partie droite
       D0=A           | de la chaine, dans le buffer
       A=R1           |
       A=A-1  A       | initialisation du compteur
       GONC   nz      | de boucle
       A=0    A       |
nz     B=A    A       |
loopd  A=DAT1 1       |
       C=DAT0 1       |
       C=C!A  P       | on effectue un "OU" entre
       DAT0=C 1       | le buffer et la chaine A$,
       D0=D0+ 1       | à la position N, déterminée
       D1=D1+ 1       | par les pointeurs
       B=B-1  A       |
       GONC   loopd   |
       A=R3           |
       D1=A           | et enfin on affiche le
       GOSUB  gdisp2  | résultat
       GOVLNG NXTSTM  .. on s'en va!

* routine de tranfert de l'affichage
* vers un buffer (voir GDISP$ dans  les IDS III)
* Entrée : D1 pointe sur le début du buffer
gdisp$ D0=(5) DD1END
       P=     0
       B=0    A
       GOSBVL CPYDD-
       D0=(5) DD2END
       GOSBVL CPYDD-
       LCHEX  4
```

```
        B=C    P
        DO=(5) DD3END
        GOSBVL CPYDD-
        RTN

* routine de transfert d'un buffer vers
* l'affichage. (voir GDISP dans les IDS III)
* Entrée : D1 pointe sur le début du buffer
gdisp2 CD1EX
        D=C    A
        A=C    A
        LCHEX  00108
        C=C+A  A
        D1=C
        DO=(5) DD3ST
        LC(2)  DD3END
        GOSBVL CPY-DD
        DO=(4) DD2ST
        GOSBVL CPY-D1
        DO=(4) DD1ST
        LC(2)  DD1END
        GOSBVL CPY-DD
        GOSBVL RCLSTA
        ST=1   BitsOK
        GOSBVL STOSTA
        DO=(4) NEEDSC
        LCHEX  E
        DATO=C P
        RTN

        END
```

Programme "PM" (Animation sur écran HP-71, nécessite GINSLEX, JPC Rom ou STRUC2 et REVLEX)

```
 10 DESTROY ALL

 20 DATA 28,62,127,127,119,34,0
 30 DATA 28,62,127,127,119,54,20
 40 DATA 28,62,127,127,127,62,28
 50 DATA 00,00,00,126,000,00,00
 60 DATA 00,00,60,114,060,00,00
 70 DATA 00,60,98,114,122,60,00
 80 DATA 24,36,98,114,122,60,24

