JPC

PPC-PC

NUMERO 17
Septembre 1984

Le numéro: 25 FF.

DES PROGRAMMES ...

Edité par PPC-PC

EDITORIAL

Chers amis,

Voici arrivé septembre avec son cortège de feuilles mortes; néanmoins, celles que vous lisez à présent sont loin d'être dans cet état, jugez vous-mêmes...

La nouvelle présentation de la rubrique HP 71 vous donnera un petit apercu de ce que sera bientôt votre Journal préferé; c'est encore une formule expérimentale et nous attendons de pied ferme vos critiques et suggestions sur ce point pour parachever notre oeuvre. Pour plus de détails, reportez-vous à l'article de Frederic; n'oubliez cependant pas qu'il n'est nullement impératif que vous nous envoyiez toutes vos idées géniales sous cette forme, mais ce serait tellement mieux pour nous tous, et pour les autres...

Utilisateurs de HP 41, 75, 16, 15, 12, etc..., êtes-vous relégués au fin fond des Ténebres Exterieures pour que nous n'ayons plus de vos nouvelles ? Il ne tient qu'à vous que vos rubriques soient plus etoffées.

Pour conclure, n'oubliez pas que pour notre Assemblée Générale annuelle, prévue en octobre, nous sommes à la recherche d'une salle pour nous réunir un dimanche, notre salle mensuelle n'étant disponible que le samedi. Avis à faire circuler !

Pour le Bureau, Olivier ARBEY

P.S. Cet éditorial a été rédigé avec l'aide du MemoMaker du HP 110... De plus amples informations dans notre prochain numero !

DIVERS

NOUS EN AVONS

Sont en vente au Club:

- Cartes magnétiques: 150 FF les 50 cartes
- Eproms (2716 et 2732) :
  Vierges: 55 FF + 3,10 FF de port
  Programmées: nous consulter
- Convertisseurs HP-IL (référence HP 82166A): 1650 FF + 3,10 FF de port. Livrés avec connecteurs et manuel.
- Il reste un module HP-IL en promotion (pour HP 41): 760 FF + 3,10 de port.
- Le MLDL Hollandais (société ERAMCO) : 3860 FF + port
  24 K Rom, 8 K Ram de microcodes. Délai de 1 à 2 mois.
- Commandes de la ZENROM et de la CCD ROM: prix de souscription: 1000 FF, envoyez nous 500 FF minimum d'arrhes.
- VASM (listings commentés par HP des 3 premières pages microcodes de la HP 41): 150 FF + port.
- Manuel de service de la HP 41: 50 FF + port
- Manuel technique du convertisseur: 20 FF + port

ACHATS GROUPES

Nous envisageons de grouper les achats de ceux qui le désirent, afin d'avoir des réductions sur les matériels HP et autres.
En effet, il vaut mieux être 5 pour demander une réduction que seul. Nous aurons ainsi plus de poids face à certains magazins. C'est aussi cela, le Club !
Faîtes nous connaître vos intentions d'achat. Nous ferons tout ce qui est possible pour négocier, mais il faut que nous ayons des promesses fermes...
Pour l'instant, il y a 3 personnes qui se sont décidées pour le module Forth/Assembleur du HP 71. A vous de jouer ...

COURRIER DU COEUR

Saunier Stéphane revend sa souscription à la CCD ROM car il est obligé de se séparer de son système 41. Cette souscription a été effectuée au SICOB pour la somme de 885 FF. Si cela interesse quelqu'un:
SAUNIER Stéphane
11 allée de la Source
78480 Verneuil/Seine

A propos de JPC.

Avec l'arrivée du 71, de nouvelles possibilités s'offrent à nous pour la présentation des articles. Jusqu'a maintenant, nous avons publié les documents que vous nous envoyez en les photocopiant. Il est maintenant possible d'adopter un format standard, et, surtout d'abandonner l'idée selon laquelle un texte d'article commence en haut d'une page et se termine en bas.

Il vous suffit, pour vous en convaincre, de parcourir la rubrique 71; vous constaterez qu'une partie des articles est enchainée, imprimée sur 3 colonnes, les listages étant séparés et structurés automatiquement. Pour aboutir à cette présentation il suffit que tous les articles nous arrivent sous forme magnétique, c'est à dire que vous les tapiez dans votre 71 sans bien sur vous occuper du format. L'opération n'est pas plus difficile que celle qui consiste à écrire un article à la machine, et autorise les corrections.

Si nous recevons les deux fichiers (programme et texte) sous cette forme, il nous suffit de quelques minutes pour traiter et enchainer les articles, et de quelques autres pour le listage. Si vous ne pouvez vous séparer d'une cassette même pendant quelques jours, sachez qu'il sera toujours possible d'en faire une copie lors de nos réunions mensuelles. D'autre part, je suis tout à fait conscient que relativement peu de gens possèdent déjà l'interface HP IL, le lecteur de cassettes, et une imprimante.

Si vous n'avez pas de lecteur, vous pourrez utiliser cette méthode en tapant votre article la veille d'une réunion, vous y trouverez tout le matériel nécéssaire pour nous fournir votre oeuvre.

D'autre part, beaucoup de membres ne peuvent nous envoyer d'articles du fait qu'ils n'ont pas d'imprimante ou de machine à écrire, et donc reculent devant les difficultés à surmonter pour nous fournir un travail propre. Grace à cette méthode, l'obstacle est surmonté, et je pense que cela devrait être un grand pas vers l'élargissement nécéssaire du cercle des auteurs.

L'utilisation du programme de mise au format que nous avons réalisé permettrait donc de gagner en place et en lisibilité. Il est d'autre part entièrement compatible avec les jeux de caractères secondaires créés grace au programme de Michel Martinet que vous trouverez dans ce numéro.

Mais, vous demandez vous, comment taper votre article? Il suffit de taper le texte sous la forme d'un programme basic ne contenant que des remarques (en utilisant le symbole ! au lieu de rem), le texte étant agrémenté de quelques commandes telles que: aller à la ligne, paragraphe suivant, sauter des lignes, tirer un trait. Une commande se compose du caractère ^ suivi de la commande elle même. Une commande doit être la seule chose contenue sur une ligne. Pour indiquer d'aller à la ligne, vous entrez donc une ligne contenant ! ^L, pour aller au paragraphe suivant , la ligne sera ! ^P. Pour sauter des lignes, la commande est S, ce qui donne, pour 3 lignes: ! ^S3. Pour tirer un trait: tapez la ligne ! ^T.

C'est, vous le voyez, très simple. Si vous désirez des renseignements supplémentaires, passez moi un coup de fil au (1) 680 07 34.

Vous remarquerez que les listages structurés sont agrémentés de commentaires. Ceux ci sont simplement entrés dans le corps du programme sous forme de remarques. Au moment du listage, leurs numéros de lignes sont purement et simplement supprimés. Cette méthode vous évite, si vous voulez documenter vos programmes, d'utiliser les classiques lignes de remarques ne contenant que des pointillés, des lignes de séparations sont en effet insérées avant chaque fonction utilisateur, chaque étiquette, chaque sous programme. Le listage est alors clair, mais le programme ne prend pas une place inconsidérée en mémoire.

Je ne vous fais bien sur là qu'une proposition. Vous n'êtes pas du tout tenu de me suivre. Tous les articles tapés à la machine restent les bienvenus. Le but de la chose est simplement de vous permettre d'écrire des articles, même si vous n'avez qu'un 71. Je trouve également agréable de faire participer nos machines à la rédaction de nos articles. Mais n'oubliez pas, ce sont vos articles qui décideront en dernier ressort si cette méthode est appelée à vivre.

J'espère bientôt vous lire.

Frédéric POUPON (#35).

QUOI DE NEUF SUR LA BOUCLE ?

Un point sur les futurs périphériques HP-IL est indispensable, pour discerner les rumeurs des certitudes.
Trois nouveaux appareils HP-IL sont annoncés, plus ou moins officiellement: l'imprimante à Jet d'Encre ("Think Jet"), un lecteur de disquettes, et une nouvelle interface vidéo 80 colonnes.

Vous connaissez déjà bien "Think Jet", puisqu'elle est annoncée officiellement, qu'elle était en démonstration au Sicob de Printemps, et qu'elle a fait l'objet d'une présentation dans les colonnes de JPC. Selon les dernières informations, son prix ne dépasserait pas 6000 FF TTC, en France. Il est actuellement de 430 $ (prix Educalc).

Le lecteur de disquette est une "rumeur certaine". D'ici la fin de l'année, nous disposerons pour environ 7000 à 8000 FF d'une unité de disquette 3½ pouces, (format HP-Sony), totalisant une capacité de stockage d'à peu près 600 Ko formattée. Certaines mauvaises langues pensent que c'est une version modifiée du lecteur de disquette qu'avait conçu Frédéric Poupon, et dont il nous avait parlé dans le numéro d'avril 84 ! Dernière chose: la référence serait HP 9114, selon des sources bien informées ...

Quant à l'interface vidéo 80 colonnes, il semblerait qu'elle soit fabriquée par une firme danoise. Justement, nous avons reçu des informations sur une firme danoise qui fabrique une interface vidéo. Quel hasard ! Ses caractéristiques sont annoncées, ce qui me permet de vous en faire part:
Elle est (paraît-il) optimisée en ce qui concerne la vitesse d'affichage. Le jeu de caractère Ascii est étendu avec 28 caractères spéciaux (symboles grecs). Les jeux de caractères nationaux sont en option. La vidéo inversé est implémentée. Du coté séquences d'échappements, je ne veux pas paraître difficile, mais l'interface vidéo fabriquée par Mountain Computers, commercialisée par La Règle A Calcul en possède beaucoup plus ... Espérons que cette interface aura toutes les capacités HP-IL. Dernière chose: la firme danoise s'appelle MICRONIX.

D'autre part, Educalc (un gros magasin américain) annonce dans son catalogue un lecteur de disquette 180 Ko 5 pouces un quart. Compatible IBM. On ne rit pas !
Plus interessant est l'écran à cristaux liquides de 25 lignes de 80 caractères, qui coutera aux alentours de 500 $. Date non annoncée... Il se branche sur l'HPIL, et mesure 23x14 cm.

Nous attendons avec impatience ces nouveaux matériels pour un essai dans JPC. Nous vous tiendrons informés dès que possible.

Pierre DAVID

PS: Pour les heureux possesseurs de l'interface vidéo Mountain Computer, Michel Martinet a modifié le jeu de caractères, et a implanté notamment un jeu semi-graphique très complet. Passez à La Règle A Calcul demander une démonstration, ce n'est pas tous les jours que l'on peut voir une interface vidéo HP-IL faire du graphique !!!

des nouvelles de ccd-rom

Cher Philippe.

Voici les renseignements désirés sur le module CCD !

Le module CCD sera une ROM micro-code de 8k. Il contiendra quatre blocs de fonctions principaux qui seront décrits ci-dessous :

- Extensions du système,
- Utilitaires.
- Fonctions mathématiques.
- Fonctions d'Entrée / Sortie.

Voici une brève description des fonctions et de la philosophie qui les supporte :

Extensions du Système

Les fonctions système suivantes ont été rendues plus puissantes : CAT, ASN et XEQ.

CAT : En plus des anciens catalogues 1 et 3, nous avons les nouveaux catalogues :

CAT 0 : c'est un catalogue de boucle qui liste tous les appareils de la boucle en affichant leur adresse et leur chaine d'identification. (Si l'appareil ne peut pas envoyer un ID, on affiche alors AID ou ???). L'appareil primaire est repéré par un astérisque. Lorsque l'on arrête le catalogue, on peut réinitialiser un appareil (Selected Device Clear) en pressant la touche "C", ou le rendre appareil primaire en appuyant sur la touche ENTER.

CAT 2 : Comme le CAT 2 normal, mais avec les "fonctions étendues" :

- R/S : Déroule le catalogue en avant,

- Shift R/S : Le déroule en arrière,

- ENTER bascule entre le catalogue des titres de ROM et le catalogue des fonctions (comme sur la 41CX). L'appui sur ENTER, si le catalogue est en mode fonction, affichera le titre de la ROM courante, et non le suivant comme le fait la 41CX.

- La touche "A" permet d'affecter la fonction affichée à une touche, directement à partir du catalogue.

- XEQ : Permet d'exécuter une fonction, directement a partir du catalogue.

CAT 4 : Est analogue à EMDIR, mais, en plus, affichera correctement les types de fichier additionnels (voir ci-dessous).

CAT 5 : C'est l'ALMCAT du module TIME. Il affichera "NO TIMER" si un tel module n'est pas enfiché.

CAT 6 : Analogue au CAT 6 de la 41CX, mais avec quelques différences importantes :

Toutes les affectations de fonctions synthétiques sont affichées correctement (c'est-à-dire comme affichées en tant que ligne de programme).

Le nom de la fonction, suivi du code touche, est justifié à droite de l'afficheur, et non à gauche. Voici un exemple :

```
"     TONE 11"
"    RCL M 12"
```

La raison en est qu'il est beaucoup plus facile pour l'œil humain de suivre ce catalogue que celui de la 41CX.

CAT 7 : C'est le DIR du module IL. Si ce module n'est pas present, l'afficheur indiquera :

"NO HPIL"

CAT 8-F : Ces catalogues sont des entrées spéciales dans CAT 2. Ils commencent à lister les fonctions à la ROM située à la page indiquée. Si la page correspondante est vide, l'afficheur indiquera :

"NO ROM"

XEQ et ASN permettent une entrée directe de codes fonction à deux octets, soit en hexadecimal, soit en decimal. On peut en outre entrer des numéros de XROM, apres avoir pressé à nouveau la touche XEQ.

Les extensions ASN, CAT et XEQ sont présentes dès que le module CCD est enfiché. Il n'est pas nécessaire de les affecter à des touches spéciales.

MODE MINUSCULES : Ce nouveau mode est mis en service en exécutant LCON. LCOFF le met hors service. Lorsque ce mode est en service, on peut introduire au clavier n'importe quelle chaine de caractères dans le registre alpha ou dans une ligne programme de texte. En mode alpha, on dispose de quatre claviers : NORM, Shift NORM, USER et Shift USER.

- USER , est le clavier ALPHA bien connu avec des caractères majuscules,

- Shift USER correspond à l'ancien clavier ALPHA Shifté,

- NORM donne tous les caractères minuscules plus certains caractères spéciaux,

- Shift NORM donne de nombreux caractères spéciaux tels que Ä()Ü ... etc ...

Six touches ont des fonctions spéciales :

Sur le clavier Shift NORM la touche ENTER appelle un octet en hexadecimal ou decimal qui sera concaténé à la chaine en cours d'édition.

Sur le clavier Shift USER les touches de la ligne supérieure permettent de concaténer des chaines prédéfinies, de 6 caracteres au plus, au registre ALPHA ou à la ligne de programme que l'on est en train d'éditer. Cette particularité est trés utile lorsque l'on programme des codes de contrôle d'imprimante.

Une particularite supplémentaire des extensions systèmes est la possibilité

d'accéder directement à tous les registres d'état comme on le fait actuellement avec les registres de la pile.

Les extensions énumérées ci-dessus rendent les programmes "LOAD BYTE", "KA" ou "MK" tout-à-fait depassés. Tout le reste - comme compiler des programmes utilisateur, renuméroter automatiquement les adresses de registres ou d'étiquettes, etc. - est facile a programmer au moyen des fonctions suivantes :

Utilitaires

C'est un groupe de fonctions permettant d'accéder à tous les emplacements de RAM de la HP-41. Contrairement à d'autres solutions travaillant avec des nombres non normalisés difficiles a créer et à déchiffrer (Finalement que vous apporte d'obtenir une chaine de 14 caractères en décodant le contenu du registre si vous ne désirez collecter qu'un octet ?), nous sommes en train d'élaborer un ensemble de fonctions opérant au niveau d'un octet. Ces fonctions permettent d'écrire des programmes qui chargent ou modifient d'autres programmes ou des portions de RAM, beaucoup plus facilement et plus rapidement qu'aucune autre méthode publiee à ce jour.

Les formats d'entrée suivants sont communs à toutes ces fonctions :

Une adresse est de la forme rrr.b ou rrr est le numéro absolu du registre (0 <= rrr < 1024) et b le numero de l'octet (0 <= b < 7) en commencant par l'extrémité droite du registre.

Lorsqu'une adresse est incrémentée, son numéro d'octet est incrémenté en premier. Si le resultat est 7, il est remplacé par 0 et le numéro du registre est incrémenté.

Une valeur d'octet bbb est une valeur numérique quelconque 0 <= bbb < 256. Tous les chiffres additionnels non nuls seront ignorés pour les entrées d'octets et d'adresses.

Voici une liste des fonctions : (X. Y. Z. T et L sont les registres de la pile).

PEEKB : Rappelle la valeur de l'octet pointé par l'adresse stockée en X, et la place en X. L'adresse est incrémentée ou décrémentée selon son signe et se retrouve en Y. L'ancienne adresse est conservée en L.

POKEB : Place la valeur donnée par X dans l'octet pointé par l'adresse située en Y. La pile tourne vers le bas (la valeur de l'octet est maintenant dans T) et l'adresse incrémentée ou décrémentée selon son signe.

PCTOX : Retourne en X le contenu du compteur de programme.

XTOPC : Ecrit une adresse dans le compteur de programme.

DBYTES : Calcule la différence en octets entre deux adresses stockées en X et Y.

PC+ : incrémente l'adresse en Y du nombre d'octets donné en X.

PEEKR et POKER : Analogues a PEEKB et POKEB. Ces fonctions opèrent sur les contenus de tout un registre (STOre et ReCaLl non normalisant). Ces deux fonctions sont fournies pour une copie rapide de blocs de registres complets entre un emplacement de RAM et un autre (par exemple : duplication d'un fichier en X-MEMORY).

Cet ensemble de fonctions est complété par des fonctions d'Entrée / Sortie appropriées et par les fonctions Booléiennes citées plus loin.

Quelques autres fonctions utilitaires seront offertes :

PLENG : Cette fonction (non programmable) invite a fournir un nom de programme et affiche sa longueur en nombre d'octets.

ABSP : Version programmable de la "flèche à gauche" ALPHA.

CLA- : Supprime les caractères de droite du registre ALPHA jusqu'à ce que l'on atteigne un espace.

SAVEBX : Sauvegarde en XF/M le tampon dont l'ID correspond à la valeur stockée en X.

GETB : Rappelle le tampon sauvegardé.

SAVEK : Sauvegarde les informations d'affectation de touches en XF/M.

GETK : Rappelle les affectations de touche après avoir effacé les anciennes.

MERGEK : Fusionne les affectations de touche contenues dans le fichier par dessus les affectations existantes.

SLFCHK : Effectue un autotest des ROMs HP-41 (y compris la ROM commutée a la page 5), des boitiers RAM (non destructif), du beeper et de l'afficheur.

