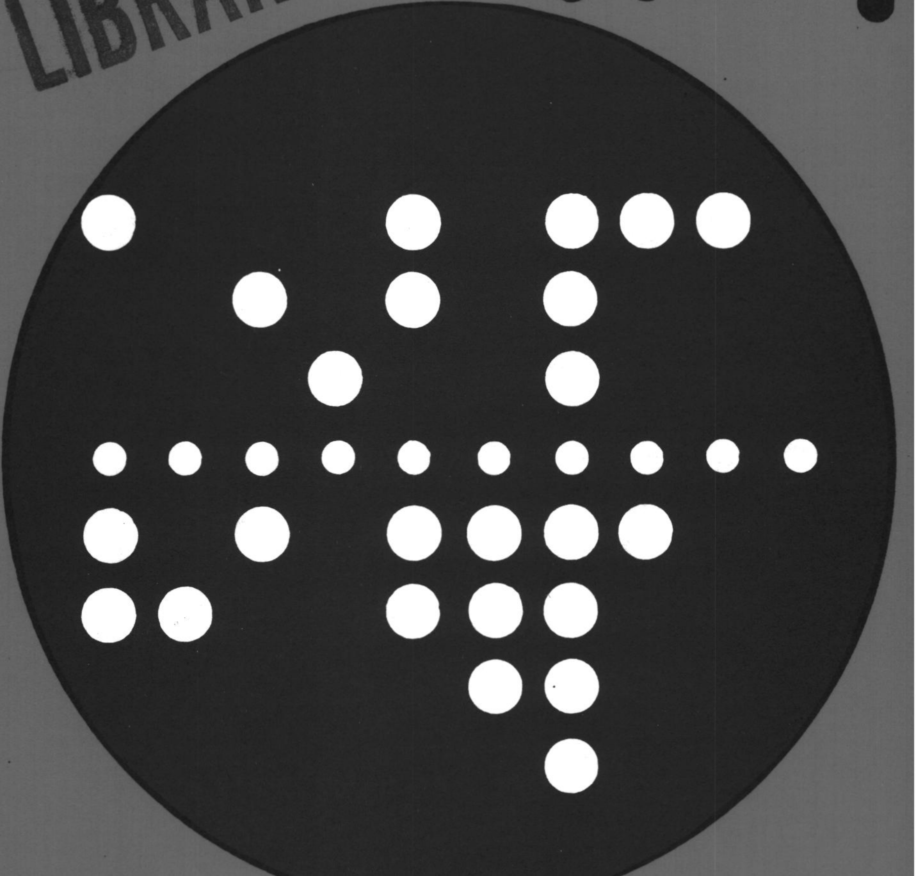
Commission of the European Communities

Joint Research Centre - Ispra

Computing Centre Newsletter



October 1976 ● No 5

Contents

Note of the Editor

The present Newsletter will be published monthly except for August and December.

The Newsletter will include:

- Developments, changes, uses of installations
- Announcements, news and abstracts on initiatives and accomplishments.

The Editor thanks in advance those who will want to contribute to the Newsletter by sending articles in English or French to one of the following persons of the Editorial Board.

Note de la Rédaction

Le présent Bulletin sera publié mensuellement excepté durant les mois d'août et décembre.

Le Bulletin traitera des:

- Développements, changements et emploi des installations
- Avis, nouvelles et résumés concernant les initiatives et les réalisations.

La Rédaction remercie d'avance ceux qui voudront bien contribuer au Bulletin en envoyant des articles en anglais ou français à l'un des membres du Comité de Rédaction.

Editor : Sylvia R. Gabbai
Layout : Paul De Hoe
Graphical and Printing Workshop, JRC Ispra

Acknowledgement should be given for their technical support to Mr. E. Eiselt, Mrs. M.G. Giaretta, Mrs. M. Van Andel, Mr. G. Clivio, A. Margnini, G. Zurlo

Le Coût du Centre de Calcul
Années 1973 - 1976

J. Pire

Le Centre de Calcul a été créé pour fournir au Centre Commun de Recherche l'instrument nécessaire au traitement automatisé des données scientifiques, techniques et administratives.
Il n'a cependant pas de dotation propre, mais travaille dans le cadre d'une limite supérieure de dépenses en matériel et personnel (budget) qui doivent être compensées par les facturations des prestations fournies aux Objectifs de Recherche, à l'Administration du Centre de Recherche et pour une moindre partie (15%) à des tiers ayant avec le CCR des contrats de recherche ou de support.