 90 DIM A$(6)[7]
100 DIM B$(12)[7]
110 RESTORE 20
120 FOR I=1 TO 3
130    FOR J=1 TO 7
140       READ N
150       A$(I)=A$(I)&CHR$(N)
160    NEXT J
170 NEXT I
180 A$(4)=A$(3) @ A$(5)=A$(2)
190 FOR I=1 TO 4
200    FOR J=1 TO 7
210       READ N
220       B$(I)=B$(I)&CHR$(N)
230    NEXT J
240 NEXT I
250 B$(5)=B$(3) @ B$(6)=B$(2) @ B$(7)=B$(1)
260 FOR I=2 TO 6 @ B$(I+6)=REV$(B$(I)) @ NEXT I
270 DIM D$[132] @ D$=" "
280 E$=CHR$(92)&CHR$(95)&"/"
290 FOR I=1 TO 7
300    D$=D$&E$
310 NEXT I
320 DISP D$ @ D$=GDISP$
330 LOOP
340    FOR I=1 TO 130
350       IF KEYDOWN THEN LEAVE
360       GINS A$(MOD(I,5)+1),I,D$
370       GINS B$(MOD(I,12)+1),MOD(I/2+70,132)
380    NEXT I
390    FOR I=131 TO 0 STEP -3
400       IF KEYDOWN THEN LEAVE
410       GINS REV$(A$(MOD(I,5)+1)),I,D$
420       GINS B$(MOD(INT(I/3),12)+1),3
430    NEXT I
440 END LOOP
```

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 trouverez donc le Lex CHARLEX nécessaire à la rédaction de votre article (voir "Ah ! Vous écrivez !"), les nouvelles fonctions, ainsi que les Lex utilisés dans les programmes Basic de ce mois-ci.

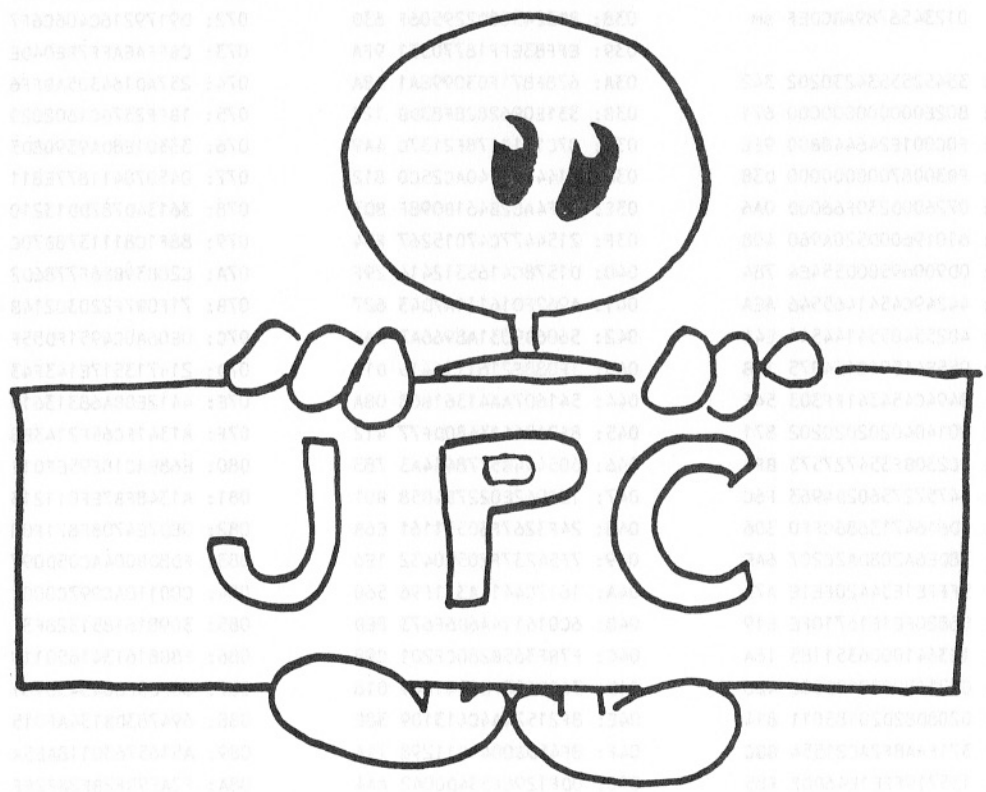
```
CHARLEX

PARRLEX   PARR        XFN    094023

GINSLEX   GINS      XWORD    225123

REVLEX    REV$        XFN    082012

STRUC2    END       XWORD    225066        WHILE     XWORD    225067
          REPEAT    XWORD    225068        UNTIL     XWORD    225069
          LEAVE     XWORD    225070        LOOP      XWORD    225096
          SELECT    XWORD    225097        CASE      XWORD    225098
          IF        XWORD    225099        ELSE      XWORD    225100
```