FONCTIONS Mathématiques

Cet ensemble de fonctions n'est pas encore terminé. Nous y travaillons, ce qui signifie que tous les détails de fonctionnement ne sont pas définitifs. Nous donnerons ici une vue d'ensemble de ce que vous pouvez attendre de la ROM - Nous pensons que cela ne sonne pas mal du tout ... :

Imaginez un tableau quelconque à une ou deux dimensions. Il peut contenir des données numériques - c'est alors un vecteur ou une matrice - ou des données alphanumériques - c'est alors un bloc. Ce bloc, une fois créé avec les fonctions appropriees de la ROM, peut résider soit en RAM principale, soit en RAM XF/M. Le module CCD fournira toutes les fonctions nécéssaires pour travailler sur ces tableaux. Ceci signifie des opérations arithmétiques sur les matrices, échange ou copie de colonnes, de lignes ou de sous-tableaux, d'un tableau à un autre, multiplication de vecteurs, opérations sur les lignes et les colonnes (élément max. normes) etc ... Bien sûr, tout ceci est supporté par le nouvel EMDIR qui décodera correctement les types de fichiers. En outre, on fournira quelques routines spéciales d'entrée et de sortie de tableau.

D'autres fonctions mathématiques seront :

RNDM : C'est un générateur de nombres aléatoires. Vous pouvez charger une semence en utilisant la fonction SEED. ou bien le calculateur construira une semence a partir de quelque information de statu$ interne. La semence est stockée dans un tampon d'I/O.

FONCTIONS BOOLÉIENNES : Cet ensemble de fonctions n'est pas encore terminé. A nouveau, nous ne voulons pas donner trop d'informations préliminaires. Bornons nous à dire qu'elles complètent les possibilités des fonctions de manipulation d'octets ...

FONCTIONS d'Entrée/Sortie

Cet ensemble de fonctions aide le programmeur à implanter facilement une interface Utilisateur / Logiciel amicale. Les raisons pour lesquelles nous mettons ces choses dans la ROM sont les suivantes :

La vérification qu'une entrée est légale, combinée avec un message d'invite approprié, est souvent consommatrice de temps de calcul et d'espace RAM. Mais les facteurs temps d'exécution et taille programme sont trés souvent limitatifs. de sorte que de nombreuses applications manquent, soit de contrôles d'entrée, soit de messages d'invite adéquats, et même des deux. Ceci va changer avec l'arrivée du module CCD ! Plusieurs fonctions d'invite font tout le travail pour vous. Voici quelques exemples :

KEY? : Ecrivez "PRESS KEY YN" dans le registre ALPHA et exécutez KEY?. Regardez l'afficheur. Vous voyez : "PRESS KEY Y:N". Seules les touches Y ou N sont maintenant actives. L'appui sur Y ou N pousse un 1 ou un 2 sur la pile, respectivement. Si le registre ALPHA est vide, "HIT ANY KEY" sera affiché, et l'appui sur n'importe quelle touche ramènera son code de touche.

INPUT : Affiche le registre ALPHA et stoppe un programme en cours d'execution, attendant une entrée numérique. L'appui sur R/S se traduit par les opérations suivantes :

- S'il n'y a pas eu d'entrée, une valeur par défaut vient du registre pointé par REG00.

- La légalite de la valeur introduite est verifiée en testant si REG01 <= X <= REG02,

- Si le test est faux, l'invite est répétée en affichant d'abord les limites de l'entrée et ensuite l'invite ALPHA.

- Si ce test est vrai, X est stocké dans le registre pointe par REG00 et une instruction ISG est exécutée sur REG00,

- Une fois qu'une entrée légale a eu lieu, l'execution du programme continue avec la ligne de programme suivante.

INPUTI : Trés voisine de INPUT, sauf qu'elle n'accepte que les entréesentieres.

Plusieurs routines de formattage de SORTIE sont en cours d'achèvement. Elles seront compatibles avec toute les imprimantes disponibles pour le svsteme HP-41 - y compris les peripheriques IL.

INREC : Concatène au fichier texte courant, sous forme d'un enregistrement, une chaine ASCII venant de la boucle HPIL.

INRECX : Identique à INREC, mais n'accepte que le nombre d'octets spécifié en X.

OUTREC : Envoie l'enregistrement courant à l'appareil primaire de la boucle HPIL.

ACREC : Accumule l'enregistrement courant dans le tampon de l'imprimante (ne supprime pas le bit de gauche des codes de contrôle de l'imprimante et fonctionne admirablement bien avec l'imprimante HP 82143).

Voici donc - trés brièvement bien sur - ce que nous voulions vous dire concernant notre ROM. De nombreuses personnes travaillent en ce moment sur le microcode, et nous avons vu des réalisations interessantes dans ce domaine. Mais, à notre avis, il subsiste quelques inconvénients trés importants dans la pratique du microcode : Les boitiers de RAM et d'EPROM sont chers ou volumineux, ou les deux. Trés peu de gens peuvent utiliser la meme version d'un module, ce qui les empêche de rédiger du logiciel en vue d'une large diffusion. Le progrés que représente le module CCD va changer cela. Ici, nous offrons un ensemble complet de fonctions d'usage courant et faciles à utiliser, compatible avec toutes les versions de la calculatrice HP-41 (Sauf les toutes premières HP-41 dotées d'une ROM 0 en revision "D", pour lesquelles les nouvelles possibilités des fonctions ASN, CAT et XEQ ne sont pas accessibles) et tous ses modules enfichables et périphériques. Il donnera à votre calculatrice une augmentation importante de sa puissance de calcul, et même beaucoup plus si vous étendez votre systeme HP-41. Déja commande à plusieurs centaines d'exemplaires, ce module deviendra une extension standard de votre calculatrice, au même titre que les modules XF/M. Nous sommes heureux d'annoncer a tous les membes étrangers qu'ils peuvent commander ce module au prix de souscription spécial de 295 DM (Marks Allemands). La livraison des ROMs débutera en octobre de cette année. Le prix comprend une ROM, un overlay de clavier, un guide de référence rapide, et un manuel en anglais. Un membre PPC peut commander un maximum de deux ROMs.

Pour la commande, veuillez adresser un chèque tiré, en monnaie Allemande, sur une banque Allemande (de préférence la "DEUTSCH BANK AG. Cologne"), à l'ordre de :

W&W, DR.W.BALTES und W.KOETZ
Postfach 800133
D 5060 Bergisch Gladbach 2
République Fédérale d'Allemagne.

Tel : 19 / (49)2202-85068
ou : 19 / (49)2202-82026

Ecrivez vos nom et adresse LISIBLEMENT.
Une autre possibilité pour commander le module consiste à virer l'argent à notre compte-courant postal :

Dr.W.BALTES und KOETZ
5060 Bergisch Gladbach 2
Compte-courant postal :
4698 97-504 (Cologne).

HP-41

LE PETIT THEATRE DES MICROCODES:

1: Fonction RAND (Pierre ANTOINE)

RPN:

2: LLIST listage des lignes de texte (Frédéric POUPON)
3: Editeur de texte (Laurent JOLIA-FERRIER)

Pas moins de 1,5 Ko pour l'éditeur de texte: époustouflant !

Voir aussi le descriptif de la CCD ROM, traduit par André OISEL.

FONCTION RAND

Voici un generateur de nombres aleatoires en microcode. Il n'est pas tres original puisqu'il utilise le traditionnel RCL 00 9821 * 0,211327 + FRC STO 00 .

Il utilise un registre dans le X-Fonctions ,a l'adresse 0BD.Il faut creer un fichier de un registre et il faut que ce soit le premier en memoire.On peut d'ailleurs creer un fichier DATA plus long,tous les registres restent accessibles par les fonctions usuelles et ils ne sont pas perturbes par RAND,sauf le premier registre du fichier.

Le generateur tire un entier entre 1 et la valeur donnee en X.Le resultat est egalement en X,les autres registres de la pile ne sont pas modifies.En enlevant certaines parties du programme,RAND peut donner un nombre decimal,un nombre entre 0 et X,4 et X...

PIERRE ANTOINE P61 T384

```
C08E    084     D
C08F    00E     N

C090    001     A
C091    012     R
C092    2A0     SETDEC
C093    130     LDI             * RAPPEL DU NOMBRE ALEATOIRE
C094    0BD     CON     0189            REGISTRE 0BD
C095    270     DADD=C
C096    038     C=DATA
C097    0AE     AC EX   W       * MULTIPLICATION PAR 9821
C098    04E     C=0     W               184D:ROUTINE C=A.C
C099    35C     PT=     12
C09A    250     LC      9
C09B    210     LC      8
C09C    090     LC      2
C09D    050     LC      1
C09E    130     LDI
C09F    003     CON     0003

C0A0    135
C0A1    060     GOSUB   184D
C0A2    0AE     AC EX   W       * ADDITION + 0,211327
C0A3    04E     C=0     W               1807:ROUTINE C=A+C
C0A4    35C     PT=     12
C0A5    090     LC      2
C0A6    050     LC      1
C0A7    050     LC      1
C0A8    0D0     LC      3
C0A9    090     LC      2
C0AA    1D0     LC      7
C0AB    226     C=C+1   X
C0AC    286     C=-C    X
C0AD    01D
C0AE    061     GSUBC   1807
C0AF    0ED

C0B0    064     GOSUB   193B    * PARTIE FRACTIONNAIRE DE C
C0B1    2F0     DATA=C          * STOCKAGE EN 0BD
C0B2    0AE     AC EX   W       * MULTIPLICATION PAR X
C0B3    04E     C=0     W
C0B4    270     DADD=C
C0B5    0F8     C=REGN  3 (X)
C0B6    135
C0B7    060     GOSUB   184D
C0B8    088     S5=     1       * PARTIE ENTIERE DE C
C0B9    0ED
C0BA    064     GOSUB   193B
C0BB    0AE     AC EX   W       * AJOUTER 1 A C
C0BC    04E     C=0     W
C0BE    35C     PT=     12

C0C0    050     LC      1
C0C1    01D
C0C2    060     GOSUB   1807
C0C3    0E8     REGN=C  3 (X)   * STOCKER LE RESULTAT DANS X
C0C4    3E0     RTN
```

Chers amis.

Je ne sais pas si vous êtes comme moi, mais chaque fois que je désire passer un programme dans le journal, il y a une chose qui me rebute: la mauvaise traduction des lignes de texte par les imprimantes, quelles qu'elles soient. J'ai donc fait ce programme pour 41: le listeur de lignes de texte.

Partant d'un programme en mémoire étendue, il décode toutes les lignes de texte et fournit les codes décimaux à entrer. En voici tout de suite le listage et, du fait de la présence de quelques lignes synthétiques, un exemple d'éxecution:

```
01LBL "LLIST"
02 FIX 0
03 CF 29
04 7
05 PSIZE
06 "L"
07 EMDIR
08 CLD
09 CRFLAS
10 "P"
11 RCLPTA
12 3
13 -
14 STO 02
15 " "
16 RCL [
17 " "
18 RCL [
19 X<> c
20 RCL 00
21 STO [
22 " ABCDE"
23 X<> Z
24 STO \
25 " AB"
26 X<> \
27 STO 00
28 X<>Y
29 STO c
30 CLX
31 STO 00
32 CF 19
33 STO 04
34LBL 40
35 RCL 00
36 XEQ 30
37 STO 03
38 CF 10
39 240
40 X<=Y?
41 SF 10
42 X<>Y
43 XEQ 50
44 FS? 19
45 FC? 05
46 ISG 04
47 DEG
48 CF 19
49 FS? 05
50 SF 19
51 X=0?
52 DSE 04
53 X=0?
54 SIGN
55 FS? 10
56 GTO 60
57 ST+ 00
58LBL 41
59 RCL 02
60 RCL 00
61 X<Y?
62 GTO 40
63 "L"
64 0
65 SEEKPTA
66 SF 21
67LBL 09
68 SF 25
69 GETREC
70 FS? 25
71 AVIEW
72 FS? 25
73 GTO 09
74 "FIN"
75 ASTO X
76 STOP
77LBL 60
78 STO 05
79 CF 19
80 "L"
81 FLSIZE
82 ARCL 04
83 " : "
84 APPREC
85 "P"
86 FLSIZE
87LBL 07
88 RCL 00
89 XEQ 30
90 "L"
91 FLSIZE
92 CLA
93 RDN
94 ARCL X
95 " ,"
96 APPCHR
97 "P"
98 FLSIZE
99 1
100 ST+ 00
101 DSE 05
102 GTO 07
103 GTO 41
104LBL 50
105 ,
106 X<>F
107 RDN
108 X=0?
109 RTN
110 16
111 X>Y?
112 SF 00
113 SIGN
114 FS? 00
115 RTN
116 CLX
117 29
118 X>Y?
119 SF 05
120 X=Y?
121 GTO 28
122 1
123 +
124 X=Y?
125 GTO 28
126 CLX
127 144
128 X>Y?
129 SF 02
130 SIGN
131 FS?C 02
132 RTN
133 CLX
134 176
135 X>Y?
136 SF 02
137 FS? 02
138 2
139 FS?C 02
140 RTN
141 CLX
142 192
143 X>Y?
144 SF 01
```

```
145 FS? 01
146 SF 04
147 FS? 01
148 2
149 FS? 01
150 RTN
151 CLX
152 206
153 X>Y?
154 GTO 51
155 X=Y?
156 SF 02
157 FS? 02
158 2
159 FS? 02
160 RTN
161 CLX
162 207
163 X=Y?
164 SF 00
165 CLX
166 2
167 FS? 00
168 RTN
169 CLX
170 223
171 X<Y?
172 SF 03
173 RDN
174 240
175 X>Y?
176 SF 01
177 -
178 1
179 +
180 FS? 01
181 3
182 RTN
183LBL 28
184 RCL 00
185 1
186 +
187 XEQ 30
188 238
189 -
190 RTN
191LBL 51
192 RCL 00
193 2
194 +
195 XEQ 30
196 240
197 X<Y?
198 GTO 51
199 3
200 RTN
201LBL 51
202 -
203 3
204 +
205 RTN
206LBL 30
207 CLA-   —
208 STO [   -
209 7
210 /
211 INT
212 SEEKPT
213 GETX
214 X<> [
215 7
216 MOD
217 X0?
218 XEQ 03
219 STO \
220 " "
221 X<> \
222 STO [
223 ATOX
224 RTN
225LBL 03
226 " "
227 DSE X
228 GTO 03
229 RTN
230 .END.

L6: 241,76,
L10: 241,80,
L15: 245,32,0,0,0,0,
L17: 243,11,224,1,
L22: 246,127,65,66,67,68
,69,
L25: 243,127,65,66,
L63: 241,76,
L74: 243,70,73,78,
L80: 241,76,
L83: 243,127,58,32,
L85: 241,80,
L90: 241,76,
L95: 242,127,44,
L97: 241,80,
L220: 242,127,0,
L226: 242,127,0,
```

Quelques mots maintenant à propos de l'utilisation:

Vider la mémoire étendue, sauver le programme à étudier sous le nom de fichier "P". Faire XEQ LLIST et attendre. si une imprimante est connectée, vous aurez le résultat noir sur blanc, sinon tapez R/S après chaque ligne. C'est aussi simple que cela.

Si vous vous amusez à décortiquer ce programme, vous vous rendrez compte que son architecture interne ressemble beaucoup à celle de "COMP", c'est normal, les deux proogrammes datent de la même époque et utilisent tous deux des routines de lecture d'un octet et de calcul de la longueur d'une instruction.

La routine de calcul de la longueur d'une instruction est située à partir du label 50. Une fonction . PEEK est implémentée au label 30. Les résultats sont enregistrés temporairement dans le fichier ascii "L".

Voila, maintenant vous n'avez plus aucune raison de garder vos programmes pour vous. On veut les voir rapidement dans JPC !

C'est tout pour cette fois ci. Au plaisir de vous lire!

Frédéric POUPON (#35).

EDITEUR DE TEXTE SUR HP-41

INTRODUCTION:
L'éditeur est composé des programmes suivants: AB, AP, BO, CA, CH, CL, CR DO, ED, FI, HE, IN, KI, OF, PR, PU, RE, TO, UP, WH, DISPLAY, SYSTEM, AP+IN. SYSTEM est un analyseur syntaxique; DISPLAY affiche les messages; les autres programmes correspondent aux différentes fonctions.

DONNEES TECHNIQUES:
L'éditeur fait un peu moins de 1,5 Ko et réclame un SIZE de 74. Le SIZE peut être ramené à 39, mais les développements ultérieurs en seront réduits. Je reste à la disposition des personnes intéressées.

DEFINITIONS ET CONVENTIONS:
[ ] signifie que les éléments entre crochets peuvent être omis; ils prennent alors la valeur par défaut.
P représente un paramètre
s représente un séparateur
C représente un couple de la forme **Ps**
alpha représente une chaîne de caractères. La valeur ASCII de chacun est comprise entre 0 et 96 inclus. La valeur par défaut de **alpha** est une chaîne vide de longueur 0.
num représente un nombre entier. Chaque caractère composant **num** a une valeur ASCII comprise entre 16 et 25 inclus (chiffres 0 à 9), plus 43 et 45 (signes + et -). La valeur par défaut de **num** est 1.
numpos représente un nombre entier positif. Même valeur ASCII que précédemment sauf 45.
asc représente une valeur ASCII. C'est un entier positif compris entre 0 et 255.

STRUCTURE D'UNE COMMANDE:
Important: TOUTE COMMANDE EST SAISIE EN MODE ALPHA.
Toute commande a la forme: CCC ... C avec 12 fois C au maximum.
Soit aussi: PsPsPs ... Ps avec 24 lettres au maximum.
s peut prendre les valeurs suivantes: a (ASCII 97) , b (ASCII 98), c (ASCII 99).

Si	alors HP 41 considère
s=a	P=alpha
s=b	P=num ou numpos
s=c	P=asc

Exemples de C:
38a chaîne de caractère "38"
JOLIA-FERRIERa chaîne de caractères "JOLIA-FERRIER"
38c valeur ASCII 38 (équivaut au caractère &)
38b valeur numérique

Exemples de commandes:
EDaPHONEa séléction de PHONE comme fichier de travail
CHaETa38c99b changement 99 fois de la chaîne "ET" en la chaîne de valeur ASCII 38, soit &
EDaPHONE commande invalide car de la forme CP
EDaPHONEb commande invalide car b = num , or PHONE est alpha

Dans ce qui suit, on peut remplacer alpha**a** par asc**c** à volonté.

COMMANDES DE L'EDITEUR DE TEXTE

ABSOLUTE
ABa[numposb]
Positionnement sur la ligne de numéro numpos. Attention !!! La valeur par défaut de numpos est 1, ce qui correspond à la deuxième ligne d'un fichier.

APPEND
APa[alphaa[numposb]]
Ajout à la fin de la ligne courante de la chaîne alpha, numpos fois. Si alpha est omis, alors exécution de la commande AP+IN.

