IREM de STRASBOURG

Groupe Informatique

PASCAL

MANIPULATION ET

TECHNIQUES DE PROGRAMMATION

oct 1981.

PASCAL

MISE EN OEUVRE SUR APPLE II

Plan

IREM DE STRASBOURG

Février 1981

P A S C A L

MISE EN OEUVRE SUR APPLE II

I Qu'est-ce que PASCAL ?

A/ Avantages

- PASCAL est un langage puissant permettant une programmation méthodique "structurée".
  Contrairement à BASIC qui ne possède que des formes très réduites de "procédures", PASCAL permet la programmation en petits modules séparés autonomes.
- PASCAL autorise dans l'écriture un degré élevé de symbolisme facilitant la lisibilité, la mise au point (on dira par exemple TOURNER A DROITE au lieu de GOSUB 1000)
- PASCAL exigeant de déclarer tous les objets utilisés par le programme est sur ce plan très lisible, les déclarations implicites étant minimales.
- PASCAL est performant à l'exécution (2 à 5 fois plus rapide) car il est compilé et non pas interprété . (En fait, la version UCSD est compilée vers un langage intermédiaire appelé P-code, ce dernier est ensuite interprété).
- PASCAL est "portable" parce que performant. Pascal permet à l'utilisateur de définir ses propres objets ce qui a évité les prolifération comme celles des dialectes BASIC incompatibles. Les incompatibilités des différentes implémentations sont beaucoup plus limitées en PASCAL qu'en BASIC.

B/ Inconvénients

Une certaine lourdeur de mise au point :

L'utilisateur doit passer obligatoirement par les étapes suivantes :

1° Saisie-correction du fichier programme en tant que chaîne de caractères (fichier-texte) avec le programme EDITOR (éditeur fourni avec PASCAL)

2° Sauvegarde du fichier-texte sur la disquette souple avec FILER (logiciel de gestion de fichier fourni avec PASCAL)

3° Compilation du fichier-texte et détection des erreurs de syntaxe avec le programme COMPILER. (Plus de 130 erreurs détaillées, beaucoup plus explicites qu'en BASIC). COMPILER crée un "fichier-code".
S'il y a des erreurs il faut revenir en 1. pour les corriger.

4° Editions de liens avec le programme LINKER. Le système place dans le fichier-code les procédures nécessaires (par exemple les procédures graphiques). En cas d'erreur retour en 1.

5° Exécution, enfin, du fichier-code.
En cas d'erreur de logique, retour en 1.

REM 1 : Les étapes 3. 4. et 5. sont chaînées automatiquement avec l'option R(UN.

REM 2 : Sur APPLE II les étapes 3. et 4. pouvaient être un goulot d'étranglement mais se révèlent très rapides (plus rapides que sur PdP11/34 en temps partagé...)

Un temps d'apprentissage plus long des fonctions du système d'exploitation, plus puissant mais aussi plus complexe qu'en BASIC.

II Une première approche de PASCAL APPLE II

(les caractères tapés par l'utilisateur sont soulignés)

A/ Chargement du système

1. Couper l'ordinateur
2. Insérer la disquette système APPLE 1 : dans l'unité un (unité d'initialisation automatique)
3. Mettre l'ordinateur sous tension, l'unité 1 s'allume, le système se charge automatiquement.
4. Le système édite la ligne de sélection globale des "commandes" disponibles
   E(DIT, R(UN, F(ILE, C(OMP, L(IN
   Tapez sur CTRL-A pour avoir la suite :
   K, X(ECUTE, A(SSEM, D(EBUG
   Retour à la gauche de l'écran avec CTRL-A
   Nettoyez l'écran avec 2 Retour-Chariot (RC).

B/ Utilisation d'un programme existant

1. Placez la disquette APPLE 3 : dans l'unité droite
2. Pour éditer le catalogue de cette disquette sélectionnez l'option FILER par F (la lecture du clavier "à la volée" évite de taper RC)
3. La ligne de sélection du système d'exploitation des fichiers apparaît :
   FILER : G, S, N, L, R, C, T, D, Q
   Sélectionnez L pour avoir le catalogue, il apparaît à l'écran
   DIR LISTING OF ? (catalogue de quelle disquette ?)
   Tapez APPLE 3 : RC
   Le programme de démonstration GRAFDEMO est dans cette liste.
4. Retour au niveau global avec Q (quitter)
5. Tapez X pour charger et exécuter le module choisi.
6. Il apparaît EXECUTE WHAT FILE ? (exécuter quel fichier ?)
   Tapez APPLE 3 : GRAFDEMO RC
7. Le programme de démonstration graphique s'exécute.

C/ Saisie, correction, exécution d'un programme PASCAL

1. Choisir E(DIT en tapant E au niveau global de sélection, le système édite :
   > EDIT :
   NOWORFILE IS PRESENT. FILE ? (RET FO
   (pas de fichier dans la zone de travail, nom du fichier ?)
   Tapez ESC RC pour ressortir, RC pour créer un fichier.
   Il apparaît :
   EDIT : A(DJUST, C(PY, D(LETE, F(IND, I(NSERT, J
   et après CTRL-A
   (UMP, R(PLACE, Q(UIT, X(CHNG, Z(AP
   qui sont les principales commandes disponibles.
   Tapez I pour Insertion
   Il apparaît :
   > Insert : TEXT [BS ACHAR DEL A LINE]

2. Les touches de fonctions :

(Vous obtiendrez [par CTRL - K et] par shift M

(Vous obtiendrez les minuscules avec CTRL-E, annulation avec un CTRL - W)

En fait les lettres apparaissent à l'écran en vidéo normale et inverse. Ce n'est qu'après la sortie de l'éditeur que vous les aurez en minuscules.