La gestion est effectuée suivant une méthode appelée "compte d'affectation".
Une des caractéristiques de cette méthode de gestion est de devoir clôturer annuellement ses comptes sans passif ni actif. Cette caractéristique implique que tout investissement doit être "amorti" en un an, ce qui n'est pas évidemment sans poser de gros problèmes.

Les dépenses du Centre de Calcul se répartissent de la façon suivante:

1. Crédits primaires couvrant:

1.1 les heures supplémentaires ou indemnités spéciales du personnel
1.2 les dépenses de fonctionnement matériels
1.2.1 coût du matériel inventoriable
1.2.2 coût de la maintenance du matériel inventoriable
1.2.3 la location et maintenance du matériel
1.2.4 la location de logiciel
1.2.5 l'achat de matières de consommation courante
1.2.6 le coût de la force motrice
1.3 les contrats de services extérieurs

2. Crédits secondaires:

2.1 coût du personnel
2.2 participation aux frais d'infrastructure de l'Etablissement d'Ispra.

Le tableau I donne les budgets des années 1973 à 1976.

Tableau I

	1973	1974	1975	1976
Primaires				
1.1 heures supplément.du pers.	12.000	12.600	7.056	16.600
1.2 dépenses de fonction.techn.	1.852.274	1.895.000	2.064.000	2.100.671
1.3 dépenses par contrat	97.000	95.000	20.000	30.000
	1.961.274	2.002.600	2.091.056	2.147.271
Secondaires				
2.1 utilisation du personnel	289.585	347.513	482.745	599.025
2.2 utilisation de l'infrastruc.	156.875	186.195	262.840	287.525
	446.460	533.718	745.585	886.552
Total	**2.407.734**	**2.536.318**	**2.836.631**	**3.033.823**

Le tableau II donne l'effectif maximum prévu pour le Centre de Calcul, il faut cependant noter que le personnel réel a toujours été en sous-nombre.

Tableau II

	1973	1974	1975	1976
Fonctionnaires et Agents d'Etablissment	22	22	22	23
Agents locaux	5	5	13	10
Total	**27**	**27**	**35**	**33**

Une rapide analyse de ces chiffres montre que comparativement au budget total de 1973, celui de 1976 est en augmentation de 26%, mais cette augmentation est due presqu'en totalité à l'augmentation des crédits secondaires.

Les crédits primaires n'ont subi qu'une augmentation relative de 7%, alors que le coût du matériel (papier) a plus que doublé et la plupart des prix de location ont eu une augmentation de plus de 36%.

Services rendus

Les services rendus par le Centre de Calcul peuvent se répartir en trois types:

- Calculs proprement dits
- Opérations périphériques
- Location de matériel pour compte d'utilisateurs identifiés (disques, terminaux, etc.)

En 1973 tout le coût de l'installation était pratiquement porté à charge des utilisateurs de temps de calcul.
A partir de 1974, les charges ont été réparties selon le type de service reçu.
Le tableau III donne le prix de l'unité d'oeuvre tel qu'il a été calculé après clôture des comptes annuels, le nombre d'unité d'oeuvre facturé et la valeur totale ainsi facturée.

Tableau III

Année	1973	1974	1975	1976
UC	1.360	1.080	920	750*
Heures facturées	1.672	2.142	2.321	3.096*
Valeur total en K. U.C.	2.274	2.313	2.135	2.322

**valeur estimée en octobre*

Ces quelques chiffres montrent clairement l'amélioration de l'utilisation de l'installation au cours des différentes années.

Considérations sur le prix de l'unité d'oeuvre

Le budget du Centre de Calcul est exprimé en U.C.
Au moment des payements, les U.C. sont converties en monnaies nationales de payement suivant un cours officiel fictif.
Le tableau IV donne la valeur de l'U.C. dans les différentes monnaies nationales selon le taux fictif officiel et le taux de change bancaire moyen établi par la DG XIX pour la période du 1 au 31 octobre 1976.

Le personnel et pratiquement toutes les dépenses primaires étant payées en Lires Italiennes, le Centre de Calcul subit une perte de change de 40,6% (au profit de la Commission).