BOTTOM
BOa
Positionnement sur la dernière ligne d'un fichier.

CATALOG
CAa
Exécution de la fonction EMDIR du module XFonctions.

CHANGE
CHaalphaa[alpha]a[numposb]
Changement de la première chaîne alpha en la deuxième, numpos fois.

CLEAR
CLaalphaa
Effacement du contenu du fichier alpha et de son contenu seul.

CREATE
CRaalphaanumposb
Création d'un fichier alpha de type ASCII et de taille numpos.

DOWN
DOa
Positionnement sur la ligne suivant la ligne courante.

EDIT
EDaalphaa
Sélection du fichier alpha comme fichier de travail.

FIND
FIaalphaa[numposb]
Recherche numpos fois de la chaîne alpha.

HELP
HEa
Affichage du nom des fonctions disponibles.

INPUT
INa[alphaa[numposb]]
Ajout après la ligne courante de numpos lignes constituées de alpha. Si alpha est omis, alors éxécution de la commande AP+IN.

KILL
KIa[numposb]
Destruction de la ligne courante et de numpos-1 lignes suivantes. Si la fin du fichier est atteinte, la ligne précédente devient la ligne courante.

OFF
OFa
Fin du programme.

PRINT
PR**a**[numpos**b**]
Affichage de numpos lignes à partir de la ligne courante.

PURGE
PU**a**alpha**a**
Destruction du fichier alpha de la mémoire étendue.

RELATIVE
RE**a**[num**b**]
Déplacement à la num-ième ligne précédente (num négatif), ou suivante (num positif).

TOP
TO**a**
Positionnement sur la première ligne d'un fichier (numérotée 0).

UP
UP**a**
Positionnement sur la ligne précédant la ligne courante.

WHERE
WH**a**
retourne le numéro de ligne sur laquelle on est positionné.

AP+IN
Accès par APPEND ou INPUT.
Si l'on frappe une chaîne alpha, elle est ajoutée à la suite de la ligne courante et devient la nouvelle ligne courante.
Si l'on frappe +alpha, alpha est rajouté à la suite de la ligne courante.
Si l'on frappe R/S, on retourne à SYSTEM.

Laurent JOLIA-FERRIER
10, square Jules Ferry
95- Sannois

DIVERS

Fonctions permettant le positionnement sur un fichier: EDIT

Fonctions nécessitant un positionnement pour opérer:
ABSOLUTE, APPEND, BOTTOM, CHANGE, DOWN, FIND, INPUT, KILL, PRINT, RELATIVE, TOP, UP, WHERE, AP+IN.

Fonctions faisant perdre le positionnement:
CATALOG, CLEAR, CREATE, PURGE.

Autres fonctions:
HELP, OFF.

Ligne synthétique de SYSTEM:
247, 0, 0, 0, 40, 0, 128, 128

Ligne synthétique de OFF:
247, 0, 0, 0, 44, 4, 128, 0

MESSAGES D'ERREURS DE L'EDITEUR DE TEXTE

OK
La commande précédente a été correctement exécutée.

SYSTEM ERROR
La commande ne respecte pas les règles de syntaxe ou la commande n'existe pas ou le séparateur a un code ASCII supérieur à 99.

CHECK X MEM
Plus de place dans la mémoire étendue ou fichier déjà existant (fonction CREATE).

EDIT A FILE
Il n'y a pas (plus) de positionnement sur un fichier; faire EDIT.

NO SUCH FILE
Le fichier concerné par la commande n'existe pas.

NO SUCH LINE
La ligne concernée par la commande n'existe pas.

FILE TOP
Le programme est déjà positionné sur la ligne 0 du fichier (début).

FILE BOTTOM
Le programme est déjà positionné sur la dernière ligne du fichier.

FILE FULL
Le fichier est plein; il se peut qu'une partie seulement de l'insertion réclamée ait été faite; le vérifier.

CARTE DES DIFFERENTS REGISTRES

00 à 23: Commande éclatée lettre par lettre
24 à 71: Commande éclatée paramètre par paramètre
72: Pointeur vers les registres 00 à 23 dans SYSTEM.
    Ligne courante avant exécution d'une commande dans les autres programmes
73: Pointeur vers les registres 24 à 71 dans SYSTEM.
    Ligne courante dans les autres programmes.
L: Pointeur vers les registres 00 à 23 dans SYSTEM.

MESSAGES ET DRAPEAUX ASSOCIES

OK	0	0	0	0	0	0	0	0	0
SYSTEM ERROR	1	0	0	0	0	0	0	0	1
CHECK X MEM	0	1	0	0	0	0	0	0	2
EDIT A FILE	1	1	0	0	0	0	0	0	3
NO SUCH FILE	0	0	1	0	0	0	0	0	4
NO SUCH LINE	1	0	1	0	0	0	0	0	5
FILE TOP	0	1	1	0	0	0	0	0	6
FILE BOTTOM	1	1	1	0	0	0	0	0	7
FILE FULL	0	0	0	1	0	0	0	0	8
	0	1	2	3	4	5	6	7	RCLFLAG

```
 01LBL "AP+IN"
RCLPT STO 72 STO 73

 05LBL 00
CLA TONE 9 STOP ALENG
X=0? GTO 03 43 ATOX
X=Y? GTO 02 XTOA - E
AROT RCL 73  E +
SEEKPT FS? 25 GTO 01
SF 25 APPREC FC? 25
GTO 04 RCLPT INT
SEEKPT STO 73 GTO 00

 34LBL 01
INSREC FC? 25 GTO 04
STO 73 GTO 00
 40LBL 02
APPCHR FC? 25 GTO 04
GTO 00
 45LBL 03
RCL 72 SEEKPT RTN

 49LBL 04
SF 25 8 X<>F GTO 03
END
```

```
 01LBL "AP"
FC? 08 GTO 03 "0000"
ASTO Y CLA ARCL 28
ARCL 29 ARCL 30
ARCL 31 ASTO X X=Y?
GTO "AP+IN" RCLPT
STO 72
 16LBL 00
APPCHR FC? 25 GTO 02
DSE 32 GTO 00
 22LBL 01
RCL 72 SEEKPT RTN

 26LBL 02
SF 25 8 X<>F GTO 01

 31LBL 03
3 X<>F END
```

```
 01LBL "AB"
FC? 08 GTO 01 RCL 28
SEEKPT FS? 25 GTO 00
4 X<>F RTN
 11LBL 00
GETREC PSE FS? 17
GTO 00 SEEKPT RTN

 18LBL 01
3 X<>F END
```

```
 01LBL "CH"
FC? 08 GTO 04 RCLPT
STO 72
 06LBL 00
ARCL 28 ARCL 29
ARCL 30 ARCL 31 POSFL
X<0? GTO 02 ALENG
DELCHR CLA ARCL 32
ARCL 33 ARCL 34
ARCL 35 INSCHR FC? 25
GTO 03 CLA DSE 36
GTO 00
 27LBL 01
RCL 72 SEEKPT FS? 25
RTN SF 25  E -
SEEKPT RTN
 37LBL 02
SF 25 7 X<>F GTO 01

 42LBL 03
SF 25 8 X<>F GTO 01

 47LBL 04
3 X<>F END
```

```
 01LBL "UP"
FC? 08 GTO 02 RCLPT
X=0? GTO 01  E -
SEEKPT
 10LBL 00
GETREC PSE FS? 17
GTO 00 SEEKPT RTN

 17LBL 01
6 X<>F RTN
 21LBL 02
3 X<>F END
```

```
 01LBL "BD"
FC? 08 GTO 03 FLSIZE
3.5 * STO 72
 08LBL 00
RCL 28 RCL 72 + 2 /
INT SEEKPT FC? 25
GTO 01 X≠Y? GTO 02
"BOTTOM " ARCL X PSE
RTN
 24LBL 01
SF 25 STO 72 GTO 00

 28LBL 02
STO 28 GTO 00
 31LBL 03
3 X<>F END
```

```
 01LBL "SYSTEM"
SIZE? 74 X>Y? PSIZE
"(   " RCL [ STO d

 09LBL 00
XEQ "DISPLAY" CLRG
TONE 9 STOP ALENG
X=0? GTO 06  E -  E3
/ STO L STO 72 24.023
STO 73
 25LBL 01
ATOX STO IND L ISG L
GTO 01 96 STO L

 32LBL 02
LASTX RCL IND 72 X>Y?
GTO IND X XTOA ISG 72
GTO 02 GTO 06
 41LBL 97
4 E-3 ST+ 73
 44LBL 03
ASTO IND 73 ASHF
ISG 73 GTO 03 GTO 05

 50LBL 98
 E-3 ST+ 73 ANUM
FC?C 22 GTO 06 CLA
STO IND 73 GTO 05

 59LBL 99
4 E-3 ST+ 73 ANUM
FC?C 22 GTO 06 CLA
XTOA
 67LBL 04
ASTO IND 73 ASHF
ISG 73 GTO 04
 72LBL 05
ISG 72 GTO 02 RCL 24
SF 25 XEQ IND X FS? 25
GTO 00 SF 25
 81LBL 06
1 X<>F GTO 00 END
```

```
 01LBL "OF"
", " RCL [ STO d
STOP END
```

```
 01LBL "CR"
CF 08 ARCL 28 ARCL 29
ARCL 30 ARCL 31 RCL 32
CRFLAS FS? 25 RTN
SF 25 2 X<>F END
```

```
 01LBL "CA"
CF 05 EMDIR ARCL X
"  FREE REG" AVIEW PSE
CLD END
```

```
 01LBL "PU"
CF 08 ARCL 28 ARCL 29
ARCL 30 ARCL 31 PURFL
FS? 25 RTN 4 X<>F
END
```

```
 01LBL "WH"
FC? 08 GTO 00 RCLPT
"LINE " ARCL X PSE
RTN
 09LBL 00
3 X<>F END
```

```
 01LBL "PR"
FC? 08 GTO 03 RCLPT
STO 72
 06LBL 00
GETREC FC? 25 GTO 02
PSE FS? 17 GTO 00
DSE 28 GTO 00
 15LBL 01
RCL 72 SEEKPT RTN

 19LBL 02
SF 25 7 X<>F GTO 01

 24LBL 03
3 X<>F END
```

```
PRP "FI"

 01LBL "FI"
FC? 08 GTO 04 RCLPT
STO 72
 06LBL 00
ARCL 28 ARCL 29
ARCL 30 ARCL 31 POSFL
X<0? GTO 03 INT
"LINE " ARCL X PSE
CLA DSE 32 GTO 02

 21LBL 01
RCL 72 SEEKPT RTN

 25LBL 02
 E + SEEKPT FS? 25
GTO 00 SF 25
 32LBL 03
7 X<>F GTO 01
 36LBL 04
3 X<>F END
```

```
 01LBL "CL"
CF 08 ARCL 28 ARCL 29
ARCL 30 ARCL 31 CLFL
FS? 25 RTN SF 25 4
X<>F END
```

```
 01LBL "KI"
FC? 08 GTO 03 RCLPT
STO 72
 06LBL 00
DELREC FC? 25 GTO 02
DSE 28 GTO 00
 12LBL 01
RCL 72 SEEKPT FS? 25
RTN SF 25  E -
SEEKPT RTN
 22LBL 02
SF 25 7 X<>F GTO 01

 27LBL 03
3 X<>F END
```

```
 01LBL "IN"
FC? 08 GTO 04 "0000"
ASTO Y CLA ARCL 28
ARCL 29 ARCL 30
ARCL 31 ASTO X X=Y?
GTO "AP+IN" RCLPT
STO 72  E + SEEKPT
FS? 25 GTO 01 SF 25

 22LBL 00
APPREC FC? 25 GTO 03
DSE 32 GTO 00 GTO 02

 29LBL 01
INSREC FC? 25 GTO 03
DSE 32 GTO 01
 35LBL 02
RCL 72 SEEKPT RTN

 39LBL 03
SF 25 8 X<>F GTO 02

 44LBL 04
3 X<>F END
```

```
 01LBL "ED"
CF 08 ARCL 28 ARCL 29
ARCL 30 ARCL 31 .
SEEKPTA FC? 25 GTO 00
SF 08 RTN
 13LBL 00
SF 25 4 X<>F END
```

```
 01LBL "TO"
FC? 08 GTO 00 CLX
SEEKPT "TOP 0" PSE
RTN
 09LBL 00
3 X<>F END
```

```
 01LBL "RE"
FC? 08 GTO 01 RCLPT
RCL 28 + X<0?  E9
SEEKPT FS? 25 GTO 00
SF 25 5 X<>F RTN

 16LBL 00
GETREC PSE FS? 17
GTO 00 SEEKPT RTN

 23LBL 01
3 X<>F END
```

```
 01LBL "DISPLAY"
. X<>F GTO IND X

 05LBL 00
"OK" RTN
 08LBL 01
"SYSTEM ERROR" RTN

 11LBL 02
"CHECK X MEM" RTN

 14LBL 03
"EDIT A FILE" RTN

 17LBL 04
"NO SUCH FILE" RTN

 20LBL 05
"NO SUCH LINE" RTN

 23LBL 06
"FILE TOP" RTN
 26LBL 07
"FILE BOTTOM" RTN

 29LBL 08
"FILE FULL" RTN END
```

```
 01LBL "HE"
"ABSOLUTE" PSE
"APPEND" PSE "BOTTOM"
PSE "CATALOG" PSE
"CHANGE" PSE "CLEAR"
PSE "CREATE" PSE
"DOWN" PSE "EDIT" PSE
"FIND" PSE "HELP" PSE
"INPUT" PSE "KILL"
PSE "OFF" PSE "PRINT"
PSE "PURGE" PSE
"RELATIVE" PSE "TOP"
PSE "UP" PSE "WHERE"
PSE END
```

```
 01LBL "DO"
FC? 08 GTO 01 RCLPT
 E + SEEKPT FS? 25
GTO 00 SF 25 7 X<>F
RTN
 14LBL 00
GETREC PSE FS? 17
GTO 00 SEEKPT RTN

 21LBL 01
3 X<>F END
```

```
CAT 1
LBL 'AB
END             40 BYTES
LBL 'AP
END             72 BYTES
LBL 'AP+IN
END             102 BYTES
LBL 'BO
END             67 BYTES
LBL 'CA
END             29 BYTES
LBL 'CH
END             90 BYTES
LBL 'CL
END             29 BYTES
LBL 'CR
END             31 BYTES
LBL 'DO
END             44 BYTES
LBL 'ED
END             35 BYTES
LBL 'FI
END             73 BYTES
LBL 'HE
END             149 BYTES
LBL 'IN
END             95 BYTES
LBL 'KI
END             55 BYTES
LBL 'OF
END             22 BYTES
LBL 'PR
END             51 BYTES
LBL 'PU
END             27 BYTES
LBL 'RE
END             48 BYTES
LBL 'TO
END             28 BYTES
LBL 'UP
END             42 BYTES
LBL 'WH
END             29 BYTES
LBL 'DISPLAY
END             134 BYTES
LBL 'SYSTEM
END             180 BYTES
```


S.O.S.

Dans le manuel du module HP-IL, page 39 "exécution automatique des programmes", il est précisé que l'indicateur 11 permet un démarrage automatique d'un programme rechargé à partir du clavier avec les instructions READP, READSUB, READA.
Alors qu'avec la première et la troisième solution, cela fonctionne correctement, le programme refuse le démarrage automatique quand j'utilise "READSUB", et je retrouve le pointeur sur un 01 END qui vient de je ne sais où.

Par ailleurs, j'ai beaucoup de problèmes pour générer des chaînes de caractères avec le BG. L'exemple ci-joint devrait me permettre de faire une chaîne dont les codes sont 247 11 224 1 105 11 224 176. La chaîne générée ne tient pas compte des deux derniers caractères. Et comment peut-on faire pour générer une chaîne du genre 246 250 12 8 15 192 255 ? (programmes P et PP de Robert Schwartz parus dans JPC de septembre).

D'avance merci

Denis CASTELAIN (P 56)

Réponse du dr JPC:

En ce qui concerne l'éxécution automatique, il est préférable de ne pas exécuter READSUB. En effet, comme ses consoeurs du lecteur de cartes et du module XFonctions, READSUB a pour effet de lire le programme et de le recopier dans l'optique d'être utilisé comme sous-programme. A cet effet, il place un END après le dernier programme en mémoire, pour ne pas l'écraser. Si ce dernier programme est déjà terminé par un END, c'est tant pis pour la beauté du catalogue ! Il y aura des END partout. Dans l'optique d'une éxécution automatique, par un programme principal appelant des sous-programmes à la chaîne, le dernier de ces sous-programmes n'a pas de END à la fin.

Pour ce qui est des chaînes synthétiques, le BG me semble être un Buldozer peu adapté à ce genre de travail, tout en finesse. HP, dans son infinie bonté, nous a fourni le MU. Il est idéal pour ce genre de chaîne, ne dépassant pas 7 caractères. Pour plus de détails, voir l'OI n° 27. Si suffisamment de personnes sont interessées, je ferai peut-être un article dessus dans un prochain numéro.

A bientôt...

Pierre DAVID

HP-15

Pour vous qui possédez, comme moi, une HP 15C, voici deux formules permettant de réaliser les fonctions "ln(1+x)" et "e^x-1" de la HP 41C:

$$\ln(1+x) = 2\,th^{-1}\left(\frac{x}{2+x}\right) \quad \text{et}$$

$$e^x-1 = 2e^{x/2}sh\left(\frac{x}{2}\right)$$
$$= (e^x+1)th\left(\frac{x}{2}\right)$$

Exemples:
Pour x= $1{,}23456789.10^{-8}$, la valeur exacte $\left(x-\frac{x^2}{2}\right)$ est

$1{,}234567882.10^{-8}$
Ma formule fournit: $1{,}234567883.10^{-8}$, alors que ln(1+x) sur HP 15C donne: 1,199999993.
Pour e^x-1, c'est aussi excellent.

Amicalement,

Daniel SAADA
15 rue Le Nôtre
78120 Rambouillet

Note de l'éditeur:
th^{-1} signifie Argument Tangente Hyperbolique, et non pas un sur tangente hyperbolique ...

HP-71

Beaucoup de choses nouvelles ce mois-ci:
D'abord la nouvelle présentation qui, nous l'espérons, vous plaira,
Ensuite, l'ouverture de la sous-rubrique Forth, en attendant
l'Assembleur... et surtout la quantité et la qualité des programmes.
Débutants ou programmeurs avertis: à vos claviers !

FORTH:

1: Course de voiture

BASIC:

2: POLYCOPY	(Pierre DAVID)
3: CHARSET	(Michel MARTINET)
4: Routines graphiques	(Jean-Pierre BONDU)
5: DATE	(Pierre DAVID)
6: MATRICE	(Laurent ISTRIA)
7: DESSIL	(Jean-Pierre BONDU)
8: Editeur de cassette	(Michel MARTINET)
9: Gestion de l'HP-IL	(Pierre DAVID)

Les programmes, du fait de la nouvelle présentation, sont listés à
la fin des articles, c'est à dire:
Pour le FORTH, en page 22
Pour le BASIC, à patir de la page 32.
Vous trouverez un exemple d'exécution du programme CHARSET à la page 31.
Il contient les caractères accentués utilisés par la HP82905B, celle
qui "écrit" le JPC. Utilisez donc ces caractères pour vos envois
d'articles.
Au plaisir de vous lire...