Avec CTRL-A éditez la deuxième moitié de l'écran où se trouve la suite de la ligne de commandes de l'éditeur :

ETX accepts, ESC ES capes

ESC permet d'échapper au travail en cours

RESET réinitialise le système et perd le contenu de la mémoire sauf si sauvé sur disque !!

CTRL-X efface la ligne tapée en dernier

CTRL-C valide l'opération d'édition qui vient d'être effectuée

CTRL-O déplace le curseur vers le haut

CTRL-L " " " le bas

← déplace le curseur vers la gauche

→ " " " la droite

3. Ecriture d'un programme

I place l'éditeur en mode Insertion

tapez :

```
PROGRAM SALUT ; RC
BEGIN RC
WRITELN ('SALUT LES COPAINS') ; RC
END. RC
```

Concluez par CTRL-C

4. Rajouter une ligne :

tapez F pour Find (trouver)

déplacez le curseur jusqu'à l'emplacement souhaité (après BEGIN par exemple) en tapant

/BEGIN/RC

aussitôt le curseur se place après le BEGIN.

tapez I puis

RC

Une nouvelle ligne : WRITELN ; WRITELN ; RC

Concluez par CTRL-C

5. Sauvegarde du travail sur disquette

tapez Q par quitter l'Editeur

Il apparaît

QUIT :

U(P date the workfile and leave

E(XIT without updating

R(ETURN to the Editor without updat

W(RITE to a file name and RETURN

t apez U

pour mettre à jour la zone de travail

Une fois dans la ligne de commande globale tapez F puis L

pour avoir le catalogue de l'unité 1.

Il apparaît que le fichier

SYSTEM.WRK. TEXT a été crée par le système. C'est dans ce fichier que se trouve votre texte.

6. Compilation, édition de liens, exécution

- placez la disquette APPLE 2 : dans l'unité 2 (elle contient le compilateur et l'éditeur de liens)
- tapez R pour RUN commande qui enchaîne
  a) compilation C(OMPILE
  b) édition de liens L(INK
  c) exécution X(ecute
- il apparaît une suite de messages à l'écran, s'il n'y a pas d'erreur le programme affiche deux sauts de ligne et SALUT LES COPAINS saut de ligne
- puis le système retourne à la ligne globale de commande.

Le FILER montre que le système a créé un fichier SYSTEM. WRK.CODE qui peut être sauvé sous un autre nom sur une autre disquette PASCAL avec l'option T (transfer)

LIMITES D'IMPLEMENTATION

NOMBRE DE VOLUMES	12
ECRAN	texte : 24 lignes de 80 caractères (par CTRL A) graphique : 279 de large par 191 de haut
DISQUETTE	280 blocs de 512 octets. blocs 0 et 1 : bootstrap bloc 2 à 5 : table des matières x : disquette d'initialisation : disquette par défaut
INTEGER	limité à 32767, ou utiliser INTEGER N pour N 36
IDENTIFICATEUR	disquette : moins de 8 caractères fichier : moins de 16 caractères
STRING	par défaut moins de 80 caractères. Ou préciser STRING N pour N =255
SET	moins de 512 éléments
PROCEDURE FUNCTION	moins de 1200 octets de code objet variables locales utilisant moins de 16 383 octets
SEGMENT	0 à 9 utilisés par PASCAL. 10 à 15 pour l'utilisateur au plus 127 procédures ou fonctions dans un segment.

ANNEXES

NOMS DE VOLUME ET DE FICHIERS

Nom de "VOLUME"

Le volume est désigné par son nom suivi de deux points :

PRINTER : (imprimante) # 6
CONSOLE : (écran) # 1
+, ou , : désignent l'unité d'initialisation

nom de disquette (≤ 7 caractère sans = $ ou ?, est suivi également de :

n° de volume (sans les :) #4 = unité 1
#5 = unité 2

Nom de fichier

Longueur ≤ 15 caractères, les 5 derniers caractères sont forcément .TEXT ou .CODE.
Tous les caractères alphanumériques sont légaux ainsi que + - / \ et le point.

Sont illégaux $: = et ?

Nom de fichier généralisé

Les caractères = et ? servent à désigner un ensemble de fichiers indiquant quelle partie du nom des fichiers doit être ignorée.
Le nom ?.TEXT indique qu'il faut rechercher tous les fichiers terminés par .TEXT.
Il faut préférer le signe ? au signe =, car dans le premier cas, le système demande à l'écran confirmation des opérations effectuées ce qui est toujours utile.
Le ? seul indiquera tous les fichiers de la disquette.