```
 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         624 octets

      0123456789ABCDEF sm

000: 34841425C4548502 35E
001: 802E000000000000 68D
002: 5E4001EFF0000000 9FD
003: FE0000000800001F D57
004: F31BF961400032BF 0EA
005: 38F14A11DB10AD23 484
006: 07D532BFB8FD7911 837
007: 11AD754D7A101743 BBA
008: 11014D1CB15D0000 F25
009: 71450375FF864834 2A2
00A: 5655581008355654 5F9
00B: 5810070507701724 93F
00C: 7700775070077517 C92
00D: 2077040708364545 FE0
00E: 4A30000A49724000 333
00F: 0808094A2C180814 69C
010: A464242008355455 9F6
011: 581000054C714000 D3C
012: 0C3142404C700832 098
013: 41414A70002078A0 3F0
014: 2F30000000000000 71B
015: 0000000000000000 A2B
016: 0000000000000000 D3B
017: 0000000000000000 04B
018: 0000000000000000 35B
019: 0000000000000000 66B
01A: 0000000000000000 97B
01B: 0000000000000000 C8B
01C: 0000000000000000 F9B
01D: 0000000000000000 2AB
01E: 0000000000000000 5BB
01F: 0000000000000000 8CB
020: 0000000000000000 BDB
021: 000000000000080C F06
022: 1A28080008080A2C 270
023: 180008040E340800 5B9
024: 08001E3018000000 8F3
025: 0000000000000000 C03
026: 0000000000000000 F13
027: 0000000000000000 223
028: 0201000000010200 539
029: 0000000201020000 84E
02A: 0001000100000002 B62
02B: 0102010000000000 E76
02C: 0000000000000000 186
02D: 045E755142400101 4D2
02E: 0101010000000000 7E5
02F: 0000000000000000 AF5
030: 0000070507000000 E18
031: 00000000083444C4 156
032: 44400D7901112D70 4B6
033: 050D750509700000 800
034: 0D70000000384540 B43
035: 4020014E322E3140 E97
036: 084E794142400000 1E7
037: 00000000002E4559 525
038: 3200000000000000 83A
039: 0000000000000026 B52
03A: 5556587008365556 EB1
03B: 5810083645464830 202
03C: 0832414248700024 543
03D: 5655587008345655 8A0
03E: 5810083446454830 BEF
03F: 0C3042414C700024 F44
040: 5556587008355654 2A1
041: 5810083546444830 5F0
042: 0C3142404C700025 946
043: 5455587008355455 CA0
044: 5810083544454830 FEE
045: 0C3140414C700875 350
046: 14141870000A4972 6A1
047: 40000E3159454E30 A01
048: 0C7A0F7949400024 D79
049: 5554587000084A71 0D5
04A: 40000C523A262D10 436
04B: 0424587458400875 78D
04C: 1415187000094A70 ADD
04D: 4000083544454830 E21
04E: 0C3140414C300C74 189
04F: 5655545000054C71 4E0
050: 40000            5D9

PARRLEX         314 octets

      0123456789ABCDEF sm

000: 05142525C4548502 357
001: 802E000000000000 686
002: 87200E5717100000 9D1
003: F710000000000000 CFF
004: 081000F705142525 04D
005: 711FF88228F45DB0 3F3
006: 4C78F439C08F724D 7A1
007: 07E10725076108FC B1B
008: 14D0772077008D48 E8A
009: CB08DE33C08DC29E 258
00A: 08D9D3D08D5F3D08 60E
00B: D363C08DCA4C08D4 9C8
00C: 59E08D796C08DE04 D75
00D: D08FFB6C060CF109 131
00E: 1368128128128128 47F
00F: 1210C79DF11172DF 816
010: 1107BCF11274CF8F BC0
011: 4C9E07F61748F8F1 F71
012: 89E070617C7F7F7F 326
013: 7D6F8F4C9E0777F8 6F3
014: F004D0737F767FBC AA7
015: C735F7D5F744F8F4 E6C
016: C9E070217E2F8FEB 230
017: 9E07111724F703F7 5B2
018: 71F8FEB9E0BCC752 989
019: F731F761F740F7C1 D30
01A: F7F1F741F701F7C0 0E4
01B: F7F0F7DFE8FC14D0 4CD
01C: 7BEE72DE7CB08F7E 8B8
01D: 3D08F004D07AA076 C4B
01E: CE79CE77BE7FCE72 03E
01F: DE70CE70AE8FE04D 41E
020: 073AEBCC7AAE208F 7F4
021: 499C010871AE74AE B95
022: 8F499C010911C816 F1A
023: 8168168168161361 271
024: 37D7DAD22031218F 608
025: 7C210DF1355B0DB8 9AD
026: DA939011115171CF D32
027: 11015171C131E014 085
028: D8DC32F0AF9AFFAF 487
029: 68DA34C0F        6A3