COURSE DE VOITURE, pour HP 71B + module Forth

En complément du banc d'essai du module Forth/Assembleur paru le mois dernier, je vous propose maintenant un "mot" Forth : la course de voiture.

C'est la traduction du programme Basic paru dans le JPC de mai 1984 (numéro 14). Il était intéressant de confronter les deux versions, sur le plan de la rapidité et de la souplesse d'utilisation, mais aussi sur le plan de la facilité de programmation.
Pour ce qui est de la rapidité, j'ai appris que le record (en Basic) était d'environ 60 secondes au niveau difficile. Je reconnais d'ailleurs que je suis largement battu ! A la dernière réunion, la version Forth circulait, et le record s'est établi aux alentours de 10 secondes . Je vous laisse juger, en précisant que mon record personnel est d'environ 1 seconde ! D'autre part, pour ce qui est de la facilité de programmation, je vous assure que Forth est plus agréable que Basic ! Avec en plus le plaisir de faire de la programmation structurée...
Mais trève de discours voyons l'utilisation !

Il est difficile de faire plus simple: tapez un nombre entre 0 et 1FFFF (si vous êtes en hexadécimal), n'oubliez pas l'espace, et tapez COURSE.
Accrochez-vous, c'est parti !

Appuyez sur [<] ou [Q] pour tourner à gauche, [>] ou [/] pour tourner à droite. Et tachez de rester le plus longtemps possible sur cette damnée route qui n'arrête pas de tourner !

Quelques remarques maintenant sur le programme lui-même: comme vous pouvez le constater, j'ai essayé de le documenter au maximum. L'algorithme est bien visible dans le mot "COURSE" . Les mots précédents ne sont que des "sous-programmes", dont certains n'ont pas été intégrés dans le corps du mot COURSE pour raison de lisibilité.
Le Forth 83 offre des possibilités de traitement de chaînes de caractères : je ne me suis pas privé de les utiliser ! Pour ceux qui ne les connaissent pas, une chaîne est représentée dans la pile par son adresse, et le nombre de caractères qui la compose. S! (analogie avec !) stocke la chaîne à l'adresse spécifiée sur le sommet de la pile. NULL$ renvoie la chaîne nulle , CHR$ stocke dans le PAD une chaîne d'un seul caractère dont le code est sur le sommet de la pile. S<& est la concaténation à droite.
Quelques notes, maintenant, sur le générateur de nombres "aléatoires" : le principe est simple: la mémoire morte qui contient le Basic contient aussi 131072 quartets dont les valeurs sont imprévisibles ! Il suffit de stocker une adresse, qui sera changée à chaque appel de RND. Cette adresse est placée dans la variable RANDOMIZE.
Pour ce qui est de la route, on affiche le caractère 128 (arbre) 22 fois, puis on passe en Basic pour changer la fenêtre d'affichage entre 9 et 13.
Pour la mise en place du jeu de caractères secondaire, j'utilise une table de valeurs, qui n'est rien d'autre qu'une suite de constantes mises dans la pile. FORTH$ retourne à Basic la chaîne placée au sommet de la pile.
Les adresses 2F946, et 2F948 codent le DELAY. Pour connaître la valeur à mettre, multipliez la durée par 32. Ca me permet de vider le tampon de touches.

Vous pourrez constater, en utilisant ce programme, que si vous appuyez trop rapidement sur les touches, une certaine inertie se fera sentir. Cela est dû à l'absence de fonction KEYDOWN en Forth.

Dans la catégorie des jeux bien débiles, celui-ci tient bien sa place. Je ne pense pas que vous vous éleverez intellectuellement, mais j'espère que, grâce à lui, vous passerez quelque bon moment...

A bientôt,

Pierre DAVID

HP 71 + Forth : COURSE DE VOITURE

```
  DECIMAL                    ( toutes les constantes sont exprimées en décimal)
  12 STRING JCAR             ( variable chaîne de 12 caractères: jeu de caractères secondaire)
   5 STRING ROUTE            ( 5 caractères pour représenter la route, et la voiture)
  VARIABLE VOITURE           ( position de la voiture sur la route, entre 0 et 4)
  VARIABLE RANDOMIZE         ( adresse pour le générateur de nombres "aléatoires")
-------------------------------------------------------------------------
: CAR                        ( mise en place du jeu de caractères)
  NULL$ JCAR S!              ( la chaîne JCAR est initialisée à la chaîne nulle)
  64 224 80 80 224 64        ( codes des caractères 128)
   0  14 31 255 31 14        ( et 129)
  JCAR 12 0                  ( boucle à éxécuter 12 fois: 2 caractères)
  DO
    ROT CHR$ S<&             ( on ajoute le code voulu)
  LOOP
  SP! JCAR                   ( SP! efface la pile)
  " CHARSET FORTH$" BASICX ; ( on établit le CHARSET à l'aide de JCAR et de BASICX: merci Hewlett !)
-------------------------------------------------------------------------
: INECRAN                    ( initialisation de l'écran)
  27 EMIT 69 EMIT            ( effacement)
  27 EMIT 60 EMIT            ( extinction du curseur)
  22 0                       ( pour les 22 caractères de l'affichage)
  DO
    128 EMIT                 ( placer un arbre)
  LOOP
  "WINDOW9,13" BASICX ;      ( Merci Hewlett - bis !)
-------------------------------------------------------------------------
: AFF                        ( affichage de la route, et de la voiture)
  "      " ROUTE S!          ( la chaîne ROUTE est initialisée à 5 espaces: rentrez donc 6 espaces)
  CR                         ( retour chariot pour effacer la route précédente)
  129 ROUTE DROP             ( dans la pile: 129, adresse de la route)
  VOITURE @                  ( position de la voiture)
  2* + C!                    ( on place 129 à l'adresse de la voiture dans la chaîne ROUTE)
  ROUTE TYPE ;               ( on affiche ROUTE)
-------------------------------------------------------------------------
: RND                        ( même rôle qu'en Basic)
  RANDOMIZE @                ( adresse où on va chercher un quartet)
  DUP N@                     ( on sauve cette adresse, et on prend le quartet)
  DUP 0=                     ( on teste s'il est nul)
  IF
    DROP 1                   ( si oui, on le remplace par 1)
  THEN
  DUP >R                     ( on sauve cette valeur dans la pile de retours)
  + RANDOMIZE !              ( nouvelle adresse, stockée dans RANDOMIZE)
  R> ;                       ( récupération du quartet)
-------------------------------------------------------------------------
: MVTALEA                    ( mouvement aléatoire)
  VOITURE @                  ( la position de la voiture reste dans la pile)
  RND
  8 <
  IF                         ( 50% pour la droite, 50% pour la gauche)
    1-                       ( à gauche)
   ELSE
    1+                       ( à droite)
  THEN
  VOITURE ! ;                ( c'est la nouvelle position de la voiture)
-------------------------------------------------------------------------
```

```
: TOUCHE                    ( test de la touche pressée)
  VOITURE @                 ( position de la voiture)
  KEY                       ( valeur de la touche)
  CASE                      ( structure-géniale !)
     8 OF 1- ENDOF          ( touche <-)
    81 OF 1- ENDOF          ( touche Q)
     9 OF 1+ ENDOF          ( touche ->)
    47 OF 1+ ENDOF          ( touche /)
    DROP MVTALEA VOITURE @  ( sinon mouvement aléatoire)
  ENDCASE                   ( simple, non ?)
  VOITURE ! ;               ( c'est donc la nouvelle position de la voiture)

  HEX                       ( toutes les constantes en hexadécimal)
-------------------------------------------------------------------
: COURSE                    ( Le Mot Principal, celui qu'on éxécute, qui fait comprendre l'algorithme !)
  RANDOMIZE ! ?STACK        ( on initialise le générateur de nombre aléatoire, et erreur si pile vide)
  2 VOITURE !               ( initialisation de la voiture)
  CAR                       ( établissement du jeu de caractères secondaire)
  INECRAN                   ( initialisation de l'écran)
  " T=TIME" BASICX          ( stockage de l'heure)
  BEGIN                     ( début de la boucle...)
    AFF                     ( affichage de la route)
    ?TERMINAL               ( teste si une touche est appuyée)
    IF
       TOUCHE               ( si oui, test)
     ELSE
       MVTALEA              ( si non, mouvement aléatoire)
    THEN
    VOITURE @ DUP           ( position de la voiture)
    0< SWAP 4 > OR          ( si <0 ou >4 alors on termine le jeu ici)
  UNTIL
  " WINDOW1" BASICX         ( on remet tout en place)
  1 2F946 C!                ( <=> DELAY#0)
  1 2F948 C!                ( pour vider le tampon de 15 touches)
  CR " TIME-T;" BASICX      ( affichage du score)
  0 2F946 C!                ( on remet le DELAY à 0,0)
  0 2F948 C! ;              ( voilà, c'est fini !)
```

POLYCOPY

Vu le nombre de HP 71 dans notre Club, il devenait urgent de créer un programme pour transférer simultanément un programme vers plusieurs 71. J'espère voir ce programme fonctionner dans toutes les réunions organisées par PPC-PARIS.
Le nombre de HP 71 est quelconque (entre 2 et 930...), donc en particulier, ce programme est valable pour passer des fichiers d'un 71 vers un seul autre.
La boucle peut contenir aussi des imprimantes, des lecteurs de cassettes, de disquettes...

L'utilisation est on ne peut plus simple: faîtes [RUN], puis entrez le nom des fichiers à envoyer. Tapez [END LINE] chaque fois. Pour terminer, frappez une dernière fois [END LINE]. Attention cependant: faîtes RESTORE IO si votre HP 71 est déclaré OFF IO. Ceci est valable pour tous les 71 de la boucle, aussi bien l'émetteur que les récepteurs.

Si il y a erreur dans l'introduction d'un fichier, le programme l'ignore.
Si le fichier existe déjà dans la mémoire d'un des récepteurs, ou si celle-ci est insuffisante pour stocker le fichier, ce HP 71 émettra un signal sonore, mais les autres recevront quand même le fichier.
Si le fichier n'existe pas dans la mémoire de l'émetteur, l'ordre est ignoré.
Pour toute erreur, un petit délai est nécessaire afin de préparer à nouveau les appareils.

Principe: deux solutions étaient possibles.
- Soit transférer successivement le fichier à tous les HP 71 dans la boucle. Cette solution n'est pas envisageable pour un fichier un peu gros (plus d'un Ko), ou pour un bon nombre de 71.
- Soit préparer tous les HP 71 à recevoir le fichier. La taille du fichier n'influe alors plus beaucoup sur le temps d'éxécution. J'ai donc appliqué cette méthode.

Voilà, ce programme occupe 505 octets. Si l'application n'est pas immédiate, il peut être profitable de l'étudier car, croyez-moi, l'HP-IL n'est pas vraiment simple !

A bientôt...
Pierre DAVID

==========

CHARSET

Si vous désirez manipuler votre jeu de caractères secondaire avec simplicité, alors ce programme vous est destiné. Il vous permet de modifier un caractère (ou plus si besoin), de l'effacer ou de l'imprimer si vous disposez d'une imprimante large (132 colonnes, exemple HP 82905B). Vous pouvez séléctionner un jeu de caractères parmi ceux que vous avez créés.

Principe:
Lors de l'initialisation, le programme crée un fichier TEXT de 768 caractères nuls (CHR$(0)), que vous pouvez modifier à loisir. Vous rentrez un caractère en donnant les codes de ses colonnes constitutives. Pour plus de détails, référez-vous au manuel d'utilisation (à partir de la page 132). A la fin du programme, ce jeu deviendra le jeu "actif" si celui-ci n'est pas entièrement nul.

Utilisation:
Faîtes RUN.
Le programme affiche "Fichier :_____" pour l'introduction du nom de fichier. Si le fichier n'existe pas, il est créé en affichant "Erreur 57: attendez".
Le programme affiche alors "CHR$(128): 1-0".
128 correspond au code du caractère.
1 correspond à la première colonne, et 0 est sa valeur.
- Les touches [>] et [<] permettent de se déplacer vers une autre colonne.
- Les touches [v] et [^] permettent de se déplacer vers un autre caractère (le numéro de colonne ne change pas).
[END LINE] duplique la colonne précédente.
- [D] déplace vers un autre caractère dont il faudra indiquer le numéro.
- [E] efface le caractère local, et déplace au suivant.
- [I] imprime le jeu de caractères tel qu'il est listé dans l'exemple joint.
- [M] modifie le code de la colonne, et déplace le "curseur" à la colonne suivante.
- [F] fin: termine le programme, et "active" le jeu secondaire, s'il n'est pas nul.
Le jeu est sauvegardé dans un fichier TEXT, d'une longueur (fixe) de 772 octets.

Heureuse Programmation.
Michel MARTINET.

==========

ROUTINES GRAPHIQUES

Afin de commencer la rédaction d'une bibliothèque complète de sous programmes, je vous propose aujourd'hui trois petits essais à vocation essentiellement graphique.
Le HP 71B peut en effet se constituer des bibliothèques de sous programmes, démontrant ainsi que son Basic n'est pas ridicule face à des langages tels que Pascal ou Fortran, pour lesquels ce genre de "bibliothèque" est chose courante. Vous êtes vivement invités à la compléter, et à le faire savoir par l'intermédiaire de JPC...

Sous-programme DTB:
L'objet est de convertir du décimal en binaire. Entrée en N, sortie en N$. Par exemple, pour convertir 255 en binaire, puis mettre le résultat en H$, faîtes CALL DTB(255,H$). Ensuite, faîtes H$ pour l'afficher.

Sous-programme PCLR et PSET:
Ils ont pour objet, respectivement, d'éteindre et d'allumer un point sur l'afficheur graphique, l'origine se situant en bas à gauche. Vivement l'assembleur !

Signalons pour les malheureux qui ne disposent pas encore du module HP-IL qu'ils peuvent remplacer la fonction BIT par une fonction utilisateur: DEFFNT(N,B)=INT(2#FP(N/2^(B+1))) voir JPC No 13 page 19, l'article de Pierre David.

Voilà, la voie est ouverte. A vous de jouer !

Au plaisir de vous lire...
Jean-Pierre BONDU.

==========

DATE

Pour tous ceux qui regrettent les fantastiques fonctions du module Horloge de la 41, et pour les autres, j'ai conçu ce sous programme qui fait l'équivalent de la fonction DOW du module. A savoir: à partir d'une date donnée, trouver le jour de la semaine. Plus facile à dire qu'à faire: cela m'a pris au moins une après-midi !

L'utilisation est des plus simples: cela se borne à un appel de sous programme ! Par exemple, pour calculer le jour correspondant au 29 juillet 1984, et le mettre dans la variable Y1:
En utilisant DATEX:
CALL DATEX (29.071984,Y1).
En utilisant DATE :
CALL DATE (29,7,1984,Y1).
Vous remarquerez que le format d'entrée de la date est européen. D'autre part, le nombre qui caractérise le jour de la semaine est semblable à la 41: 0 pour Dimanche, 1 pour Lundi, ... 6 pour Samedi. Si la date n'est pas valide, le nombre -1 est retourné.

Le principe est déjà plus complexe. Tout algorithme de calcul de date doit se rapporter à une date fixe, primitivement connue. Le reste se calcule, sachant qu'il y a un cycle de 400 ans et qu'à l'intérieur d'un siècle, il y a un cycle de 28 ans (sauf la première année si elle n'est pas bissextile). Le but de ce sous programme est de réduire le nombre de boucles, qui ralentissent l'éxécution, sans pour autant remplir la mémoire avec des données. Le compromis que j'ai trouvé me semble plutôt bon, et tant pis pour la modestie. Le programme connaît la date du premier de chaque 1500, 1600, 1700 et 1800. Le 1er Janvier 1500 est une date sans signification puisque le calendrier grégorien est valable seulement à partir du 15 Octobre 1582. Cette date est néanmoins nécessaire pour calculer les dates entre le 15 Octobre 1582 et le 31 décembre 1599.
Une fois ces quatres premiers de l'an connus, on connaît tous les premiers janvier des années séculaires jusqu'en 4000 environ. C'est la date où le calendrier grégorien devra rattraper une journée.
A partir du premier janvier de l'année séculaire, on détermine ensuite l'année modulo 28 la plus proche (00, 28, 56 ou 84). On compte alors le nombre de jours de la différence, puis le nombre de jours écoulés dans l'année.
Si vous avez suivi jusqu'ici, je vous félicite, mais ce n'est pas fini: si l'année séculaire n'est pas bissextile, il faut rajouter un jour, car le cycle n'est pas parfait. Voilà, c'est fini pour le principe. Vous pouvez décortiquer le programme en annexe.

Voilà, ce programme occupe 559 octets dans la mémoire de votre HP 71B. Vivement un module Horloge pour cette fantastique machine !

A bientôt...
Pierre DAVID.

==========

MATRICE

Après avoir acquis mon HP 71 et alors que je cherchais quelle pourrait en être son utlisation, une idée de

programme me vint. Lors de mon dernier passage à la Règle à Calcul, j'avais vu la ROM math du HP 75; une petite merveille pour ceux qui la connaissent. Une de ses grandes forces étant le traitement de matrices. Pourquoi alors ne pas donner au HP 71 ces possibilites ?

M'attelant à cette idée au matin, je n'en sortais que le soir du lendemain après une nuit quasiment blanche. Mais, je pense que le résultat en valait la peine; car parti de l'intention de copier strictement la ROM du HP 75, je me suis dit qu'il serait peut être plus avantageux de pouvoir utiliser une pile matricielle de hauteur infinie (la mémoire seule limitant celle-là) ainsi que sur notre bonne vieille 41 C.
C'est alors que sortit tout droit de mon cerveau diabolique ce programme de traitement de matrices.