L'EDITEUR DE TEXTE

Adjust	décale, à l'aide de - ou - la ligne où se trouve le curseur. Déplace les lignes du dessus par (CTRL O) ou du dessous par (CTRL L) du même montant. Conclure par (CTRL C)
Copy	copie à l'endroit où se trouve le curseur - une partie d'un texte comprise entre deux étiquettes - le texte que l'on vient d'effacer (dans BUFFER)
Delete	efface le texte à l'endroit où l'on déplace le curseur. Le retour en arrière fait reapparaître le texte.
Find	recherche la n ième occurence d'une chaine de caractères, comprise entre / et /. S répète cette recherche
Insert	insère un texte. permet d'effacer le texte écrit.
Jump	saute au début (B) à la fin (E) où à l'endroit où est placée une étiquette (M)
Margin	ajuste le paragraphe où se trouve le curseur entre les marges définies dans l'Environement.
Page	déplace le curseur d'une page.
Quit	quitte l'Editeur, soit en mettant à jour le fichier de travail sur la disquette (U) soit en quittant sans mise à jour (E) soit sauvegarde le fichier de travail sur la disquette (W) ou retourne à l'Editeur (R).
Replace	remplace une chaine de caractères par une autre, un nombre de fois défini par le facteur de répétition, dans la direction définie, à partir du curseur. Les chaines sont placées entre / et / .
Set	Marker : place une étiquette à l'endroit du curseur. Environment : permet de définir les options de décalage automatique (A) de marges.
eXchange	remplace le caractère sous le cursuer par le caractère frappé.

FILER (DOS)

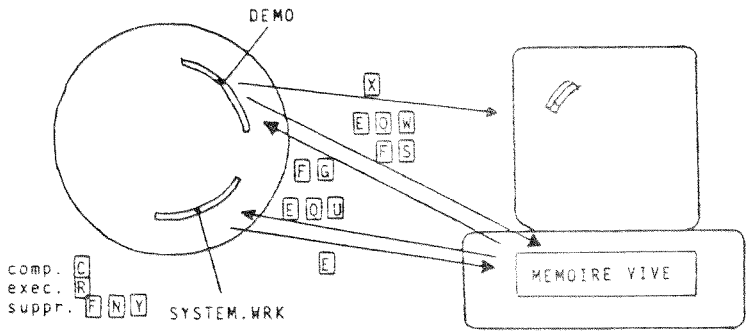
Bad	vérifie si l'un des 280 blocs est physiquement endommagé.
Change	change le nom d'une disquette ou d'un fichier.
Date	met à jour la date du système.
Extended list	table des matières détaillée d'une disquette.
Get	désigne un fichier sur une disquette comme fichier de travail, qui peut être Edité, Compilé, ou éxécuté par Run. Ce fichier ne remplace SYSTEM.WRK qu'après Update.
Krunch	rassemble les espaces inutilisés sur une disquette (UNUSED) à la fin de la disquette.
Make N	réserve N blocs pour un fichier, et le place dans la table des matières.
New	efface SYSTEM.WRK de la disquette.
Prefix	permet d'attribuer un nom (autre que celui de la disquette d'initialisation) à la disquette " préfixe " qui pourra être désignée par :
Quit	quitte Filer pour retourner au niveau de sélection global.
Remove	retire le fichier désigné de la table des matières d'une disquette.
Save	sauvegarde SYSTEM.WRK sous un nom propre.
Transfer	transfert un fichier vers un autre volume. Est utilisé pour transférer un fichier, copier une disquette, ou imprimer un fichier.
Volume	précise quels volumes sont actuellement branchés.
What	précise quel est le fichier actuellement considéré comme WORKFILE.
eXamine	essaye de remédier aux défauts détectés par Bad.

L'ARBRE DES COMMANDES APPLE PASCAL.

- EDIT
  - Adjust
  - Copy
    - File
    - Buffer
  - Delete
  - Find
  - Insert
  - Jump
    - Beginning
    - End
  - Margin
  - Page
  - Quit
    - Update
    - Exit
    - Write
    - Return
  - Replace
  - Set
    - Marker
    - Environment
      - Left margin
      - Right marg.
      - Paragr. Marg.
      - Filling
  - Verify
  - eXchange
- FILER
  - Bad
  - Change
  - Date
  - Extended list
  - Get
  - Krunch
  - Make
  - New
  - Prefix
  - Quit
  - Remove
  - Save
  - Transfer
  - Volume
  - What
  - eXamine
- COMPILE
- ASSEMBLE
- LINK
- eXECUTE
- RUN

LES FICHIERS SUR DISQUETTE

depuis le niveau de sélection global (COMMAND) :



			BLOCS	DISQUETTE	
UTILITAIRES	SYSTEM.APPLE		32	1	3
	SYSTEM.PASCAL		36	0 1	
	SYSTEM.MISCINFO		1	0 1	
	SYSTEM.EDITOR		45	0 1	
	SYSTEM.FILER		28	0 1	
	SYSTEM.LIBRARY		36	0 1	
	SYSTEM.CHARSET		2	0 1	
	SYSTEM.SYNTAX		14	0 1	
	SYSTEM.COMPILER		71	0	2
	SYSTEM.LINKER		24	0	2
	FORMATTER.CODE		4		3
	FORMATTER.DATA		6		3
	LIBRARY.CODE		9		3
	LIBMAP.CODE		9		3
DEMONSTRATION	CALC	calculatrice de poche			3
	CROSSREFF	tableau de références			3
	SPIRODEMO	dessin			3
	HILBERT	récursion graphique			3
	GRAFDEMO	dessin			3
	GRAFCHARS	dessin			3
	TREE	pointeur			3
	BALANCED TREE	pointeur			3
	DISKIO	fichier			3
SYSTEM.LIBRARY	LONGINTI	entiers supérieurs à 32000			
	PASCALIO				
	TRANSCEN	pour SIN COS ATAN LN LOG EXP			
	APPLESTUFF	pour RANDOM PADDLE BUTTON NOTE			
	TURTLEGRPHICS	pour le dessin graphique			