GINSLEX         375 octets

      0123456789ABCDEF sm

000: 7494E435C4548502 378
001: 802E000000000000 6A7
002: 2F2001EB7B700000 A19
003: F710000000000000 D47
004: 035000D77494E435 0B3
005: B71FF8D394508FAB 47B
006: 630311F966218FD9 802
007: 630311F962908D27 B75
008: 1308DD9730BCFFFD F47
009: CFFF201331001330 2BD
00A: 78F7A51034022000 62F
00B: 532EEE208FD79114 9D2
00C: B0228D1F39013310 D3F
00D: 1110133203400000 062
00E: 1357EA17D021FEF3 40B
00F: E215B01003410080 75B
010: DAF4F4F41590CECE B3D
011: 5FE1101590AF01F0 ECC
012: 01E215931FC43E21 24F
013: 5931FF86008FE0C1 5F2
014: 07FB1762005F4943 970
015: 535F4E4024472146 CDE
016: 52594C412D0A0FF0 072
017: 71358FE0C1070208 3EA
018: BCBDBEFDDCFCF8F8 822
019: F8F0F0767C3C1600 BC0
01A: 71081113480100CA F17
01B: 10211913572A0340 25F
01C: E00010B111130112 5A2
01D: 13134801008F771B 906
01E: 111311AC21341181 C57
01F: 351527157 70E7E15 FC9
020: 0717F16F15271577 33E
021: 0E7E150711213173 69F
022: 703400800CE5DF11 A1E
023: 3A6CA6C103310296 D9E
024: 66832EEE8F14A118 144
```

```
025: F17510068D84A801 4C0
026: BC43E220D18FF34C 878
027: 11B062E28FF34C13 C0D
028: 04A851B061E28FF3 FA7
029: 4C101137D7DA3480 32A
02A: 100C21351B401E23 68B
02B: 1068F0C4C11A002E A14
02C: 8F2C4C11A003E31C DB0
02D: 48F0C4C18F0AB108 157
02E: 518FBAB101AA49F3 50A
02F: 0E1540018F2C6004 877
030: 8F013400002CE5DF C06
031: 01F              CB0

REVLEX          33 octets

     0123456789ABCDEF sm

000: 255465C454850202 35C
001: 802E000000000000 68B
002: 6400025C0C000000 9D2
003: F710000000000000 D00
004: 071000F725546542 056
005: C01FF4118FE83B18 405
006: DC32F0F          5B4