Les différentes opérations utilisables sont les suivantes:

-Addition
-Multiplication
-Entrée de matrice
-Poussée de matrice dans la pile (comparable a la fonction ENTER de la HP 41 C
-Stockage de matrice
-Rappel de matrice
-Triangularisation de matrice
-Calcul du déterminant
-Entrée de la matrice identité
-Multiplication d'ume ligne par une constante
-Multiplication d'une colonne par une constante
-Addition du multiple d'une colonne a une autre colonne
-Addition du multiple d'une ligne a une autre ligne
-Multiplication de matrice par une constante
-Détermination du maximum et du minimum (en valeur absolue ou non)
-Déroulement des 3 dernières matrices entrées
-Edition d'une matrice
-Effacement de la matrice de travail
-Résolution de système
-Visualisation de la matrice de travail

Mais passons maintenant au mode d'utilisation du programme Après avoir entré le programme 'MATRICE' (ou vous l'être procuré à la programathèque) et le fichier 'MATRKEYS', faites:
RUN
Vous voyez apparaitre:
Dim. Mat(L,C)?
Entrez les dimensions de la matrice, dans l'ordre: ligne puis colonne, puis ENDLINE
A l'affichage apparait alors: (0)(1,1)=? qui signifie d'une part que la pile est vide(0) et d'autre part que le HP 71 attend l'élément de la première ligne, première colonne
Continuer alors jusqu'à ce que le HP 71 affiche OK(1), indiquant que l'opération est terminée et qu'il y a 1 matrice dans la pile
Pour visualiser celle là, appuyez sur 'V'(pour visualisation)
Le HP 71 affiche alors les résultats sous la forme:
k(I,J)=N
ou k représente le nombre d'éléments dans la matrice (lgn*cln), I et J les numéros lignes et des colonnes, et N, le nombre situé à la place I,J (à la fin le HP 71 affiche à nouveau OK(1))

Voyons maintenant sur des exemples concrets comment s'utilisent les différentes opérations

Entrons par exemple la matrice suivante:

1 3 7
9 4 -2
3 1 4
Faites donc 'C' pour effacer la matrice que vous venez d'entrer (je m'en doute un peu)
vous voyez: OK(0) puis, faites 'E' (pour entrée de matrice)
Vous voyez: (0)(1,1)=? entrez dans l'ordre: 1 puis ENDLINE
s'affiche alors: (0)(1,2)=? entrez 3 et, continuez ainsi jusqu'a la fin
en faisant 'V', vous voyez s'afficher:
(1)(1,1)=1
(1)(1,2)=3
(1)(1,3)=7
(1)(2,1)=9
(1)(2,2)=4
(1)(2,3)=-2
(1)(3,1)=3
(1)(3,2)=1
(1)(3,3)=4
pour élever cette matrice au carré, il suffit de la pousser à l'aide de la touche 'P'; puis, d'effectuer la multiplication en appuyant sur '*'.
Vous obtenez alors:
(1)(1,1)=49
(1)(1,2)=22
(1)(1,3)=29
(1)(2,1)=39
(1)(2,2)=41
(1)(2,3)=47
(1)(3,1)=24
(1)(3,2)=17

(1)(3,3)=35
puis le message OK(1)
(appelons cette matrice la matrice A)
Pour stocker cette matrice en mémoire, il suffit de taper 'S' (pour stockage)
vous voyez alors: sto No
Entrez le numéro de stockage (nous dirons 2)
faites 2 puis ENDLINE
Effacez alors la matrice A à l'aide de la touche C (pour clear); il s'affiche OK(0) (pile vide)
faites 'R' (pour rappel) vous voyez Rcl No; entrez 2 ENDLINE
s'affiche OK(1) prouvant que la matrice est à nouveau présente (visualisez la à l'aide de 'V'
Pour additioner à la matrice A la matrice identité 3*3 de la

forme
1 0 0
0 1 0
0 0 1
entrez simplement 3 au clavier puis 'I' (pour identite); vous voyez alors OK(2) qui signifie qu'il y a 2 matrices dans la pile:
-La matrice identité au sommet
-La matrice A en dessous
puis tapez '+'
La visualisation montrera que la matrice est identique à la matrice A, excepté:
(1)(1,1)=50
(1)(2,2)=42
(1)(3,3)=36
pour calculer le déterminant de cette matrice, faites 'D'. S'affiche alors: [DET]=19573
Cette opération ne modifie pas la matrice de travail

Voyons maintenant les 4 opérations sur les lignes et les colonnes
en appuyant sur 'gK', vous voyez:
Cln♯,Cln+,Cte ?
Entrez dans l'ordre:
-la colonne qui doit être multipliée
-La colonne dans laquelle aura lieu l'addition
-la constante de multiplication

Par exemple, nous allons ajouter 3 fois la colonne 2 à la colonne 1
Entrez pour cela 2,1,3 puis ENDLINE
vous voyez toujours OK(1)
appuyez sur 'gL'
le message est cette fois:
Lgn♯,Lgn+,Cte?
la syntaxe étant la même que précédement, pour ajouter 1/2 fois la ligne 3 à la ligne 2 ,tapez: 3,2,1/2, puis ENDLINE
les éléments de la matrice sont alors dans cet ordre (vérifiez à l'aide de 'V') 116, 22, 29, 202.5, 50.5, 65, 75, 17, 36
divisez maintenant la première ligne par 4, pour cela, taper 'L'
Le message est: Ligne,Cte?
entrez: 1,1/4 puis ENDLINE
puis, divisez la troisième colonne par 5 en tapant: 'K'
le message est: Colonne,Cte
entrez 3,1/5, puis ENDLINE
la matrice a alors cette forme:

(1)(1,1)=29
(1)(1,2)=5.5
(1)(1,3)=1.45
(1)(2,1)=202.5
(1)(2,2)=50.5
(1)(2,3)=13
(1)(3,1)=75
(1)(3,2)=17
(1)(3,3)=7.2
pour multiplier la matrice par une constante, tapez 'gM'
Vous voyez alors: Constante?
entrez par exemple 1/4, puis ENDLINE
A présent, pour visualiser cette matrice, nous allons utiliser la routine d'édition:
faites donc 'gE'
vous voyez: (9)(1,1)=7.25
tapez ENDLINE
(9)(1,2)=1.375
nous allons changer la valeur de l'élément 1,2; pour cela, tapez 3, puis -line , puis ENDLINE
(9)(1,3)=.3625
tapez 4 -line ENDLINE
(9)(2,1)=50.625
tapez: ›› (2 fois) puis -line puis ENDLINE
(9)(2,2)=12.625
c'est alors que vous vous apercevez que l'élément 1,1 (7.25), doit etre mis a 7
faites alors ♯1,1, -line, ENDLINE
(9)(1,1)=7.25
faites '›', -line ENDLINE
(9)(1,2)=3
puis ♯2,2 pour revenir ou vous en étiez:
(9)(2,2)=12.625
si l'édition est terminée, taper simplemnt '♯' (il n'est pas nécessaire de faire -line) puis ENDLINE
vous voyez alors s'afficher OK(1)
la matrice est alors
(1)(1,1)=7
(1)(1,2)=3
(1)(1,3)=4
(1)(2,1)=50
(9)(2,2)=12.625
(9)(2,3)=3.25
(9)(3,1)=18.75
(9)(3,2)=4.25
(9)(3,3)=1.8

Afin de trouver le minimum et le maximum de cette matrice, tapez simplement 'M' (pour minmax); s'affiche alors
Min=1.8 Max=50
il faut toutefois remarquer que cette commande a une variante: en effet si la matrice contient des nombres négatifs et que le drapeau 3 est baissè, ceux là sont considérés signé, si, au contraire, le drapeau 3 est levé, il sont considérés en valeurs absolues et apparaissent non signés dans le resultat.
si vous desirez maintenant triangulariser cette matrice, tapez 'T' (pour triangularisation)
vous voyez alors OK(1), toujours une seule matrice dans la pile
les valeurs de celle la sont d'ailleur les suivantes
(1)(1,1)=-13.6521739135
(1)(1,2)=-0.00000000001
(1)(1,3)=1.301542771 E-11
(1)(2,1)=16.1458333331
(1)(2,2)=4.95138888887
(1)(2,3)=-0.00000000001
(1)(3,1)=18.75
(1)(3,2)=4.25
(1)(3,3)= 1.8
Mais comme vous pouvez le constater, ces résultats ne sont pas optimisés; en effet, en (1,2), (1,3), et (2,3), les valeurs sont en fait '0'. tapez alors 'O' (pour optimisation) et en 1,2; 1,3; et 2,3 vous trouverez '0'
cette fonction permet en effet d'amener a '0' tous les nombres compris entre -1E-8 et et +1e-8, et a 1 ceux compris entre 1-E-8 et 1+E-8 (utile dans le cas de résolution de système).
On pourra noter que cette routine est utilisée pour le calcul du déterminant et la ré solution de système

Afin de bien montrer la souplesse d'utilisation de ce programme, nous allons un peu laisser tomber cette matrice et nous occuper d'autres.
Pour cela, appuyez sur 'E' et entrez cette matrice
1 2 4 2
6 -3 1 -1

puis cette seconde
1 4 9 7
2 1 -4 1
-3 2 -6 3
et enfin cette troisième
1 2
2 5

3 1
Vous devez obtenir enfin: OK(4) qui vous déclare que la pile contient 4 éléments

Si vous essayez d'éxécuter '+' ou '*', vous obtenez un message d'erreur, dont la liste sera donnée plus loin
En effet, on ne peut ni additioner, ni multiplier entre elles une matrice 3*4 et une matrice 3*2. Pour qu'une opération soit possible, il faut rappeler la première matrice entrée (de dimension 2*4). Cela est rendu possible par la fonction RDN qui agit sur les 3 matrices situées au sommet de la pile
En appuyant 2 fois sur 'R' vous obtenez le message OK(4). Si vous appuyez maintenant sur '*', l'opération a lieu, et la réponse est OK(3). Vous pouvez alors réaliser l'addition des 2 matrices 3*3 en appuyant sur '+' on lit alors que la pile ne contient plus que 2 matrices. On va d'ailleur conserver la matrice que l'on vient d'obtenir en se débarrassant de celle qui nous a rendu de bons services tout à l'heure
Pour cela, tapez: 'R' puis 'C' ce qui a pour effet de faire tourner les matrices (R), pui s d'effacer la matrice de travail et de faire monter la pile.
On peut alors écrire la matrice que l'on a obtenu précédemment sous la forme d'un systeme de ce type:
14X + 0Y + 15Z = 7
34X - 10Y + 9Z = 0
6X + 5Y + 7Z = 8
a l'aide de la touche 'gS'(pour résolution de systeme), on peut calculer X, Y, et Z. Après avoir appuyé sur 'gS', vous obtenez le message 'en cours de résolution', puis le message OK(1) qui signifie que le système est résolu. La matrice a alors le format suivant:
1 0 0 .214673913047
0 1 0 .969565217398
0 0 1 .26630434782
après avoir bien entendu utilisé la routine d'optimisation, ces résultats nous donnent les valeurs de X, Y,et Z

Il existe enfin 2 routines qui permettent de vider la mémoire en fin d'utilisation en mode utilisation matricielle
ces 2 routines sont assignées aux touches 'gD' et 'gR'
Celle assignée à 'd' permet d'effacer les matrices qui ont été stockées,
celle assignée à 'r' permet de restaurer les assignations précédentes

Comme annoncé tout a l'heure voici les quelques messages d'erreur que peut retourner le HP 71:
'pile vide': la pile ne contient aucun élément
'mémoire vide': on a tenté de rappeler une mémoire qui n'a pas été remplie precedemment
'dif de dimension': on a tenté d'effectuer des opérations sur des matrices dont les dimensions sont différentes
Lgn*Col': Idem dif de dimension

'colonne inexistante': on a tenté une opération sur un numéro de colonne trop grand
'Ligne inexistante': on a tenté une opération sur un numéro de ligne trop grand
'un seul élément': on a tenté une opération diadique alors qu'un seul élément est présent dans la pile
'op. non valide': L'opération spécifiée ne peut être effectuée.

Laurent ISTRIA (*3)

==========

DESSIL

Ce programme permet de faire du DAO (Dessin Assisté par Ordinateur). Vous pourrez dessiner sur l'afficheur de votre Ordinateur Favori, avec les quatre touches de curseur, et quelques autres...

Variables:
X= No de colonne
Y= No de ligne
C= Valeur de la colonne courante

A$= Sauvegarde de l'affichage
B$= Idem si mode DEPlacement

Drapeaux:
6: (1)mode TRACE
6: (0)mode GOMME
4: (1)mode DEPlacement
0: (1)mode MOVE

Commandes:
<,>,^,v: Déplacement du curseur

T: (T)race le chemin du curseur

G: (G)omme le chemin du curseur

P: (P)ositionne le curseur à une adresse absolue
D: (D)éplacement du curseur -Bascule-
I: (I)nverse vidéo !...
B: (B)ouge le dessin en bloc
C: (C)odes des caractères dessinés

Mode d'emploi:
Le curseur figure la pointe d'un crayon.
En mode TRACE, -celui dans lequel vous êtes lorsque vous lancez le programme-, le chemin parcouru par le crayon reste à l'affichage.
En mode GOMME, le curseur figure une gomme (!) ... devinez la suite.

En mode DEPLACEMENT, le curseur apparaîtra en vidéo inverse sans modifier le dessin à l'affichage.
INVERSE vidéo: appuyez sur la touche <I> et votre chef d'oeuvre connaîtra l'inversion vidéo.
POSITION du curseur: appuyez sur <P> et vous pourrez spécifier la position à laquelle le curseur doit se placer (coordonnées cartésiennes (X,Y)). Avec: 0<X<131 et 0<Y<7
BOUGE: déplace le dessin suivant les 4 flèches habituelles. Touche <B> Ce qui sort de l'affichage est perdu.
CODES: touche <C> Permet de (V)isualiser ou (S)auver le dessin. Ce programme est la version 3.0 , et permet de faire des choses intéressantes sur le HP 71.

Heureuse programmation...

Jean-Pierre BONDU (#33).

==========

EDITEUR DE CASSETTE

Le programme que je vous propose modifie et écrit toutes sortes d'informations sur les cassettes numériques du lecteur HP82161.

Rappels (extraits du manuel)

Le lecteur comporte trois caratéristiques dont la connaissance facilite l'utilisation:
mémoire tampon 0, mémoire tampon 1 et le pointeur d'octet.

Mémoire tampon 0. Cette mémoire comporte 256 octets et contient temporairement les informations transférées entre la bande et la boucle HPIL.

Mémoire tampon 1. Cette mémoire comporte 256 octets et contient temporairement les informations envoyées ou reçues de la boucle d'interface.

Pointeur d'octet. Ce pointeur est un indicateur dirigeant le transfert d'informations dans et hors des mémoires tampon 0 et 1. Il "pointe" à l'un des octets de la mémoire (octets de 0 à 255). Le transfert suit en général le pointeur qui se déplace d'un octet vers le suivant. Le pointeur agit dans les deux mémoires à la fois.

Le lecteur de casette dispose de deux modes principaux pour l'écriture d'informations sur la bande: continue et partielle. L'écriture continue est utilisée par l'ordre DDL2 et l'écriture partielle par l'ordre DDL6.

Lors d'une écriture partielle (utilisée par le programme), l'enregistrement courant de la bande est copié dans la mémoire tampon 0 et une partie (ou l'ensemble dans le programme) de ces informations est remplacée par les données en provenance de la boucle d'interface. Lorsque la mémoire est pleine, son nouveau contenu remplace l'ancien dans l'enregistrement sur la bande et l'enregistrement suivant est copié dans la mémoire tampon 0. Cette méthode permet de modifier une partie d'un enregistrement sans affecter le reste.

La cassette est structurée en 2 pistes de 256 enregistrements chacune et chaque enregistrement en 256 octets. 256*256*2 = 131072 octets.

le programme numérote les enregistrements de 0 à 511 et les octets de 0 à 255.

Le programme travaillera toujours avec le premier lecteur de cassettes dela boucle.

UTILISATION:

Une fois le programme lancé, vous disposez de plusieurs commandes:

- [v], écrit sur la cassette les modifications faites sur l'enregistrement courant et déplace le pointeur à l'enregistrement suivant. A l'enregistrement 511, le pointeur revient à l'enregistrement 0. Cette opération n'affecte pas la position du pointeur d'octet.

- [^], écrit sur la cassette les modifications faites sur l'enregistrement courant et déplace le pointeur à l'enregistrement précédent. A l'enregistrement 0, le pointeur va à l'enregistrement 511. Cette opération n'affecte pas le pointeur d'octet.

- [>], déplace le pointeur d'octet d'une position en avant. A l'octet 255, le pointeur revient à l'octet 0 du même enregistrement.

- [<], déplace le pointeur d'octet d'une position en arrière. A l'octet 0, le pointeur va à l'octet 255 du même enregistrement.

- [E], écrit sur la cassette les modifications faites sur l'enregistrement courant et déplace le pointeur d'enregistrement à l'adresse indiquée.

- [O], déplace le pointeur d'octet à l'adresse indiquée.

- [C], modifie dans la mémoire du HP 71 l'octet courant. Les modifications ne sont écrites sur la cassette qu'avec les fonctions [v], [^], [E] et [F].

- [R], remplace les caractères en partant de la position du pointeur jusqu'à la fin de l'enregistrement par des caractères 255.

- [F], fin de programme: écrit sur la cassette les modifications faites sur l'enregistrement courant et rembobine la cassette.

- [ATTN/ON], arrête le programme sans écrire l'enregistrement courant.

Heureuse programmation.
Michel MARTINET.

==========

GESTION DE L'HPIL

Ce programme a pour but de gérer une boucle HP-IL comportant un nombre quelconque de HP 71B, une ou plusieurs unités de mémoire de masse, une ou plusieurs imprimantes, un ou plusieurs modems... Il faut prévoir grand, n'est-ce pas ? Ne désespérez pas: tout ce matériel pourra être réuni au cours des réunions mensuelles de PPC-PARIS, où vous pourrez, bien sûr, amener le vôtre.
Nanti de ce programme, il sera possible de travailler, chacun son tour, sur la boucle. Ceci s'appelle le "partage de ressources".
Un des HP 71 disposera de ce programme: c'est le contrôleur général. Les autres, pour disposer de la boucle, demanderont le contrôle, effectueront leur tâche, puis repasseront la main au contrôleur général. Celui-ci consultera alors une liste d'attente pour connaître le suivant.

Utilisation:
Il faut charger le programme dans la mémoire du HP 71 contrôleur général, puis le lancer par [RUN]. Il prépare alors tous les HP 71 de la boucle, puis en affiche le nombre.
Sont ensuite affichés en permanence:
- Soit le nombre de 71 demandeurs, ainsi que leurs numéros.
- Soit le numéro du 71 qui dispose du contrôle, le cas échéant.
Vous pouvez alors, sur un des HP 71 non contrôleur, demander le contrôle en appuyant sur [g] [ON]. Un premier signal sonore vous informe que votre demande est enregistrée, un deuxième vous prévient quand vous aurez le contrôle.
Attention: il faut obligatoirement rendre le contrôle en appuyant sur [g] [ON], une fois votre tâche accomplie.
Pour arrêter le programme du contrôleur général, il suffit d'appuyer sur [g] [ON].

Attention si vous étudiez le fonctionnement de ce programme: il ne faut pas confondre numéro et adresse de HP 71. L'adresse est l'adresse normale, telle qu'elle est décrite dans le manuel d'utilisation. Le numéro est le numéro d'ordre du HP 71 dans la boucle. Par exemple, 3ème HP 71 peut être différent de HP 71 d'adresse 3.