1 erreur dans un type simple
2 identificateur attendu
3 'PROGRAM' attendu
4 '(' attendu
5 ')' attendu

6 symbole illégal
7 erreur dans la liste de paramètres
8 'OF' attendu
9 '(' attendu
10 erreur de type

11 ')' attendu
12 ':' attendu
13 'END' attendu
14 ';' attendu
15 entier attendu

16 '=' attendu
17 'BEGIN' attendu
18 erreur dans la déclaration
19 erreur dans la liste
20 '.' attendu

21 'x' attendu
22 'interface' attendu
23 'implémentation' attendu
24 'Unit' attendu

50 erreur dans la constante
51 ':=' attendu
52 'THEN' attendu
53 'UNTIL' attendu
54 'DO' attendu
55 'TO' ou 'DOWNTO' de l'instruction FOR attendu

56 'OF' attendu
57 'FILE' attendu
58 erreur dans le facteur (expression incorrecte)
59 erreur dans la variable

101 identificateur déclaré deux fois
102 limite inférieure plus grande que limite supérieure
103 identificateur dans une classe iappropriée
104 identificateur non déclaré
105 signe non autorisé

106 nombre attendu
107 types sous ensembles incompatibles
108 fichiers non permis ici
109 le type ne peut pas être réel
110 étiquette doit être un scalaire ou un sous-ensemble

111 incompatibilité avec l'étiquette
112 l'indexe ne doit pas être un réel
113 le type de l'indexe doit être scalaire ou sous-ensemble
114 le type de base ne peut pas être réel
115 le type de base doit être scalaire ou sous-ensemble

116 erreur dans le type des paramètres de la procédure standard
117 référence anticipée non satisfaite
118 identificateur de type anticipé dans une déclaration de variable
119 ne pas redéclarer les paramètres d'une procédure FORWARD
120 le résultat de la fonction doit être scalaire, sous ens ou point.

121 les fichiers ne peuvent pas être paramètres valeur
122 le type du résult. d'une fonct. déclarée FORWARD ne peut être redécl.
123 il manque le type du résultat dans la déclaration de la fonction
124 F-format pour les réels seulement
125 erreur dans le type des paramètres de la procédure standard

126 le nombre de paramètres est en desaccord avec la déclaration
127 substitution de paramètres illégale
128 résultat en desaccord avec la déclaration
129 conflit dans le type des opérandes
130 l'expression n'est pas du type ensemble

131 seuls les test d'égalité sont permis
132 l'inclusion stricte non autorisée
133 comparaison de fichiers non permise
134 type des opérandes illégal
135 le type des opérandes doit être booléen

136 le type des éléments d'un ensemble doit être scalaire ou sous-ens.
137 les types des éléments des ensembles doivent être compatibles
138 le type de la variable n'est pas tableau
139 le type de l'indexe n'est pas compatible avec la déclaration
140 le type de la variable n'est pas enregistrement

141 le type de la variable doit être fichier ou pointeur
142 solution avec un paramètre illégal
143 variable de contrôle de boucle de type illégal
144 expression de type illégal
145 conflit de types

146 affectation de fichiers interdite
147 type de l'étiquette incompatible avec l'expression qui sélectionne
148 les limites d'un sous ensemble doivent être scalaires
149 le type de l'indexe doit être entier
150 l'affectation à une fonction standard non permise

151 l'affectation à une fonction formelle n'est pas permise
152 pas de tel champ dans cet enregistrement
153 erreur de type dans la lecture
154 le paramètre actuel doit être une variable
155 la variable de contrôle ne peut pas être formelle ou non-locale

156 étiquette de CASE définie plusieures fois
157 trop de cas dans CASE.
158 pas de telle variante dans cet enregistrement
159 étiquette réelle ou STRING non permises
160 la déclaration précédente n'était pas anticipée

161 déclaré de façon anticipée à nouveau
162 la taille de paramètres doit être constante
163 variante non présente dans la déclaratio,
164 substitution d'une proc/fonct. standard non permise
165 étiquette définie plusieures fois

166 étiquette déclarée plusieures fois
167 étiquette non déclarée
168 étiquette non définie
169 erreur dans l'ensemble de base
170 paramètre valeur attendu

171 le fichier standard a été re déclaré
172 fichier extérieur non déclaré
173 fonction ou procédure PASCAL attendue

---

182 imbrication des unités non permise
183 déclaration externe pas permise à ce niveau d'imbrication
184 déclaration externe non permise dans la partie interface
185 déclaration de segment non permise dans une unité

186 étiquettes non permises dans la partie interface
187 tentative d'ouverture de librairie sans succès
188 unité non déclarés au préalable par USES
189 'USES' non autorisé à ce niveau d'imbrication
190 unité pas trouvée dans la librairie

191 pas de fichier privé
192 'USES' doit être dans la partie interface
193 pas assez de place pour cette opération
194 le commentaire doit apparaître au sommet du programme
195 l'unité ne peut être importée

201 erreur dans le nombre réel -chiffre attendu
202 la constante STRING ne doit pas excéder la ligne source
203 la constante entière dépasse les limites
204 8 ou 9 en base octale

250 trop d'étendue des identificateurs imbriqués
251 trop de procédures ou fonctions imbriquées
252 trop de références anticipées de procédures
253 procédure trop longue
254 trop de constantes longues dans cette procédure

256 trop de références externes
257 trop d'éléments externes
258 trop de fichiers locaux
259 expression trop compliquée