STRUC2          1541 octets

     0123456789ABCDEF sm

000: 3545255534230202 342
001: 802E000000000000 671
002: F0C001E246448B00 9EE
003: FB30087000000000 D38
004: 072600D230F6600D 0A6
005: 61019600D520A960 408
006: 0D90069500D554E4 784
007: 44249C4541465546 AEA
008: 4B25540554144544 E44
009: 955E44594C454975 1CB
00A: 8494C454341FF303 54E
00B: 0014040202020202 871
00C: CC230BF354727573 BFF
00D: 64757275602D4963 F6C
00E: 7D6164713686CFF0 306
00F: 38DE6A208DA2C207 6AE
010: 5FFFE1E34420FE1E A7E
011: 06820FE1E16710FE E19
012: 1E36410006351185 16A
013: 0121650022185800 4B6
014: 0208D82D20185011 814
015: 371FAA8F2AC21554 BBC
016: 135719FFE1E46DDF F85
017: 007E408FB394031A 30A
018: 8966A18F8EC20AA6 6C1
019: BB6BB679AF6B1072 A7E
01A: 207430313F966D08 DEC
01B: F8EC2070208FEA23 196
01C: 044B8D271308F9DF 53A
01D: 30870935CE8F9DF3 8E9
01E: 0870A230187340E6 C50
01F: 8161371FAA8F2153 FDC
020: 4948111359420080 33B
021: 53E201554135018F 6A1
022: D96307B9F7BCE4F9 A73
023: 000068401818FB39 DDB
024: 40310F9628131049 134
025: 62F03112962606F1 49D
026: 0626F1371F088F21 81C
027: 57413594E40018D5 B8D
028: 30308D0745014B7B EFF
029: D4474D61708F6242 284
02A: 0510220300339464 5BE
02B: 23653037C4F4F405 93B
02C: 2765203B3554C454 CA7
02D: 34452B6110397584 007
02E: 94C4542974914B8 382
02F: D303507D80390245 6EB
030: 8454E4296FDF7D54 A9E
031: 5213754C43554276 E00
032: 8CF153331A896631 183
033: 8F0E160754017252 4EE
034: 27240313F966C137 85D
035: 0245F40227722017 BB0
036: 114B7F108FCE2503 F46
037: 1C221770017166AF 2B9
038: 8D324508D229506F 63D
039: EFF83EFF18770351 9FA
03A: 678F871F030998A1 D8A
03B: 331E0962828F83DB 120
03C: 07C151B178F2137C 4A9
03D: 2144164140AC25C0 812
03E: 7DF4AC2B461B098F BD3
03F: 2154477C47015267 F34
040: D1578C4165312414 29F
041: A962F016114A7D43 627
042: 5606D9031A8966A3 9AB
043: 1F088F2161156415 D18
044: 541607AA41361B08 08A
045: 8F2156413480DF77 412
046: 605444B5778414A3 783
047: 13F962E0227B4058 B01
048: 24F3267F30551161 E6B
049: 7F54237F205C0432 1E6
04A: 1617C4414A311F96 560
04B: 6C016114A6D6F673 8ED
04C: F7BF365B280CF201 C98
04D: 361B698F21441B09 018
04E: 8F21524A4C413109 38E
04F: 8F45DB0044B11298 71A
050: 0DF1298F534D0042 AA4
051: 065B08D51DB0AC78 E4C
052: F83DB0137C21351B 1DF
053: 678F2146841840AC 564
054: 08AA218A8868A601 8F0
055: 8505418A8C25C38B C82
056: 670851D618414213 FDF
057: 08F7C1B120AC0482 36F
058: 860013028160E4E6 6D6
059: B20871B13018160E A45
05A: 4E681014A14F9E29 DD7
05B: E3048160E4EACB0E 18B
05C: 4694C201B698F214 50E
05D: 61340163DFF47BFF 8BF
05E: 187762328733360D8 C2D
05F: 1EFCFF9FAFF70D28 029
060: FE3F804033098169 3B0
061: 4752DB13416314A8 728
062: F3E3203471006010 A81
063: 449000066210C0C8 DDF
064: 0C020A87309A0B80 15B
065: D07A6276C26021D2 4E0
066: CFFC9AFF14A7EF04 8D4
067: D0300902725D1313 C30
068: 479E17F915E07032 FB7
069: 7CB16FD08D84A803 368
06A: 10677C175A168C05 6E5
06B: DBFFD3AFF7212776 ABB
06C: 15E078F174816BCF E61
06D: 207A4264A051CFF0 1F1
06E: 1AFF78616FAF89BF 5D1
06F: F00AFF1361B698F2 980
070: 1443144736611B698 CE0
071: F21461347C014E07 056
072: D9179216C406C6F7 3F0
073: CBFFAEAFF7BE04DE 80F
074: 257AD164305ABFF6 BBA
075: 1BFF2376C1602020 F39
076: 33301E8DA93908D5 2CB
077: 045078411877E811 62E
078: 36134D787DD13210 9A8
079: 88F1C8111378B70C D3B
07A: C28B39B86F7786D2 0EE
07B: 71F0B7F22030214B 46B
07C: 0E06A0C4951FD55F 817
07D: 214713517E143F43 B80
07E: 4412E08A68313610 EE5
07F: A1341FC65F2143E6 27E
080: E68BAC18F95EF011 641
081: A1348FB7EF011213 9D4
082: 08D7E4708F871F08 D73
083: FD8DB004AC05D097 12C
084: C00110AC097C0001 499
085: 3098161851328F31 7FA
086: F808161341650110 B4D
087: 88FE3F8043430981 ED9
088: 694783DB134AF015 269
089: A51657630118AE5A 5F1
08A: F2AE9BF2BF2BF2BF 9EE
08B: 233FE1E9720061AE D86
08C: 1B198F2146134011 0E8
08D: 361B198F21441340 44F
08E: 11371F1C6F214513 7C2
08F: 5011F1C6F21471E9 B4C
```