A bientôt...
Pierre DAVID

JEU DE CARACTERES SECONDAIRES : HP82905B

CHR$(128)		000	000	000	000	000	000	CHR$(192)	à	032	086	085	086	120	000
CHR$(129)		000	000	000	000	000	000	CHR$(193)	è	056	086	085	086	024	000
CHR$(130)		000	000	000	000	000	000	CHR$(194)	ò	056	070	069	070	056	000
CHR$(131)		000	000	000	000	000	000	CHR$(195)	ù	056	066	065	066	120	000
CHR$(132)		000	000	000	000	000	000	CHR$(196)		000	000	000	000	000	000
CHR$(133)		000	000	000	000	000	000	CHR$(197)	é	056	084	086	085	024	000
CHR$(134)		000	000	000	000	000	000	CHR$(198)		000	000	000	000	000	000
CHR$(135)		000	000	000	000	000	000	CHR$(199)		000	000	000	000	000	000
CHR$(136)		000	000	000	000	000	000	CHR$(200)	à	032	084	085	086	120	000
CHR$(137)		000	000	000	000	000	000	CHR$(201)	è	056	085	086	084	024	000
CHR$(138)		000	000	000	000	000	000	CHR$(202)		000	000	000	000	000	000
CHR$(139)		000	000	000	000	000	000	CHR$(203)	ù	060	065	066	064	124	000
CHR$(140)		000	000	000	000	000	000	CHR$(204)		000	000	000	000	000	000
CHR$(141)		000	000	000	000	000	000	CHR$(205)	ë	056	085	084	085	024	000
CHR$(142)		000	000	000	000	000	000	CHR$(206)		000	000	000	000	000	000
CHR$(143)		000	000	000	000	000	000	CHR$(207)	ü	060	065	064	065	124	000
CHR$(144)		000	000	000	000	000	000	CHR$(208)		000	000	000	000	000	000
CHR$(145)		000	000	000	000	000	000	CHR$(209)	î	000	074	121	066	000	000
CHR$(146)		000	000	000	000	000	000	CHR$(210)	ø	062	081	073	069	062	000
CHR$(147)		000	000	000	000	000	000	CHR$(211)	£	124	010	127	073	073	000
CHR$(148)		000	000	000	000	000	000	CHR$(212)		000	000	000	000	000	000
CHR$(149)		000	000	000	000	000	000	CHR$(213)		000	000	000	000	000	000
CHR$(150)		000	000	000	000	000	000	CHR$(214)		000	000	000	000	000	000
CHR$(151)		000	000	000	000	000	000	CHR$(215)		000	000	000	000	000	000
CHR$(152)		000	000	000	000	000	000	CHR$(216)		000	000	000	000	000	000
CHR$(153)		000	000	000	000	000	000	CHR$(217)		000	000	000	000	000	000
CHR$(154)		000	000	000	000	000	000	CHR$(218)		000	000	000	000	000	000
CHR$(155)		000	000	000	000	000	000	CHR$(219)		000	000	000	000	000	000
CHR$(156)		000	000	000	000	000	000	CHR$(220)	é	124	084	086	085	084	000
CHR$(157)		000	000	000	000	000	000	CHR$(221)	ï	000	069	124	065	000	000
CHR$(158)		000	000	000	000	000	000	CHR$(222)		000	000	000	000	000	000
CHR$(159)		000	000	000	000	000	000	CHR$(223)		000	000	000	000	000	000
CHR$(160)		000	000	000	000	000	000	CHR$(224)		000	000	000	000	000	000
CHR$(161)		000	000	000	000	000	000	CHR$(225)		000	000	000	000	000	000
CHR$(162)		000	000	000	000	000	000	CHR$(226)		000	000	000	000	000	000
CHR$(163)		000	000	000	000	000	000	CHR$(227)		000	000	000	000	000	000
CHR$(164)		000	000	000	000	000	000	CHR$(228)		000	000	000	000	000	000
CHR$(165)		000	000	000	000	000	000	CHR$(229)		000	000	000	000	000	000
CHR$(166)		000	000	000	000	000	000	CHR$(230)		000	000	000	000	000	000
CHR$(167)		000	000	000	000	000	000	CHR$(231)		000	000	000	000	000	000
CHR$(168)		000	000	000	000	000	000	CHR$(232)		000	000	000	000	000	000
CHR$(169)		000	000	000	000	000	000	CHR$(233)		000	000	000	000	000	000
CHR$(170)		000	000	000	000	000	000	CHR$(234)		000	000	000	000	000	000
CHR$(171)		000	000	000	000	000	000	CHR$(235)		000	000	000	000	000	000
CHR$(172)		000	000	000	000	000	000	CHR$(236)		000	000	000	000	000	000
CHR$(173)		000	000	000	000	000	000	CHR$(237)		000	000	000	000	000	000
CHR$(174)		000	000	000	000	000	000	CHR$(238)		000	000	000	000	000	000
CHR$(175)		000	000	000	000	000	000	CHR$(239)		000	000	000	000	000	000
CHR$(176)		000	000	000	000	000	000	CHR$(240)		000	000	000	000	000	000
CHR$(177)		000	000	000	000	000	000	CHR$(241)		000	000	000	000	000	000
CHR$(178)		000	000	000	000	000	000	CHR$(242)		000	000	000	000	000	000
CHR$(179)	▪	000	000	007	005	007	000	CHR$(243)		000	000	000	000	000	000
CHR$(180)		000	000	000	000	000	000	CHR$(244)		000	000	000	000	000	000
CHR$(181)	ç	056	068	196	068	068	000	CHR$(245)		000	000	000	000	000	000
CHR$(182)		000	000	000	000	000	000	CHR$(246)		000	000	000	000	000	000
CHR$(183)		000	000	000	000	000	000	CHR$(247)		000	000	000	000	000	000
CHR$(184)		000	000	000	000	000	000	CHR$(248)		000	000	000	000	000	000
CHR$(185)		000	000	000	000	000	000	CHR$(249)		000	000	000	000	000	000
CHR$(186)		000	000	000	000	000	000	CHR$(250)		000	000	000	000	000	000
CHR$(187)	£	072	126	073	065	066	000	CHR$(251)		000	000	000	000	000	000
CHR$(188)		000	000	000	000	000	000	CHR$(252)		000	000	000	000	000	000
CHR$(189)	§	032	078	085	057	002	000	CHR$(253)		000	000	000	000	000	000
CHR$(190)		000	000	000	000	000	000	CHR$(254)		000	000	000	000	000	000
CHR$(191)		000	000	000	000	000	000	CHR$(255)		000	000	000	000	000	000

Programme "POLYCOPY" (transmission simultanée de fichiers)

```
1000 INTEGER I,J,N
   - I: Compteur
     J: Brouillon (adresse du i-ème HP 71)
     N: Nombre de HP 71 dans la boucle (non compris l'émetteur).
1010 DIM F$[8]
   - F$: Nom du fichier.
1020 GOSUB 'RESTAURE'
   - Prépare la boucle.
1030 DISP 'Il y a';N;'HP 71.'
1040 BEEP
1050 IF N=0 THEN STOP
   - On arrête si il n'y a pas d'autre HP 71 dans la boucle...
1060 ON ERROR GOTO 'ERREUR'

================================================================
1070 'BOUCLE':
1080 INPUT 'Fichier: ';F$
1090 SEND IFC
   - IFC: InterFace Clear, supprime tous les états émetteurs ou récepteurs sur la boucle.
1100 IF F$='' THEN 'FIN'
1110 FOR I=1 TO N
1120 SEND LISTEN A(I)
   - Déclare le i-ème HP 71 comme récepteur (le prépare à recevoir le fichier).
1130 NEXT I
1140 SEND MTA
   - MTA: My Talk Adress, prépare le HP 71 émetteur à envoyer le fichier.
1150 OUTPUT :LOOP ;'COPY:LOOP TO'&F$
   - Ordonne à tous les 71 de réceptionner le programme.
1160 COPY F$ TO :LOOP
   - L'envoi proprement dit...
1170 GOTO 'BOUCLE'
   - On recommence.

================================================================
1180 'RESTAURE':
1190 RESET HPIL
   - Redispose correctement le module HP-IL: utilisé surtout après une condition d'erreur.
1200 RESTORE IO
   - assigne la boucle.

     Le tableau A contient les adresses de tous les HP 71 sur l'HP-IL. Il est dimensionné
     exactement à N au sortir de la boucle suivante:
1210 FOR I=1 TO INF
1220 J=DEVADDR('HP71('&STR$(I)&')')
   - Adresse du i-ème HP 71.
1230 IF J<0 THEN 1280
   - Si elle est négative, le i-ème HP 71 n'existe pas.
1240 SHORT A(I)
   - Redimensionnement de tableau: merci Hewlett !
1250 A(I)=J
1260 REMOTE :J
   - Place le HP 71 en mode télécommande: toutes les données envoyées seront interprétées
     comme des commandes Basic.
1270 NEXT I
1280 N=I-1
   - Nombre de HP 71 dans la boucle.
```

```
1290 RETURN

============================================================
1300 'ERREUR':
1310 FOR I=1 TO 20 @ OUTPUT :LOOP ;'' @ NEXT I
```
- Si il y a eû erreur, les 71 récepteurs restent bloqués sur l'ordre COPY. On envoie alors une série de commandes "nulles" pour débloquer la situation. Le nombre 20 peut dépendre des HP 71. A modifier éventuellement.

```
1320 DISP ERRM$
```
- On affiche le message d'erreur après avoir débloqué la situation, car :LOOP ne peut plus fonctionner correctement si un périphérique de visualisation (DISPLAY IS) est déclaré (cf page 186 du manuel du HP IL).

```
1330 BEEP
1340 GOTO 'BOUCLE'
```
- On remet la boucle en place.

```
============================================================
1350 'FIN':
1360 OFF ERROR
1370 FOR I=1 TO N
1380 LOCAL :A(I)
1390 NEXT I
1400 END
```
- On replace la boucle en mode LOCAL (mode normal).

****

Programme "CHARSET" (gestion du jeu de caractères secondaire)

```
1000 DIM A$[1],C$[768],D$[24],E$[2],H$[2],K$[4],N$[8]
```
- A$: caractère secondaire "courant".
  C$: CHARSET courant.
  D$: Variable d'affichage avec champ protégé (cf page 237 du manuel d'utilisation).
  E$, H$: délimiteurs de champs.
  K$: Valeur de la touche pressée.
  N$: Nom du fichier (non détruit en cas de réutilisation).

```
1010 INTEGER A,B,C,I
```
- A: Adresse de la colonne courante dans c$ (de 0 à 767)
  B: Numéro de la colonne courante (de 1 à 6)
  C: Numéro de caractère (de 128 à 255).
  I: Comme d'habitude...

```
1020 A=0
1030 E$=CHR$(27)&'('
1040 H$=CHR$(27)&')'
1050 DELAY 0,0
1060 LC OFF
1070 GOSUB 'ASSIGNE'

============================================================
1500 'RETOUR':
1510 GOSUB 'AFFICHE'

============================================================
1520 'BOUCLE':
1530 K$=KEY$ @ IF K$='' THEN 'BOUCLE'
1540 IF K$='#38' THEN GOSUB 'DUPLIC'
1550 IF K$='#47' THEN A=MOD(A-1,768)
1560 IF K$='#48' THEN A=MOD(A+1,768)
```

```
1570 IF K$='#50' THEN A=MOD(A-6,768)
1580 IF K$='#51' THEN A=MOD(A+6,768)
1590 IF K$='D' THEN GOSUB 'DEPLACE'
1600 IF K$='E' THEN GOSUB 'EFFACE'
1610 IF K$='I' THEN GOSUB 'IMPRIME'
1620 IF K$='M' THEN GOSUB 'MODIF'
1630 IF K$='F' THEN 'FIN' ELSE 'RETOUR'

================================================================
2000 'FIN':
2010 DISP 'Sauvegarde: attendez'
2020 FOR I=1 TO 768
2030 IF C$[I,I]<>CHR$(0) THEN 2070
2040 NEXT I
2050 PURGE N$
2060 GOTO 2100
2070 RESTORE #1
2080 PRINT #1;C$
2090 SECURE N$
2100 DESTROY C$
2110 PUT '#38'
   - Permet de récupérer le curseur après un DISP et une fin de programme.
2120 END

================================================================
2500 'ASSIGNE':
2510 ON ERROR GOTO 2650
2520 INPUT 'Fichier :',N$;N$
2530 D$=ADDR$(N$)
2540 UNSECURE N$
2550 ASSIGN #1 TO N$
2560 READ #1;C$
2570 GOTO 2620
2580 DISP 'Erreur '&STR$(ERRN)&': attendez'
2590 FOR I=1 TO 768
2600 C$[I,I]=CHR$(0)
2610 NEXT I
2620 CHARSET C$
2630 OFF ERROR
2640 RETURN
2650 BEEP 2000
2660 IF ERRN=24 OR ERRN=1024 THEN 2710
   - Mémoire insuffisante.
2670 IF ERRN=54 OR ERRN=1054 THEN 2580
   - Fin de fichier (il existe, mais vide).
2680 IF ERRN=57 OR ERRN=1057 THEN CREATE TEXT N$ @ ASSIGN #1 TO N$ @ GOTO 258(
   - Fichier non trouvé.
2690 IF ERRN=64 OR ERRN=1064 THEN 2580
   - Périphérique non trouvé.
2700 DISP ERRM$ @ WAIT 1 @ GOTO 2520
2710 POP
2720 DISP ERRM$
2730 END

================================================================
3000 'AFFICHE':
3010 C=128+A DIV 6
3020 B=1+MOD(A,6)
3030 A$=CHR$(C)
3040 IF A$='' THEN A$=' '
3050 D$='CHR$('&STR$(C)&'): '&A$&' '&STR$(B)&'-'&STR$(NUM(C$[A+1,A+1]))
```

```
3060 BEEP INF,EPS
3070 DISP D$
3080 RETURN

================================================================
3500 'DEPLACE':
3510 D$=E$&'CHR$('&H$&STR$(C)&E$&'): '&A$&' '&STR$(B)&'-'&STR$(NUM(C$[A+1,A+1]))
   - Affichage avec champs protégés.
3520 ON ERROR GOTO 3530
3530 BEEP 2000,.1
3540 INPUT '',D$;C
3550 IF C<128 OR C>255 THEN 3530
3560 OFF ERROR
3570 A=(C-128)%6
3580 RETURN

================================================================
4000 'EFFACE':
4010 A=(C-128)%6
4020 FOR I=1 TO 6
4030 C$[A+I,A+I]=CHR$(0)
4040 NEXT I
4050 A=MOD(A+6,768)
4060 CHARSET C$
4070 RETURN

================================================================
4500 'MODIF':
4510 D$=E$&'CHR$('&STR$(C)&'): '&A$&' '&STR$(B)&'-'&H$&STR$(NUM(C$[A+1,A+1]))
   - Affichage avec champs protégés.
4520 ON ERROR GOTO 4530
4530 BEEP 2000,.1
4540 INPUT '',D$;B
4550 IF B<0 OR B>255 THEN 4530
4560 OFF ERROR
4570 C$[A+1,A+1]=CHR$(B)
4580 A=MOD(A+1,768)
4590 CHARSET C$
4600 RETURN

================================================================
5000 'DUPLIC':
5010 IF A=0 THEN RETURN
5020 IF A=767 THEN C$[768]=C$[767,767] @ GOTO 5050
5030 C$[MOD(A+1,768),MOD(A+1,768)]=C$[A,A]
5040 A=MOD(A+1,768)
5050 CHARSET C$
5060 RETURN

================================================================
5500 'IMPRIME':
5510 ON ERROR GOTO 5930
5520 RESTORE IO
5530 Q=DEVADDR('HP82905B')
   - Code à changer si votre imprimante est différente.
5540 IF Q=-1 THEN RETURN
5550 PRINT CHR$(27)&'&l96f8D'&CHR$(27)&'&k2S'
   - Eventuellement à changer.
5560 OPTION BASE 1
5570 INTEGER H,J,K,P(6),Q
   - H: Compteur de lignes d'impression.
```

```
     J: Compteur du tableau P.
     K: inversion de la matrice de caractères pour le graphique de l'imprimante HP82905B.
     P: Valeur des 6 colonnes du caractère.
     Q: Valeur de la colonne à imprimer.
5580 IMAGE #,A,X,'CHR$(',3Z,')',X,A,X,12A,X,A,6(X,3Z),X,A
5590 GOSUB 5830
5600 PRINT TAB(20);'¦';
5610 PRINT TAB(40);'JEU DE CARACTERES SECONDAIRES : '&N$;
5620 PRINT TAB(107);'¦'
5630 GOSUB 5830
5640 FOR H=0 TO 63
5650 PRINT TAB(20);
5660 FOR I=H TO H+64 STEP 64
5670 P$=CHR$(27)&'*b6G'
5680 FOR J=1 TO 6
5690 P(J)=NUM(C$[I*6+J])
5700 Q=0
5710 FOR K=0 TO 7
5720 IF BIT(P(J),K) THEN Q=Q+2^(7-K)
5730 NEXT K
5740 P$[5+J]=CHR$(Q)
5750 NEXT J
5760 PRINT USING 5580;'¦';I+128;'¦';P$;'¦';P(1);P(2);P(3);P(4);P(5);P(6);'¦'
5770 NEXT I
5780 PRINT
5790 NEXT H
5800 GOSUB 5830
5810 PRINT CHR$(12)
5820 RETURN
5830 PRINT TAB(21);
5840 FOR I=1 TO 86
5850 PRINT '-';
5860 NEXT I
5910 PRINT
5920 RETURN
5930 BEEP 2000
5940 DISP ERRM$
5950 OFF ERROR
5960 RETURN
```

##

Routines graphiques

```
================================================================
1000 SUB DTB(N,N$)
1010 N$=''
1020 FOR J=7 TO 0 STEP -1
1030 N$=N$&STR$(BIT(N,J))
1040 NEXT J
1050 END SUB

================================================================
2000 SUB PCLR(X,Y)
2010 I=0
2020 GOTO 3020
```

```
================================================================
3000 SUB PSET(X,Y)
3010 I=1
3020 IF X<1 OR X>132 OR Y<1 OR Y>8 THEN 3100
3030 DIM A$[132]
3040 A$=GDISP$
3050 C=NUM(A$[X]).
3060 IF I AND NOT BIT(C,8-Y) THEN C=C+2^(8-Y)
3070 IF NOT I AND BIT(C,8-Y) THEN C=C-2^(8-Y)
3080 A$[X,X]=CHR$(C)
3090 GDISP A$
3100 END SUB
```