300 division par 0
301 pas de cas prévu pour cette valeur
302 expression de l'indice au dela des limites
303 la valeur à affecter dépasse les limites
304 expression de l'élément en dehors des limites

350 pas de segment de données alloué
351 segment utilisé deux fois
352 pas de segment de code alloué
353 unité non intrinsèque appelée comme intrinsèque

398 limitation d'implémentation
399 limitation d'implémentation

400 caractère dans le texte illégal
401 fin de l'entrée inattendue
402 erreur dans l'écriture du fichier code, pas assez de place
403 erreur de lecture comprenant le fichier
404 erreur d'écriture du fichier, pas assez de place
405 appel non autorisé dans une procédure séparée

406 fichier inclus non légal
407 trop de librairies

ERREURS D'ENTREE/SORTIE

0 pas d'erreur
1 mauvais bloc. Erreur de parité (CRC)
2 mauvais numéro d'unité
3 mauvais mode, opération illégale
4 erreur de matériel, non définie

5 unité perdue, unité plus branchée
6 fichier perdu, fichier plus dans la table des matières
7 mauvais titre, nom de fichier illégal
8 plus de place, espace insuffisant
9 pas d'unité, pas de tel volume en ligne

10 pas de fichier, pas de tel fichier dans le volume
11 fichier dupliqué
12 non fermé, tentative d'ouverture d'un fichier ouvert
13 non ouvert, tentative d'accès à un fichier fermé
14 mauvais format, erreur en lisant un réel ou un entier

15 débordement de la mémoire tampon
16 erreur de protection d'écriture.

ERREURS D'EXECUTION

0 erreur système (fatale)
1 indexe non valide, valeur en dehors des limites
2 pas de segment, maivais fichier code
3 procédure non présente au moment de la sortie
4 dépassement de la pile

5 dépassement d'entier
6 division par 0
7 référence mémoire incorrecte
8 interruption utilisateur
9 erreur entrée/sortie système

10 erreur entrée/sortie utilisateur
11 instruction non implémentée
12 erreur de point flottant
13 chaine de caractères trop longue
14 arrêt, point d'arrêt (sans présence du programme DEBUG)

15 mauvais bloc.

VOLUMES ET CONNECTEURS

NUMERO ET NOM DES VOLUMES

numero	nom	description du périphérique
=0:		inutilisé
=1:	CONSOLE	écran et clavier avec écho à l'écran
=2:	KEYBOARD	lit le clavier sans écho
=3:		inutilisé
=4:	nom de disquette	disquette d'initialisation (con. 6, un.1)
=5:	nom de disquette	2ième unité de disque (con.6, unité 2)
=6:	PRINTER	imprimante
=7:	REMIN	entrée à distance
=8:	REMOUT	sortie à distance
=9: à =12:	nom de disquette à nom de disquette	unites de disques supplémentaires (con. 4,5)

La demande Volume de Filer appelle =2: SYSTERM cependant.

CONNECTEURS DES EQUIPEMENTS PERIPHERIQUES

num		
0	carte langage	stockage de l'interpréteur du P CODE et le système d'entrée sortie
1	imprimante (carte d'interface de com communication, carte interface série ou carte interf. imprim. parallèle)	PRINTER ou =6:
2	modem, communication APPLE-APPLE etc.. (carte interface de communication ou carte interface série)	REMIN: ou =7: remout: ou =8:
3	console esterne (carte d'interface de communication. La carte d'interface série est tolérée).	CONSOLE: ou =1:(avec écho) KEYBOARD: ou =2: (sans écho)
4	unités de disque supplémentaires (carte de control eur de disque)	l'unité 1 est =9: l'unité 2 est =10: (ou le nom de la disquette)
5	unités de disque supplémentaires (carte de controleur de disque)	l'unité 1 est =11: l'unité 2 est =12: (ou le nom de la disquette)
6	premières unités de disque (carte de controleur de disque)	l'unité 1 est =4:. (ou le nom de la disquette)'le système est initialisé par la disquette placée ici) l'unité 2 est =5: (ou le le nom de la disquette)
7	ne doit pas contenir de carte de controleur de disque.	

Remarque 0 et 6 doivent impérativement être placés à ces endroits.

LE CODE ASCII

Voici la correspondance entre un caractère et son numéro. Nous avons aussi indiqué en tête (de 0 à 31) des caractères ne figurant pas toutes sur notre clavier, mais faisant partie de ce code.

0	NUL	32	SP	64		96	'
1	SOH	33	!	65	A	97	a
2	STX	34	"	66	B	98	b
3	ETX	35	#	67	C	99	c
4	EOT	36	S	68	D	100	d
5	ENQ	37	%	69	E	101	e
6	ACK	38	&	70	F	102	f
7	BEL	39		71	G	103	g
8	BS	40	(	72	H	104	h
9	HT	41	)	73	I	105	i
10	LF	42	x	74	J	106	j
11	VT	43	+	75	K	107	k
12	FF	44	,	76	L	108	l
13	CR	45	-	77	M	109	m
14	SO	46	.	78	N	110	n
15	SI	47	/	79	O	111	o
16	DLE	48	0	80	P	112	p
17	DC1	49	1	81	Q	113	q
18	DC2	50	2	82	R	114	r
19	DC3	51	3	83	S	115	s
20	DC4	52	4	84	T	116	t
21	NAK	53	5	85	U	117	u
22	SYN	54	6	86	V	118	v
23	ETB	55	7	87	W	119	w
24	CAN	56	8	88	X	120	x
25	EM	57	9	89	Y	121	y
26	SUB	58	:	90	Z	122	z
27	ESC	59	;	91		123	
28	FS	60		92		124	
29	GS	61		93		125	
30	RS	62		94	^	126	
31	US			95	_	127	DEL

LE CLAVIER

LES TOUCHES DE CARACTERES

Passer en mode Edit puis Insert.