Tableau IV

1 U.C. =

	Cours	Cours bancaire	Rapport
	officiel	moyen	de
Monnaies nationales	1	2	1/2
Couronne Danoise	7.50	7.46	1.008
Mark Allemand	3.66	3.10	1.181
Livre Sterling	0.417	0.714	0.584
Franc Français	5.55	6.097	0.911
Lire Italienne	625.	1052.6	0.594*
Florin	3.62	3.26	1.110
Franc Belge	50.	50.	1
Dollar U.S.	1.25	1.25	1

**** Le Centre de Calcul reçoit 625 Lires par U.C***

A titre indicatif, le coût estimé de l'heure de calcul pour 1976 est 750 U.C. soit 768.750 Lires Italiennes (estimé au même cours de change que celui utilisé pour les payements du Centre de Calcul).

Le tableau V fournit la conversion de ce prix dans les autres monnaies, selon qu'on retransforme les Lires selon le cours fictif ou le cours bancaire.

Tableau V

	750 U.C. =	768.750 Lit
Monnaies nationales	Cours officiel	Cours bancaire
Couronne Danoise	5.595	3.322
Mark Allemand	2.745	1.380
Livre Sterling	313	318
Franc Français	4.166	2.715
Florin	2.715	1.452
Franc Belge	37.500	22.266

Il est clair que les prix convertis au cours fictif officiel seraient difficilement acceptés; par contre ceux établis selon les cours bancaires sont compétitifs dans l'ensemble de la Communauté.

Catalogued Procedure PLPCLGS

C.L. van den Muyzenberg

The procedure is basically the same as PLPCLG; compile link and go using the PL/I optimizing compiler (all symbolic parameters in PLPCLG are also present in PLPCLGS).

Added possibilities are:

1. using structured programming macros,
2. writing the linkage editor output (load module) on a private library (with possibility of updating an old load module),
3. printing only part of the compiler output and printing a cross-reference list between line numbers of the original input and statement numbers,
4. compressing a partitioned data set (used for macro instructions),
5. adding members to the PDS containing macro-instructions,
6. compressing the load module library.

The procedure contains 10 steps:

S0	..	dummy step to set the condition code = 0
S1	..	prepare control cards for step S2
S2	..	scratch members of the user macro library
S3	..	compress the user macro library
S4	..	update the user macro library
CMP	..	compile
S5	..	print selected compiler output
S6	..	compress the user load module library
LKED	..	linkage editor
GO	..	execution.

Standard Use: // EXEC PLPCLGS

This gives the same execution as PLPCLG, a macro library is defined containing structured programming macros. To use these macros:

1. use cards * PROCESS M,IS; before each separately compiled procedure,
2. use%INCLUDE STRPLI; or //INCLUDE STRPLI,.. at the beginning of each separately compiled procedure (before or immediately after the PROCEDURE statement).

For standard use with a user loadmodule library, see **Linkage editor.**

Execution of Steps S1 (and S2), S3, S4, S5, S6

Write as parameter in the EXEC statement: CSn = 1

Ex.: // EXEC PLPCLGS,CS5=1 to execute step S5.

Execution of Steps CMP, LKED, GO

These steps are executed by default. If you do not want to execute them, write as parameter in the EXEC statement: CSstepname=0

Ex.: // EXEC PLPCLGS,CSCMP=0,CSLKED=0,CSGO=0 to suppress the 3 steps

User Macro Library: concatenated with the SYSLIB macro library.

The user macro library must have a BLKSIZE < = 3120.
Use the parameters MACL (dsname) and MACV (volume, default value is USER01) to specify your library.

Ex.: // EXEC PLPCLGS,MACL='SYS1.MYLIB',MACV=USER03

This will specify SYS1.MYLIB as a macro-library concatenated to the standard library residing on volume USER03.
%INCLUDE MYMAC; will include macro MYMAC.

If you want to use a **separate** macro library, add a DD statement to the CMP step.

Ex.:
```
// EXEC PLPCLGS
//CMP.MACRO DD DSN=SYS1.MYLIB,VOL=SER=USER01,
/        UNIT=3330,DISP=SHR
```

This will specify SYS1.MYLIB as a separate macro library.
%INCLUDE MACRO (MYMAC); will include macro MYMAC

A separate macro library may be used if it is necessary to use a user macro with the same name as a standard macro, that is not to be used.

Scratch Members of the user macro library

Execution is in 2 steps:

1. a program that reads MEMBER names and generates the control cards for step 2 (do not forget the ; after each member specification),
2. the IEHPROGM program that scratches the member.