##

Programme "DATE" (calcul du jour de la semaine)

```
================================================================
1000 SUB DATEX(X,D)
1010 J=INT(X)
1020 X=FP(X)*100
1030 M=INT(X)
1040 A=FP(X)*10000
1050 CALL DATE(J,M,A,D)
1060 END SUB
  - Sans commentaires...

================================================================
2000 SUB DATE(J,M,A,D)
  - J: Jour, M: Mois, A: Année, D est la réponse.
2010 OPTION BASE 1
2020 INTEGER N(12),P(4),A1,I,K,P1
  - N: contient le nombre de jours de chaque mois de l'année.
    P: est la valeur du premier de l'an "1500", 1600, 1700 et 1800.
    I: Usage habituel
    K: Siècle correspondant au tableau P
2030 FOR I=1 TO 12
2040 READ N(I)
2050 NEXT I
  - Remplissage du tableau N.
2060 FOR I=1 TO 4
2070 READ P(I)
2080 NEXT I
  - Remplissage du tableau P.
2090 N(2)=28+FNB(A)
  - Février comprend 28 ou 29 jours.
2100 D=-1
  - D vaut -1 si il y a erreur,
2110 IF M>12 OR M<1 THEN 2310
  - comme un mois inexistant,
2120 IF J<1 OR J>N(M) THEN 2310
  - un jour non valide,
2130 IF A*10000+M*100+J<15821015 THEN 2310
  - ou une date antérieure au 15 Octobre 1582.
2140 K=MOD((A-1500) DIV 100,4)+1
  - K vaut 1 si 1500-1599, 1900-1999, 2300-2399, etc...
          2 si 1600-1699, 2000-2099,
```

```
          3 si 1700-1799, 2100-2199,
          4 si 1800-1899, 2200-2299, etc...
2150 D=P(K)
   - D est initialisée au premier jour de l'année séculaire.
2160 FOR I=A-MOD(MOD(A,100),28) TO A-1
2170 D=D+365+FNB(I)
   - Dans l'année I, il y a 365 jours, ou 366 si elle est bissextile.
2180 NEXT I
2190 FOR I=1 TO M-1
2200 D=D+N(I)
   - N(I) est le nombre de jours dans le mois I.
2210 NEXT I
2220 IF K#2 AND MOD(A,100) THEN D=D-1
   - Si l'année séculaire n'est pas bissextile, on ajoute un jour.
2230 D=MOD(D+J-1,7)
   - On ramène le jour entre 0 et 6.

2240 DATA 31,0,31,30,31,30,31,31,30,31,30,31
   - Données pour le tableau N. Pour Février, il n'est pas nécessaire de mettre une valeur,
     car le mois de Février est initialisé ligne 2090. Gain de 1 quartet, toujours bon à
     prendre. Il n'y a pas de petits profits !
2250 DATA 1,6,5,3
   - Le "1er Janvier 1500" est un Lundi, etc...

================================================================
2260 DEF FNB(A)
   - FNB retourne 1 si l'année est bissextile, 0 si elle ne l'est pas.
2270 FNB=0
2280 IF NOT MOD(A,4) THEN FNB=1
2290 IF NOT MOD(A,100) AND MOD(A,400) THEN FNB=0
2300 END DEF

================================================================
2310 END SUB
```

****

Programme "MATRICE" (traitement de matrices)

```
0010 RENAME KEYS TO MATRK @ RENAME MATRKEYS TO KEYS
0020 DESTROY A,B,C,C1,D,L1,L2,K1,K2,M,M1,M2,N,O,P,P1,R$,S$ @ OPTION BASE 1 @ DELAY .5,.5
0030 ASSIGN #1 TO * @ ASSIGN #2 TO *

================================================================
0040 'ENTMAT': INPUT 'Dim. mat.(L,C)? ';M,N
0050 DIM A(M,N)
0060 FOR I=1 TO M @ FOR J=1 TO N
0070 DISP USING "#,'('(',K,')('(',K,',',K,')= '";P1,I,J
0080 INPUT A(I,J)
0090 NEXT J @ NEXT I
0100 P1=P1+1
0110 ASSIGN #1 TO 'MAT'&STR$(P1)
0120 PRINT #1;M,N,A(,)
0130 ASSIGN #1 TO *
0140 DISP 'OK(';P1;')'
0150 END

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

```
0160 'POUMAT': IF P1=0 THEN DISP 'Pile vide !' @ BEEP @ END
0170 COPY 'MAT'&STR$(P1) TO 'MAT'&STR$(P1+1)
0180 ASSIGN #1 TO 'MAT'&STR$(P1)
0190 READ #1;M,N,A(,)
0200 ASSIGN #1 TO *
0210 P1=P1+1
0220 DISP 'OK(';P1;')'
0230 END

================================================================
0240 'CLMAT': PURGE 'MAT'&STR$(P1) @ DESTROY A
0250 IF P1=1 THEN 310
0260 ASSIGN #1 TO 'MAT'&STR$(P1-1)
0270 READ #1;M,N
0280 DIM A(M,N)
0290 READ #1;A(,)
0300 ASSIGN #1 TO *
0310 P1=P1-1
0320 DISP 'OK(';P1;')'
0330 END

================================================================
0340 'DESTMAT': FOR I=1 TO P1 @ PURGE 'MAT'&STR$(I) @ NEXT I @ END

================================================================
0350 'ADDMAT': IF P1=0 THEN DISP 'Pile vide !' @ BEEP @ END
0360 GOSUB 'AM'
0370 IF M#O OR N#P THEN DISP 'Dif. de dimension !' @ BEEP ELSE 390
0380 ASSIGN #1 TO * @ ASSIGN #2 TO * @ END
0390 DIM A(O,P),B(M,N)
0400 READ #2;A(,) @ READ #1;B(,)
0410 FOR I=1 TO M @ FOR J=1 TO N
0420 A(I,J)=A(I,J)+B(I,J)
0430 NEXT J @ NEXT I
0440 RESTORE #2
0450 PRINT #2;M,N,A(,)
0460 ASSIGN #1 TO * @ ASSIGN #2 TO *
0470 GOTO 720

================================================================
0480 'SMAT': IF P1=0 THEN DISP ' pile vide !' @ BEEP @ END
0490 FOR I=1 TO M @ FOR J=1 TO N
0500 DISP USING "K,'(',K,',',K,')= ',K";M#N,I,J,A(I,J)
0510 NEXT J @ NEXT I
0520 DISP 'OK(';P1;')'
0530 END

================================================================
0540 'MINMAX': M1=INF @ M2=-INF
0550 FOR I=1 TO M @ FOR J=1 TO N
0560 IF NOT FLAG(3) THEN M1=MIN(M1,A(I,J)) ELSE M1=MIN(M1,ABS(A(I,J)))
0570 IF NOT FLAG(3) THEN M2=MAX(M2,A(I,J)) ELSE M2=MAX(M2,ABS(A(I,J)))
0580 NEXT J @ NEXT I
0590 DISP 'Min=';M1;' Max=';M2
0600 END

================================================================
0610 'MULTMAT': IF P1=0 THEN DISP 'Pile vide !' @ BEEP @ END
0620 GOSUB 'AM'
0630 IF P#M THEN DISP 'Lgn#Col !' @ BEEP @ END
0640 DIM B(O,P),C(M,N),A(O,N)
```

```
ASSIGNATIONS DU PROGRAMME MATRICE


DEF KEY 'E','GOSUB ENTMAT':
DEF KEY 'R','GOSUB RAPMAT':
DEF KEY 'T','GOSUB TRIMAT':
DEF KEY 'I','GOSUB MATIDT':
DEF KEY 'O','GOSUB OPTRES':
DEF KEY 'P','GOSUB POUMAT':
DEF KEY 'S','GOSUB STOMAT':
DEF KEY 'D','GOSUB DETMAT':
DEF KEY 'K','GOSUB MULTCOL':
DEF KEY 'L','GOSUB MULTLGN':
DEF KEY '*','GOSUB MULTMAT':
DEF KEY 'C','GOSUB CLMAT':
DEF KEY 'V','GOSUB SMAT':
DEF KEY 'M','GOSUB MINMAX':
DEF KEY '#51','GOSUB RDN':
DEF KEY '+','GOSUB ADDMAT':
DEF KEY 'e','GOSUB EDITMAT':
DEF KEY 'r','GOSUB RESTKEYS':
DEF KEY 's','GOSUB RESSYS':
DEF KEY 'd','GOSUB DESTMAT':
DEF KEY 'k','GOSUB ADMLCMA':
DEF KEY 'l','GOSUB ADMLLMA':
DEF KEY 'm','GOSUB MULTMATK':
```

```
0650 READ #1;C(,) @ READ #2;B(,)
0660 FOR I=1 TO D @ FOR J=1 TO N @ S=0 @ FOR K=1 TO P @ S=S+B(I,K)*C(K,J) @ NEXT K @ A(I,J)=S
0670 NEXT J @ NEXT I
0680 ASSIGN #1 TO *
0690 RESTORE #2
0700 PRINT #2;D,N,A(,) @ ASSIGN #2 TO *
0710 M=0
0720 PURGE 'MAT'&STR$(P1)
0730 P1=P1-1
0740 DISP 'OK(';P1;')'
0750 END

================================================================
0760 'STOMAT': INPUT 'Sto n ';S$
0770 IF S$[LEN(S$)]='*' THEN 830
0780 IF P1=0 THEN DISP 'Pile vide !' @ BEEP @ END
0790 ON ERROR GOTO 860
0800 COPY 'MAT'&STR$(P1) TO 'SMAT'&S$
0810 DISP 'OK(';P1;')'
0820 OFF ERROR @ END
0830 PURGE 'SMAT'&S$[1,LEN(S$)-1]
0840 DISP 'OK(';P1;')'
0850 END
0860 PURGE 'SMAT'&S$ @ GOTO 800

================================================================
0870 'RAPMAT': INPUT 'Rcl n ';R$
0880 ON ERROR GOTO 980
0890 COPY 'SMAT'&R$ TO 'MAT'&STR$(P1+1)
0900 P1=P1+1
0910 ASSIGN #1 TO 'MAT'&STR$(P1)
0920 READ #1;M,N
0930 DIM A(M,N)
0940 READ #1;A(,)
0950 ASSIGN #1 TO *
0960 DISP 'OK(';P1;')'
0970 END
0980 OFF ERROR @ DISP 'Memoire vide !' @ BEEP @ END

================================================================
0990 'RDN': IF P1=1 THEN DISP 'Un seul element !' @ BEEP @ END
1000 IF P1=2 THEN GOTO 1060
1010 RENAME 'MAT'&STR$(P1) TO MATPASS
1020 RENAME 'MAT'&STR$(P1-1) TO 'MAT'&STR$(P1)
1030 RENAME 'MAT'&STR$(P1-2) TO 'MAT'&STR$(P1-1)
1040 RENAME MATPASS TO 'MAT'&STR$(P1-2)
1050 GOTO 1090
1060 RENAME 'MAT'&STR$(P1) TO MATPASS
1070 RENAME 'MAT'&STR$(P1-1) TO 'MAT'&STR$(P1)
1080 RENAME MATPASS TO 'MAT'&STR$(P1-1)
1090 ASSIGN #1 TO 'MAT'&STR$(P1)
1100 READ #1;M,N
1110 DIM A(M,N)
1120 READ #1;A(,)
1130 ASSIGN #1 TO *
1140 DISP 'OK(';P1;')'
1150 END

================================================================
1160 'MULTLGN': INPUT 'Ligne,Cte ? ';L,C1
1170 IF L>M THEN DISP 'Ligne inexistante !' @ BEEP @ END
```

```
============================================================
1180 'SUBMLGN': FOR I=1 TO N
1190 A(L,I)=A(L,I)*C1
1200 NEXT I
1210 GOTO 1390

============================================================
1220 'MULTCOL': INPUT 'Colonne,Cte ? ';K,C1
1230 IF K>N THEN DISP 'Colonne inexistante !' @ BEEP @ END

============================================================
1240 'SUBMCOL': FOR I=1 TO M
1250 A(I,K)=A(I,K)*C1
1260 NEXT I
1270 GOTO 1390

============================================================
1280 'ADMLCMA': INPUT 'Cln*,Cln+,Cte ? ';K1,K2,C1
1290 IF K1>N OR K2>N THEN DISP 'Colonne inexistante !' @ BEEP @ END

============================================================
1300 'SUBAMC': FOR I=1 TO M
1310 A(I,K2)=A(I,K2)+A(I,K1)*C1
1320 NEXT I
1330 GOTO 1390

============================================================
1340 'ADMLLMA': INPUT 'Lgn*,Lgn+,Cte ? ';L1,L2,C1
1350 IF L1>M OR L2>M THEN DISP 'Ligne inexistante !' @ BEEP @ END

============================================================
1360 'SUBAML': FOR I=1 TO N
1370 A(L2,I)=A(L2,I)+A(L1,I)*C1
1380 NEXT I
1390 IF FLAG(1) THEN RETURN
1400 ASSIGN #1 TO 'MAT'&STR$(P1)
1410 RESTORE #1
1420 PRINT #1;M,N,A(,)
1430 ASSIGN #1 TO *
1440 DISP 'OK(';P1;')'
1450 END

============================================================
1460 'AM': IF P1=1 THEN DISP 'Op. non valide !' @ BEEP @ END
1470 ASSIGN #1 TO 'MAT'&STR$(P1) @ ASSIGN #2 TO 'MAT'&STR$(P1-1)
1480 READ #1;M,N @ READ #2;O,P @ RETURN

============================================================
1490 'MATIDT': DESTROY A
1500 M=VAL(DISP$) @ N=M
1510 DIM A(M,N)
1520 FOR I=1 TO M
1530 A(I,I)=1
1540 NEXT I
1550 GOTO 100

============================================================
1560 'RESTKEYS': RENAME KEYS TO MATRKEYS @ RENAME MATRK TO KEYS @ DESTROY A,B,C @ END

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

```
1570 'EDITMAT': FOR I=1 TO M @ FOR J=1 TO N
1580 DISP USING "#,'(',K,')(',K,',',K,')= '";M#N,I,J
1590 LINPUT '',STR$(A(I,J));E$
1600 IF E$[1,1]#'#' THEN 1660
1610 IF E$[1,1]='#' AND NOT POS(E$,',') THEN 1400
1620 I=VAL(E$[2,POS(E$,',')-1]) @ J=VAL(E$[POS(E$,',')+1])
1630 IF I>M THEN I=M
1640 IF J>N THEN J=N
1650 GOTO 1580
1660 A(I,J)=VAL(E$)
1670 NEXT J @ NEXT I
1680 GOTO 1570

================================================================
1690 'TRIMAT': SFLAG 1
1700 FOR K=M TO 2 STEP -1
1710 FOR J=1 TO K-1
1720 L1=K @ L2=J
1730 IF A(K,K)=0 THEN 1760 ELSE C1=-A(J,K)/A(K,K)
1740 GOSUB 'SUBAML'
1750 NEXT J
1760 NEXT K @ CFLAG 1
1770 IF FLAG(2) THEN RETURN ELSE GOTO 1400

================================================================
1780 'DETMAT': IF M#N THEN DISP 'Lgn#Col !' @ BEEP @ END ELSE SFLAG 2
1790 GOSUB 'TRIMAT'
1800 CFLAG 2
1810 D=1 @ FOR I=1 TO M
1820 D=D*A(I,I)
1830 NEXT I
1840 DISP '[DET]=';D
1850 ASSIGN #1 TO 'MAT'&STR$(P1)
1860 READ #1;M,N,A(,)
1870 ASSIGN #1 TO *
1880 END

================================================================
1890 'RESSYS': DISP 'En cours de resolution' @ SFLAG 2
1900 GOSUB 'TRIMAT' @ CFLAG 2
1910 FOR K=1 TO M-1
1920 FOR J=M TO K+1 STEP -1
1930 L1=K @ L2=J
1940 IF A(K,K)=0 THEN 1980 ELSE C1=-A(J,K)/A(K,K)
1950 SFLAG 1
1960 GOSUB 'SUBAML'
1970 NEXT J
1980 NEXT K
1990 FOR J=1 TO M
2000 IF A(J,J)=0 THEN 2030
2010 L=J @ C1=1/A(J,J)
2020 GOSUB 'SUBMLGN'
2030 NEXT J @ CFLAG 1 @ GOTO 1400

================================================================
2040 'OPTRES': FOR I=1 TO M @ FOR J=1 TO N
2050 IF ABS(A(I,J))<.00001 THEN A(I,J)=0 @ BEEP INF,.1
2060 NEXT J @ NEXT I @ END

================================================================
2070 'MULTMATK': INPUT 'Constante ? ';C1
```

```
2080 FOR I=1 TO M @ FOR J=1 TO N
2090 A(I,J)=A(I,J)*C1
2100 NEXT J @ NEXT I @ END
```