- à l'intérieur du trait gras de la figure 43 page 37, les caractères apparaissent tels qu'il figurent sur les touches.
- attention O est distinct de 0 et I de 1
- les caractères du dessus s'obtiennent en maintenant shift appuyé puis en pressant la touche.
- en plus s'obtient par (CTRL M) et par (SHIFT M)

LES TOUCHES DE FONCTIONS

Déplacement du curseur

~~déplace à droite d'une case~~

→ déplace à droite d'une case

← déplace à gauche d'une case

(CTRL O) déplace vers le haut d'une ligne

(CTRL L déplace vers le bas d'une ligne

(CTRL I) tabule (déplace par sauts) de 8 espaces

RETURN va à la ligne

P déplace le curseur de 24 lignes

Autres fonctions

ESC quitte le mode où l'on se trouve (Insert ou Del)

RESET reinitialise le système

REPT répète la touche que l'on maintient enfoncée

(CTRL A) visualise l'autre moitié de l'écran

(CTRL Z) déroule l'écran pour suivre le curseur

(CTRL C) entérinne l'insertion ou l'effaage.

(CTRL X) efface toute la ligne, en mode Insert.

(CTRL Q) annule le décalage automatique

INITIALISATION ET COPIE DE DISQUETTE SYSTEME A DEUX UNITES DE DISQUE

vous allez:	APPLE réagit:
INITIALISATION	
- débrancher APPLE (fig. 3.6)	
- insérer APPLE1 dans l'unité 1 et APPLE2 dans l'unité 2	
- brancher le moniteur	
- brancher APPLE (fig. 3.6)	-> APPLE caquette, puis au bout de moins d'une minute apparait la ligne de sélection globale
FORMATTAGE	
- vous placer au niveau de sélection globale	
- insérer APPLE3 dans l'unité 1 et la disquette à formatter dans l'unité 2	
- taper X	-> EXECUTE WHAT FILE ?
- taper APPLE3:FORMATTER RETURN	-> FORMAT WHICH DISC ?
- taper #5: RETURN	-> formatte la disquette et lui donne le nom de BLANK.
	-> FORMAT WHICH DISC ?
- terminer en répondant RETURN	
COPIE DE LA DISQUETTE DEMO SUR BLANK	
- placer DEMO dans l'unité 1 et BLANK dans l'unité 2.	
- depuis le niveau de sélection global global, taper F et T	-> TRANSFER ?
- taper DEMO: RETURN	-> TO WHERE ?
- taper BLANK: RETURN	-> TRANSFER 280 BLOCS ?
- taper Y	-> effectue le transfert et annonce DEMO: -- BLANK:

Exemple

```
PROGRAM  SCAL;
  CONST  N=20;
  TYPE   DIMENSION = 1..N;
         VECT = ARRAY[DIMENSION] OF REAL;
  VAR    U,V  : VECT;
         UV   : REAL;
         I    : DIMENSION;
  BEGIN
    UV:=0;
    WRITELN ('PRODUIT SCALAIRE')
    FOR I:= 1 TO N DO
      BEGIN
        READLN (U[I], V[I]);
        UV:= UV + U[I]*V[I]
      END;
    WRITELN ('PRODUIT SCALAIRE', UV)
  END.
```

→ Ce texte est écrit dans l'ÉDITEUR de TEXTE et stocké dans SCAL.TEXT

→ Il est ensuite COMPILE (et lié aux procédures standard) et donne naissance à un programme-objet en langage machine appelé SCAL.CODE directement exécutable.

En cas d'erreur de programmation, on revient à l'éditeur de texte et on modifie SCAL.TEXT, qu'on recompile.

Pour ces manipulations voir le fascicule PASCAL Mise en Service

→ les caractères utilisés - opérateurs

- Alphabet + chiffres + caractères de ponctuation
- + - * DIV (division entière) MOD (reste de la division / division réelle , < , <= , >= , <>
- AND , OR , NOT

Quelques instructions

→ Instructions de déclaration

- les constantes : CONST PI = 3.14 ; TAUX = Ø.Ø3
- types de données : INTEGER REAL CHAR BOOLEAN
  STRING
  + Types personnalisés
- variables : VAR I,J : INTEGER;
  M : ARRAY [1..5, 1..7] OF RE
  V : PACKED ARRAY [1..7] OF CHA
  S : STRING

! • commentaire placé n'importe où entre { }

→ Ordres PASCAL exécutables

- ils sont delimités par une séquence BEGIN
  ⋮
  END; (ou END. pc
  la dernière)

- (;) est le séparateur d'instructions (:=) affecte des valeurs aux va

! (toute instruction est suivie de ; sauf si elle est suivie de END, ELSE ou UN

- Lecture-ecriture

READ
ou READLN (A, B, U[I]) lecture au clavier et stockage dans les zones correspondantes en outre READLN précise la saisi sur une "ligne" de l'écran

WRITE
ou WRITELN (A, B, 'FIN') écriture sur l'écran (eventuellement sur une ligne)

le cadrage des résultats peut être programmé :