```
090: 95F1458D681F0D01 EE0
091: 4A136C213401614E 24D
092: 80CF208F5341080D 5D7
093: F80C0201F088F215 95F
094: 501F765F214787D3 CE7
095: 1321088F1C811137 04D
096: C2D731FE8F99A805 40A
097: CA795F16114A311E 7A1
098: 962606DA016114A3 B14
099: 124962023F340644 E6C
09A: 16365426462F8F89 1EE
09B: 0B14F78F3E32034F 58A
09C: 5006850441503664 8D3
09D: 016B30001F088F21 C3F
09E: 5708F62420850811 F9E
09F: 760E01D01D60F707 324
0A0: F05D0A8058008028 69B
0A1: 6D002767000C0C80 A09
0A2: CF2070211F088F21 D8E
0A3: 554798E1817FCE63 131
0A4: 0F7A7E16514A7D1D 4DD
0A5: 51160D031547FD04 850
0A6: 716BFC314670D048 BDE
0A7: 02261906CA031267 F3F
0A8: BB043F215C70C0C8 2E0
0A9: 0CF20AC714A33462 66E
0AA: 6238F890B15C3312 9EC
0AB: 4966911611520301 D33
0AC: 90242302902C1ACB 0AE
0AD: 8126F0F31267F504 42D
0AE: 702150275DD1817B 7A5
0AF: 1E6F4E3146704048 B25
0B0: E0C0C80CF2071BD1 ED2
0B1: 6514A745C4028129 240
0B2: 0681789D18172504 5A7
0B3: 0077DD6B0E632C14 93D
0B4: A96200011F495F21 CB2
0B5: 43E41418F156818B 035
0B6: E911F088F2157013 3B4
0B7: 11C01550018DD449 726
0B8: 01FF85F214717414 AA5
0B9: 38A200CC14113115 E0A
0BA: 701F088F21550011 16D
0BB: E064600000FB3000 4CD
0BC: 00000000D1091BFF 83A
0BD: D8204C7FFD00030A BDA
0BE: FFD61035BFF9B005 F97
0BF: 6BFFD73414355426 32B
0C0: 754C435544639464 697
0C1: 367C4F4F40506B35 A27
0C2: 54C4543445161FF  D73
```



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

La maquette de ce numéro a été préparée et réalisée par Jacques Belin grâce à un système comprenant un HP71B, un lecteur de disquettes HP9114A, un HP9807A, deux HP9154 et une imprimante LaserJet Séries II.

Les dessins sont de Jean-Jacques Dhénin et Paul Courbis.

Directeur de la publication : Jean Reibel
Numéro ISSN : 0762 - 381X

Veuillez adresser toute correspondance à :
PPC Paris, BP 604, 75028 Paris Cedex 01.

Imprimé par Copy-Express, 42 86 91 94.

ENGLISH SUMMARY

JPC 63 - APRIL 89

During the *Consumer Electronic Show*, in Chicago, in USA, the June 3 1989, a meeting will be made for the tenth birthday of the HP-41, the 75 years of Hewlett Packard, and the 13 years of *Chips*. Unfortunately, nobody, from Paris, can go to this fest. Too bad!, everybody thinks that if HP wants to do revelations, it's will be this day ! Perhaps an American reader...

This issue of *JPC* begins with a letter of C. Dupré, who made sun-dials and wants to know if somebody have programmed his HP-41 for this kind of job. He searchs also MLE1H Eproms for a MLDL.

The HP-28 section opens with a review of the book written by S. Lalande and P. Courbis (two members of PPC-Paris) "Journey in the center of the HP-28C/S". This book shows us the internal running of this machine, with some programs, etc... After that, P. Heilbronn give us a fifth version of his program of constant factors. Next, J. F. Garnier brings some news functions date and time for the HP-28. Finally, comming from Sweeden by the Unix Network, and written by O. Gallmo, an impressive Lisp compiler, for 28S only (it's too big!).

One program for HP-41 this month, but very pro ! It is a program of civil ingineering, for the determination of constraints in a foundation subject of excentrics loads.

Two Lex in the HP-71 section. the first computes the values of parallels resistors with complex variables, but it needs the MathRom. The second Lex works with the display, for inlay a picture into another. You can see a little demo of the possibilities of this function, whith the program "PM", at the end of the section.

Until next month and Happy Programming !