****

Programme "DESSIL" (dessin sur HP 71B)

```
1000 DELAY 0
1010 DESTROY ALL @ RESET
1020 DIM A$[132],B$[132]
1030 INTEGER X,Y,C,I,J
1040 X=1 @ Y=7 @ A$=CHR$(128)
1050 GDISP A$
1060 SFLAG 6
   - Test clavier

================================================================
1070 'KEYTEST':
1080 IF KEYDOWN('#50') THEN 'UP'
1090 IF KEYDOWN('#51') THEN 'DWN'
1100 IF KEYDOWN('#47') THEN 'L'
1110 IF KEYDOWN('#48') THEN 'R'
1120 IF KEYDOWN('D') THEN 'KEYD'
1130 IF KEYDOWN('I') THEN 'INV'
1140 IF KEYDOWN('B') THEN 'MOVE'
1150 IF KEYDOWN('P') THEN 'POS'
1160 IF FLAG(4) THEN 'KEYTEST'
1170 IF KEYDOWN('T') THEN SFLAG 6 @ GOTO 'AFF'
1180 IF KEYDOWN('G') THEN CFLAG 6 @ GOTO 'AFF'
1190 IF KEYDOWN('C') THEN 'CHRDATA'
1200 GOTO 1080

================================================================
1210 'KEYD': ! E/S mode deplacement
1220 I=FLAG(4,NOT FLAG(4))
1230 IF FLAG(4) THEN B$=A$ @ RADIANS ELSE DEGREES
1240 GOTO 'AFF'
   - Déplace le curseur

================================================================
1250 'UP': IF Y>0 THEN Y=Y-1
1260 GOTO 'AFF'

================================================================
1270 'DWN': IF Y<7 THEN Y=Y+1
1280 GOTO 'AFF'

================================================================
1290 'L': IF X>1 THEN X=X-1
1300 GOTO 'AFF'

================================================================
1310 'R': IF X<132 THEN X=X+1
   - Affichage

================================================================
1320 'AFF': IF FLAG(4) THEN A$=B$
```

```
1330 C=NUM(A$[X])
1340 IF FLAG(4) THEN 'DEP'

================================================================
1350 'T': IF FLAG(6) AND NOT BIT(C,Y) THEN C=C+2^Y

================================================================
1360 'G': IF NOT FLAG(6) AND BIT(C,Y) THEN C=C-2^Y
1370 GOTO 'FIN'

================================================================
1380 'DEP':
1390 IF BIT(C,Y) THEN C=C-2^Y ELSE C=C+2^Y

================================================================
1400 'FIN': BEEP 900,.005
1410 A$[X,X]=CHR$(C)
1420 GDISP A$ @ A$=GDISP$
1430 GOTO 'KEYTEST'

================================================================
1440 'POS': A$=GDISP$
1450 DISP @ CFLAG 4
1460 INPUT 'De 0 a 131: X=';X
1470 INPUT 'De 0 a 7: Y=';Y
1480 X=MOD(X+1,133)
1490 Y=7-MOD(Y,8)
1500 GOTO 'KEYD'

================================================================
1510 'CHRDATA': A$=GDISP$ @ B$=A$
1520 DISP 'Codes:'
1530 IF NUM(A$)=0 THEN A$=A$[2] @ GOTO 1530
1540 DISP "(V)oir   (S)auver"
1550 IF KEYDOWN('V') THEN DELAY INF @ GOTO 1570
1560 IF KEYDOWN('S') THEN GOSUB 1990 ELSE 1550
1570 FOR I=132 TO 1 STEP -1
1580 IF NUM(A$[I])<>0 THEN 1600
1590 NEXT I
1600 I=6*CEIL(I/6)
1610 FOR J=1 TO I STEP 6
1620 IF NOT FLAG(5) THEN DISP USING "#,'CHR',2D,':'";CEIL(J/6)
1630 FOR C=J TO J+5
1640 IF NOT FLAG(5) THEN DISP USING "#,K,',',";NUM(A$[C])
1650 IF FLAG(5) THEN PRINT #1;NUM(A$[C])
1660 NEXT C
1670 IF NOT FLAG(5) THEN DISP
1680 NEXT J
1690 DELAY 0,0 @ GDISP B$
1700 ASSIGN #1 TO *
1710 CFLAG 5
1720 GOTO 'KEYTEST'

================================================================
1730 'INV': A$=GDISP$
1740 FOR J=1 TO 132
1750 C=NUM(A$[J])
1760 C=255-C
1770 A$[J,J]=CHR$(C)
1780 GDISP A$
1790 NEXT J
```

```
1800 B$=A$
1810 GOTO 'KEYTEST'

============================================================
1820 'MOVE':
1830 BEEP 1500,.05 @ SFLAG 0 @ GOTO 1850
1840 IF KEYDOWN('B') THEN BEEP 1500,.05 @ CFLAG 0 @ A$=GDISP$ @ GOTO 1800
1850 IF KEYDOWN('#48') THEN GDISP CHR$(0)&GDISP$
1860 IF KEYDOWN('#47') THEN GDISP GDISP$[2]
1870 IF KEYDOWN('#50') THEN J=0 @ GOSUB 1900
1880 IF KEYDOWN('#51') THEN J=1 @ GOSUB 1900
1890 GOTO 1840
1900 A$=GDISP$ @ USER ON
1910 FOR I=1 TO 132
1920 C=NUM(A$[I])
1930 IF C=0 THEN 1970
1940 IF J THEN C=2*C ELSE C=C DIV 2
1950 A$[I,I]=CHR$(C)
1960 GDISP A$
1970 NEXT I @ USER OFF
1980 RETURN
1990 SFLAG 5 @ DISP
2000 INPUT "Nom du fichier: ";N$
2010 CREATE SDATA N$
2020 ASSIGN #1 TO N$
2030 RETURN
```

****

Programme "K71" (éditeur de cassettes)

```
1000 DIM E$[256],C$[4]
```
- E$: variable alphanumérique contenant l'enregistrement spécifié.
  C$: variable alphanumérique contenant la valeur des touches pressées dans la boucle d'attente.

```
1010 INTEGER C,D,E,I,O
```
- C: valeur du code ASCII où se trouve le pointeur (variable O).
  D: adresse du lecteur dans la boucle.
  E: valeur locale du pointeur d'enregistrement.
  I: comme d'habitude.
  O: valeur locale du pointeur d'octet

```
1020 RESET HPIL
1030 RESTORE IO
1040 CFLAG 0
```
- L'indicateur binaire 0 sert à valider la modification de l'enregistrement courant.

```
1050 DELAY 0,0
1060 D=DEVADDR('%16')
```
- Si plusieurs lecteurs de cassette se trouvent dans la boucle, le programme utilisera le premier qu'il rencontrera.

```
1070 IF D>0 THEN 'ENTREE'
1080 BEEP
1090 DISP 'Pas de lecteur !!?'
1100 END

============================================================
1110 'ENTREE':
1120 CLEAR :D @ I=SPOLL(D)
```
- Teste si le lecteur n'est pas en condition d'erreur.

```
1130 IF I>16 AND I<29 THEN BEEP @ DISP 'Erreur lecteur !!?' @ END
1140 SFLAG -23
   - Merci Pierre DAVID. Lorsque le drapeau -23 est à 1, un message EOT (End Of Transmission)
     suit l'envoi d'une variable dans la fonction 'ENTER'.
1150 GOSUB 'LECTURE'

================================================================
1160 'RETOUR':
1170 GOSUB 'AFFICHE'

================================================================
1500 'BOUCLE':
1510 C$=KEY$
1520 IF C$='' THEN 'BOUCLE'
1530 IF C$='#51' THEN GOSUB 'IENR'
1540 IF C$='#50' THEN GOSUB 'DENR'
1550 IF C$='#48' THEN GOSUB 'IOCT'
1560 IF C$='#47' THEN GOSUB 'DOCT'
1570 IF C$='E' THEN GOSUB 'XENR'
1580 IF C$='O' THEN GOSUB 'XOCT'
1590 IF C$='C' THEN GOSUB 'CARACT'
1600 IF C$='R' THEN GOSUB 'EFFACE'
1610 IF C$<>'F' THEN 'RETOUR'
   - [ATTN/ON], arrête le programme sans écrire l'enregistrement courant.

================================================================
2000 'FIN':
2010 IF FLAG(0,0) THEN GOSUB 'ECRITURE'
   - Si l'indicateur binaire 0 égale 1: des modifications ont été faites sur l'enregistrement
     courant, il faut réécrire ces modifications.
2020 SEND LISTEN D
   - Déclare le lecteur comme appareil récepteur.
2030 SEND DDL 7
   - Ordre récepteur dépendant 7 (DDL7): rembobinage. Rembobine la bande complètement. La
     bande ne peut alors être utilisée que si la porte est ouverte et refermée, un ordre HPIL
     Appareil libre est reçu ou un ordre Recherche (DDL4) repositionne la bande.
2040 SEND IFC
   - Interface libre (InterFace Clear): efface l'état émetteur, l'état récepteur ou les
     ordres en attente de tous les périphériques de la boucle.
2050 CFLAG -23
   - Voir 'ENTREE'
2060 END

================================================================
3000 'IENR':
3010 IF FLAG(0,0) THEN GOSUB 'ECRITURE'
   - Si l'indicateur 0 est disposé, des modifications ont été faites sur E$.....
3030 E=MOD(E+1,512)
3040 GOSUB 'LECTURE'
3050 RETURN

================================================================
3100 'DENR':
3110 IF FLAG(0,0) THEN GOSUB 'ECRITURE'
3130 E=MOD(E-1,512)
3140 GOSUB 'LECTURE'
3150 RETURN

================================================================
3200 'IOCT':
3210 O=MOD(O+1,256)
```

```
3220 RETURN

================================================================
3300 'DOCT':
3310 O=MOD(O-1,256)
3320 RETURN

================================================================
3400 'XENR':
3410 IF FLAG(0,0) THEN GOSUB 'ECRITURE'
3430 INPUT 'E-',STR$(E);E
3440 GOSUB 'LECTURE'
3450 RETURN

================================================================
3500 'XOCT':
3510 DISP TAB(7);'O-';
3520 INPUT '',STR$(O);O
3530 IF O<0 OR O>255 THEN 'XOCT'
3540 RETURN

================================================================
3600 'CARACT':
3610 DISP TAB(13);'C-';
3620 INPUT '',STR$(C);C
3630 E$[O+1,O+1]=CHR$(C)
3640 SFLAG 0
3650 RETURN

================================================================
3700 'EFFACE':
3710 FOR I=O+1 TO 256
3720 E$[I,I]=CHR$(255)
3730 NEXT I
3740 SFLAG 0
3750 RETURN

================================================================
5000 'ECRITURE':
5010 GOSUB 'POINTEUR'
   - Place la bande à la bonne adresse.
5020 SEND DDL 6
   - Ecriture partielle. L'enregistrement courant est copié dans la mémoire tampon 0 et la
     bande revient au début du même enregistrement. Les octets de données reçus remplacent le
     contenu de la mémoire 0 en commençant à l'emplacement du pointeur (le pointeur est à 0
     pour le programme). Lorsque la mémoire est pleine, son contenu est transféré dans l'en-
     registrement qui a été lu.
5030 SEND DATA E$
   - La longueur de E$ égale 256 octets, 'Lorsque la mémoire est pleine.....': voir DDL6.
5040 RETURN

================================================================
6000 'LECTURE':
6010 GOSUB 'POINTEUR'
6020 SEND DDL 6
6030 SEND MLA
   - Définit le HP71 comme récepteur (My Listen Adress).
6040 SEND TALK D
   - Définit le lecteur comme émetteur (il ne peut y avoir qu'un seul émetteur à la fois sur
     la boucle), et annule l'état récepteur du lecteur.
6050 SEND DDT 0
```

- Envoi mémoire 0. Un message Envoi données suivant (inclus dans ENTER) transfère le contenu de la mémoire 0 sur la boucle à partir de la position du pointeur.

6060 ENTER :D ;E$

- E$=nouvel enregistrement.

6070 RETURN

==

7000 'POINTEUR':

7010 SEND UNL

- Efface l'état récepteur de tous les périphériques. Vidéo par exemple qui afficherait les octets passant devant elle.

7020 SEND MTA

- Le HP71 devient l'émetteur de la boucle.

7030 SEND LISTEN D

- Le lecteur est le seul récepteur.

7040 SEND DDL 4

- Recherche. Les deux octets de données suivants définissent les numéros de piste (0 ou 1) et d'enregistrement (0 à 255) et y positionnent la bande. Cet ordre annule le mode d'écriture partielle (DDL6).

7050 SEND DATA CHR$(MOD(E DIV 256,2))&CHR$(MOD(E,256))

- Envoi des deux octets.

7060 SEND DDL 3

- position du pointeur. Les octets suivants définissent la position du pointeur de 0 à 255.

7070 SEND DATA CHR$(0)

- Envoi de l'octet 0.

7080 RETURN

==

8000 'AFFICHE':

8010 IMAGE 'E-',3Z,' O-',3Z,' C-',3Z,X,A,X,2A

- Affichage du type 'E-052 O-012 C-065 A 41': enregistrement 52, octet 12, caractère 65, A, code hexa 41.

8020 C=NUM(E$[O+1])

8030 IF C>31 AND C<127 THEN C$=CHR$(C) ELSE C$=' '

8040 DISP USING 8010;E,O,C,C$,DTH$(C)[4,5]

8050 C$=''

8060 RETURN

!!!

Programme "GESTHPIL" (gestion de la boucle HPIL)

1000 RESET HPIL @ RESTORE IO

- Dispose et assigne la boucle.

1010 OPTION BASE 1

1020 DIM A$[60]

- A$: Variable brouillon.

1030 INTEGER A(1),D(30),I,J,K,N,P @ P=0

- A: adresse des HP 71.
  D: Numéro des HP 71 demandeurs.
  N: nombre de HP 71 dans la boucle.
  P: pointeur dans la pile des demandeurs (pile FIFO).
  I,J,K: variables brouillon.

==

1040 DEF FNH$(X)='HP71('&STR$(X)&')'

- En clair: HP 71(X)

```
================================================================
1050 DEF FNT(X)
   - Teste si le HP 71 numéro X est déjà demandeur.
     1 s'il ne l'est pas, 0 s'il l'est.
1060 FNT=1
1070 FOR K=1 TO P
1080 IF D(K)=X THEN FNT=0 @ GOTO 1100
1090 NEXT K
1100 END DEF

================================================================
1110 'INIT':
1120 FOR I=1 TO 30
1130 J=DEVADDR(FNH$(I))
   - J: adresse du i-ème HP 71.
1140 IF J<0 THEN 1190
   - Si elle est négative , alors le i-ème 71 n'existe pas.
1150 INTEGER A(I)
   - Le tableau A est dimensionné exactement au nombre de HP 71 dans la boucle.
1160 A(I)=J
1170 GOSUB 'RESETHP'
   - On prépare le 71 à l'adresse J.
1180 NEXT I
1190 N=I-1
   - N: Nombre de HP 71.
1200 DISP 'Il y a';N;'HP71.'
1210 BEEP
1220 IF N=0 THEN STOP
   - On arrête si il n'y a pas d'autre HP 71 dans la boucle...
1230 J=DEVADDR('HP71')
   - On initialisera (ligne 1260) un HP 71. Alors, autant que cette adresse existe !

================================================================
1240 'BOUCLE':
1250 CONTROL ON
   - On reprend le contrôle (le HP 71 demandeur vient de rendre la main).
1260 GOSUB 'RESETHP'
   - On redispose ce HP 71.
1270 DISP P; @ FOR I=1 TO P @ DISP D(I); @ NEXT I
   - On affiche le nombre de demandeurs, et leurs numéros.
1280 FOR I=1 TO N
1290 IF SPOLL(A(I))=160 AND FNT(I) THEN GOSUB 'EMPILE'
   - Si le ième HP 71 demande le contrôle, et qu'il n'est pas dans la pile D, alors on
     empile son adresse.
1300 IF KEY$='#155' THEN 'FIN'
   - Arrêt si [g] [ON].
1310 NEXT I @ DISP
1320 IF P>0 THEN 'PASSE' ELSE 1270
   - Si le nombre de demandeurs (P) est non nul, on passe le contrôle, sinon on boucle.

================================================================
1330 'EMPILE':
1340 IF P=30 THEN RETURN
   - Pile pleine.
1350 P=P+1
1360 D(P)=A(I)
   - L'adresse A(I) est mise au sommet de la pile D.
1370 RETURN

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

```
1380 'RESETHP':
1390 REMOTE :J
   - Place le HP 71 d'adresse J en mode télécommande (les données envoyées sont interprétées
     comme des commandes Basic).
1400 OUTPUT :J ;'USERON@REQUEST0'
   - Initialisation:  REQUEST 0 veut dire "je ne demande pas le controle".
1410 OUTPUT :J ;'KEY"#155","REQUEST160@BEEP999,.1":'
   - Assignation de la touche [g] [ON].
1420 LOCAL :J
   - Remise en mode LOCAL (mode normal).
1430 RETURN

================================================================
1440 'PASSE':
1450 J=D(1)
1460 FOR I=1 TO P-1
   - Si P<=1, la boucle n'est pas éxécutée.
1470 D(I)=D(I+1)
1480 NEXT I
   - Dépile le tableau P.
1490 P=P-1
1500 FOR I=1 TO N
1510 IF A(I)=J THEN 1530
   - On recherche le numéro connaissant l'adresse.
1520 NEXT I
1530 A$="BEEPINF,.1@REMOTE"&FNH$(N-I+1)&"@OUTPUT"&FNH$(N-I+1)&";'RUN,BOUCLE'"
1540 OUTPUT :J ;'REQUEST0@KEY"#155","'&A$&'":@RESTOREIO@BEEP'
   - Assignation de la touche [g] [ON].
1550 LOCAL :J
1560 A$='eme' @ IF I=1 THEN A$='er'
   - Plus agréable...
1570 DISP 'Controle: ';STR$(I);A$;' HP71.'
1580 PASS CONTROL :J
   - Le passage de contrôle proprement dit.
1590 STOP

================================================================
1600 'FIN':
1610 FOR I=1 TO N
1620 REMOTE :A(I)
1630 OUTPUT :A(I) ;'USEROFF@KEY"#155"'
   - "Désassignation" de la touche [g] [ON].
1640 LOCAL :A(I)
1650 NEXT I
1660 END
```

PPC PARIS CHAPTER

ASSOCIATION REGIE PAR LA LOI DE 1901, ENREGISTREE A PARIS LE 2 DECEMBRE 1982 SOUS LE NUMERO 82/3240

BULLETIN D'ADHESION

NOM ¦_¦
PRENOM ¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦_¦ DATE DE NAISSANCE ¦_¦_¦/¦_¦_¦/¦_¦_¦
ADRESSE ¦_¦
¦_¦
¦_¦
COMMUNE ¦_¦
CODE POSTAL ¦_¦_¦_¦_¦_¦ PAYS ¦_¦
TELEPHONE DOMICILE ¦_¦_¦/¦_¦_¦_¦_¦_¦_¦_¦_¦ BUREAU ¦_¦_¦/¦_¦_¦_¦_¦_¦_¦_¦_¦

PROFESSION ____
INTERETS ____
____
MATERIEL HP EN VOTRE POSSESSION ______________________________
____
____
AUTRE MATERIEL MICRO-INFORMATIQUE ______________________________
____
____

COMMENT AVEZ-VOUS CONNU PPC PARIS CHAPTER ?
PUBLICITE ______________________ MAGAZINE ______________________
AUTRE CLUB ______________________ HP ______________________
RELATIONS, MEMBRES DU CLUB, AUTRES ______________________________

QUE RECHERCHEZ-VOUS AU SEIN DU PPC PARIS CHAPTER ? ______________________
____
____
____

Je souhaite adhérer au club PPC PARIS CHAPTER 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 du PPC PARIS CHAPTER.

DATE ¦_¦_¦/¦_¦_¦/19¦_¦_¦
SIGNATURE, PRECEDEE DE LA MENTION "LU ET APPROUVE" ____________________

LE MONTANT DE LA COTISATION AU PPC PARIS CHAPTER S'ELEVE A 30,00 FF.
L'ABONNEMENT POUR UN AN AU JOURNAL J.P.C. S'ELEVE A 170,00 FF.
PAIEMENT EFFECTUE LE ¦_¦_¦/¦_¦_¦/19¦_¦_¦ A L'ORDRE DE PPC PARIS CHAPTER.
PAR [ ] CHEQUE BANCAIRE N° ____________ BANQUE ______________
[ ] CHEQUE POSTAL 3 VOLETS N° ______________________
[ ] MANDAT LETTRE
EVENTUELLEMENT: JE M'ABONNE A COMPTER DU ¦_¦_¦/¦_¦_¦/19¦_¦_¦
JOINDRE A VOTRE INSCRIPTION UNE ENVELOPPE TIMBREE A VOTRE ADRESSE.

VEUILLEZ ENVOYER TOUTE CORRESPONDANCE A:
MR PHILIPPE GUEZ, 56 RUE J.J. ROUSSEAU, 75001 PARIS (FRANCE)