WRITE (K:3) affiche, si K:=5, ØØ5

WRITE ('BONJOUR', X:1Ø:4) affiche, si X:=25.673,

4

BONJOUR, , , 25.6738Ø,

Structures alternatives

- IF condit THEN bloc1 (ELSE bloc2);

  bloc1 et bloc2 sont des séquences d'instructions pouvant être délimitées par BEGIN .... END; et s'emboiter.

exemples: • IF N=1 THEN WRITELN (I, J)
ELSE BEGIN
I := 0 ;
J := 0 ;
WRITELN (I,J)
END;

• IF N=0 THEN WRITELN (I,J);
(pas d'alternative)

- CASE var OF
  v1 : bloc1
  v2 : bloc2
  ⋮
  END;

branchement au bloc i si la variable var prend la valeur v_i

exemples: (remise selon type de client A, B, C ou D

```
VAR LC : CHAR ;
    MONTANT , APAYER : REAL ;
.....
CASE LC OF
  'A','C' : APAYER := 0.9 * MONTANT ;
  'B'     : BEGIN
              IF MONTANT > 1000.0
                 THEN APAYER := 0.95* MONTANT
                 ELSE APAYER := MONTANT
            END ;
  'D'     : APAYER := 0.95 * MONTANT
END;
```

Instructions repetitives

- FOR I:=1 TO N DO
  BEGIN
  ⋮
  END;

séquence répétitive contrôlée par I

- WHILE condition DO
  BEGIN
  ⋮
  END;

séquence répétitive exécutée tant que la condition est vraie

- REPEAT
  ⋮
  UNTIL condition

séquence répétitive contrôlée jusqu'à ce que la condition soit vraie

Exemple

→ WHILE A>B DO A:=A-B

→
```
PROGRAM PGCD;
VAR A,B,X,Y : INTEGER;
BEGIN
  READ (A,B); X:=A ; Y:=B ;
  REPEAT
    WHILE A>B DO A:=A-B;
    WHILE A<B DO B:=B-A;
  UNTIL A=B;
  WRITELN ('PGCD DE ', X, ' ET DE ', Y, '= ', A
END.
```

Les procédures et fonctions

- permet le regroupement d'instructions appelés en bloc (pendant du GOSUB en BASIC)

```
PROCEDURE  nom  (param1, param2, ....)
  BEGIN
    :
  END;
```

(paramètres : éventuellement)

La procédure est simplement APPELEE par son nom (éventuellement suivi de paramètres variable ou constantes)

exemple : Procédure de tracé d'une ligne contenant N fois un caractère contenu dans C

```
PROCEDURE TRACE (C:CHAR , N:INTEGER);
  CONST MAX = 100;
  BEGIN
    IF N>MAX THEN N:=MAX;
    FOR I:=1 TO N DO
      WRITE(C);
    WRITELN
  END;
```

→ Un programme principal contiendra dans l'ordre

1) les instructions de déclaration
2) la (ou les) procedure(s)
3) les instructions d'appel et la suite du programme

par :

```
BEGIN
  N:= 120
  WRITELN(N);
  TRACE('*', N)
  WRITELN('FIN');
END.
```

donne { 120
***..... *** (19
FIN

fonctions - récursivité

- permet de définir ses propres fonctions d'une ou plusieurs variables (arguments). le résultat est nécessairement un scalaire.

ex:
```
FUNCTION ATG (X:REAL):REAL;
BEGIN
ATG := COS(X)/SIN(X)
END;
```

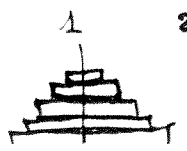
Programme principal :
```
MTT := 2;
⋮
WRITELN ATG(MTT);
⋮
```

! les fonctions standard sont, avec la signification classique :
ABS, SQR (carré), SIN, COS, ARCTAN, EXP, LN
SQRT (racine carré), TRUNC (partie entière avec troncation)
ROUND (arrondi), ODD (vrai si nbre pair)

- Il y a possibilité de RECURSIVITE
une procédure ou une fonction peuvent s'appeler elles-mêmes.

ex:
```
FUNCTION FACTORIELLE (N:INTEGER): INTEGER;
BEGIN
  IF N=0 THEN FACTORIELLE := 1
         ELSE FACTORIELLE := FACTORIELLE (N-1) *
END;
```

(les adresses de retour des sous-programme sont stockées dans une PILE (d'où les appels emboités), ainsi que les valeurs successives des arguments et variables internes aux sous-programme



exercice Programmer les tours de Hanoï à l'aide de la procédure DEPLACER(N,I,J) qui déplace N disques du piquet I au piquet J (DEPLACER(1,I,J) déplace le disque I

le mode graphique la TORTUE TURTLEGRAPHICS

• l'écran est à haute résolution 192 lignes x 280 colonnes

exemple :

```
USES TURTLEGRAPHICS ;
VAR I : INTEGER ;
BEGIN
   INITTURTLE ;
   PENCOLOR (GREEN) ;
   FOR I:=1 TO 4 DO
      BEGIN
         MOVE (60) ;
         TURN(90) ;
      END ;
END.
```

Ce programme dessine un carré 60 x 60

Lexique

INITTURTLE	vide l'écran, place la tortue au centre
TURN (n)	tourne la tortue de n degrés (sens trigono.
MOVE (m)	déplace la tortue de m positions dans la direction où elle se trouve pointée par TURN
TURNTO (n)	tourne la tortue d'un angle absolu de n degrés
MOVETO (x,y)	déplace la tortue au point de coordonnées x, y
PENCOLOR (couleur)	choix de la coloration de trajectoire
VIEWPORT (G, D, B, H)	délimite une fenêtre dans l'écran ex VIEWPORT (5, 275, 180, 12)
FILLSCREEN (couleur)	remplit l'écran de la couleur
FILLSCREEN (BLACK)	vide l'écran

TURTLEX, TURTLEY, TURTLEANG fournit les coordonnées (et angle) actuelles de la tortue

quelques fonctions et procédures particulières

KEYPRESS

saisie "à la volée" d'un caractère, est validée dès que l'on frappe une touche du clavier.

exemple d'usage :

```
    .
    .
    .
• REPEAT
    .
    .
    .
  UNTIL KEYPRESS
```

procédure d'attente exécutée tant qu'il n'y a pas de réponse au clavier

• IF KEYPRESS THEN WRITELN 'OK'

Si l'on veut récupérer le caractère frappé il faut consulter le fichier KEYBOARD par ex :

IF KEYPRESS THEN READ (KEYBOARD, TOTO)

la zone TOTO (déclarée CHAR) reçoit ce caractère

WAIT (n)

interrompt le programme pendant n cycles

RANDOM contient un entier compris entre 0 et 32767

exemple : WRITE (RANDOM)

EXIT (nom) permet de quitter la procédure nom

la "variable" SCREENBIT (X, Y) est vraie si le point X, Y est occupé (i.e. sur la trajectoire décrite par la tortue), faux sinon.

ex: IF SCREENBIT(10, 280) THEN PENCOLOR (GREEN)

WCHAR (var) ou WSTRING (chaine) permet d'afficher la variable var ou la chaine au point où se trouve la tortue

ex:
```
PENCOLOR(RED)
MOVETO (137, 90)
WCHAR ('X')
```
écrit un 'x' au point de coordonnées 137, 90

(la tortue est déplacée de 7 points par caractère affiché vers la droite)

CHARTYPE (n) permet de préciser la présentation de ce caractère ex: n = 10 caractère blanc sur fond noir inverse

n = 5

n = 6 superposition de caractère sur l'image existante (utile pour des messages sur graphique)

Les raquettes sont accessibles par PADDLE(0) et PADDLE(1)
la valeur est comprise entre 0 et 255

exemple: WRITELN (PADDLE(0));

mode graphique et musique

- mode graphique basse résolution 24 x 40

```
USES APPLESTUFF;
  GOTOXY (X, Y)      0 ≤ X ≤ 39
                     0 ≤ Y ≤ 23   déplace le curseur
  WRITE ('U')        écrit U à cette endroit
  WRITE (' ')        efface
  PAGE (OUTPUT)      vide l'écran
```

- musique

```
USES APPLESTUFF;
NOTE (niveau, durée)      0 ≤ durée ≤ 255
                          0 ≤ niveau ≤ 50
```

exemple :

```
PROGRAM GAMME;
USES APPLESTUFF;
VAR NIVEAU, DUREE : INTEGER;
 BEGIN
   DUREE := 100;
   FOR NIVEAU := 12 TO 24 DO
     NOTE (NIVEAU, DUREE)
 END.
```

LE LANGAGE PASCAL

Bibliographie

IREM = disponibles au "Centre Informatique" IREM

En français

(IREM) Découvrez PASCAL sur l'APPLE II
John COLIBRI. Mnémodyne 1980 384

- Le langage PASCAL
J.M. Crozet, D. Serain. Masson 1980 - 215 p

(IREM) Programmer en PASCAL
D.J. David, J.L. Deschamps. PSI 1980 - 159 p

- Le langage de programmation PASCAL
Ph. Kruchter. Eyrolles 1979 - 93 p

- PASCAL 1. Manuel de base, 2. Techniques de programmation et concepts avancés
P. Lignelet. Masson 1980 - 279 et 229 p

- Le langage PASCAL
N. Magnenat - Thalmann et al. Gaëtan Morin - Dunod 1979 - 329 p

(IREM) Introduction au PASCAL
P. Le Beux. SYBEX 1980 - 491 p

- Introduction à la programmation systématique
N. Wirth. Masson 1977 - 157 p
(Niklaus Wirth est le créateur de PASCAL)

Bibliographie (suite)

En anglais

(IREM) Manuels d'utilisation APPLE II PASCAL
(langage et système d'exploitation) + Apple PASCAL Update version 1.1
Apple PASCAL Addendum

- Microcomputer problem solving using PASCAL
K.L. Bowles. Springer Verlag 1977 - 563 p
Bowles est le créateur de la version UCSD (university of California San Diego) de Pascal implémentée sur Apple II, en particulier les fonctions graphiques puissantes et souples.

(IREM) PASCAL user manual and report.
K. Jensen. Niklaus Wirth. Springer Verlag 1978.
Définition du langage par son créateur, c'est l'ouvrage de référence.

- Structured programming and problem solving with PASCAL
R.B. Kieburtz. Printice Hall 1978 - 365 p

- PASCAL an introduction to methodical programming
W. Findlay & DA Watt. Pitman, 306 p

- The Byte book of PASCAL
B.W. Liffick. Byte 1979. 333p.
Pour ceux que cela intéresse, p 107-155, important article sur l'écriture d'un programme d'échecs dont le listage est fourni en PASCAL CDC (3600 lignes !)