Ex.:
```
// EXEC PLPCLGS,S1=1,MACL='SYS1.MYLIB',MACV=USER01
//S1.SYSIN DD *
  MEMBER='MYMAC'; MEMBER='MYMAC2';
```
This will result in scratching members MYMAC and MYMAC2 from library SYS1.MYLIB on volume USER01.

Compress the Macro Library

If members of a partitioned data set are replaced or deleted, the space that is freed will only be available after a compress of the data set.

Ex.: // EXEC PLPCLGS,CS3=1,MACL='SYS1.MYLIB'

This specifies that in step S3 the IEBCOPY program must make a compress of partitioned data set SYS1.MYLIB.

Update the User Macro Library

The operations permitted are:

1. adding new members,
2. replacing existing members,
3. changing existing members.

For the syntax of the control cards and a detailed description of the possibilities see:

IBM SYSTEM/360 OPERATING SYSTEM, UTILITIES GC28-6586
IEBUPDTE program.

Ex.:

```
// EXEC PLPCLGS,CS4=1,MACL='SYS1.MYLIB',MACV=USER01
//S4.SYSIN DD *
./ ADD NAME=MEMB1,LEVEL=00,SOURCE=0,LIST=ALL
./ NUMBER NEW1=10,INCR=10

   MEMB1 CARD DECK

./ REPL NAME=MEMB2,LEVEL=01,SOURCE=0,LIST=ALL
./ NUMBER` NEW1=10,INCR=10

   MEMB2 CARD DECK

./ ENDUP
```

This will result in updating library SYS1.MYLIB on volume USER01:

1. member MEMB1 is added, printed and numbered,
2. member MEMB2 is replaced by a new deck, printed and numbered.

N.B.: Records in a macro library should contain in cols. 73/80 either spaces, or a sequence number (with cols. 73/75='000' or cols. 73/75 = spaces).

Compilation as usual.

See also the preceding paragraphs **Standard Use** and **User Macro Library.**

Print Selected Compiler Output

To obtain a selected print, it is necessary to redefine the compiler print data set and to define the input data set for the print program.

Ex.:

```
UDSNP='UNIT=SYSSQ,DSN=&&PRINT',
DSPP='DISP=(,PASS),SPACE=(CYCL,(1,1))',
DCBP='DCB=(RECFM=VBA,LRECL=125,BLKSIZE=129)',
DSP=',DISP=(OLD,PASS)'
```

Furthermore it is necessary to specify execution of the print program and input data.

Ex.: CS5=1,INLIST='N',PRILIST='0101011'

The INLIST and Prilist values as given specify the default options and are not necessary.
Permitted values for INLIST and PRILIST are:

INLIST N no list of the original output,
Y a list of the original print output just as it would have been without using this program,
L a list of the original print output without carriage control (control characters will be printed),

PRILIST each of the 7 positions indicates a part of the original print output (0=not to be printed, 1=to be printed)

position 1 .. option list
2 .. preprocessor input (program)
3 .. preprocessor input (included text)
4 .. preprocessor error messages
5 .. source listing (input to the compiler after expansion of preprocessor macro instructions)
6 .. attribute, cross-reference list (if printed, identifiers generated by structured programming macros will be deleted)
7 .. compiler messages.

A cross-reference list will be printed containing the statement numbers of the source list (also if not printed) and the lines in the preprocessor input. To print this list correctly, the source deck should contain sequence nrs in cols. 73/80 (ols. 73/75 not containing spaces nor zeros). The included macro text records should contain in cols. 73/75 either spaces or zeros.

Ex.: using default options:

```
// EXEC PLPCLGS,CS5=1,
// UDSNP='UNIT=SYSSQ,DSN=&&PRINT',
// DSPP=',DISP=(,PASS),SPACE=(CYL,(1,1))',
// DCBP=',DCB=(RECFM=VBA,LRECL=125,BLKSIZE=129)',
// DSP=',DISP=(OLD,PASS)'
```

Compress the User Load Module Library

The replacement of load modules by new versions will cause free space to be not accessible. A compress makes the space available.

```
Ex.: // EXEC PLPCLGS,CS6=1,GOSET='SYS1.MYSOURCE',
     //          UNISP=3330,DSL=SHR
```

This will result in making a compress of data set SYS1.MYSOURCE.

Linkage Editor

The linkage editor permits not only (1) linking an entire program, but also (2) taking an existing load module, adding new procedures and substituting corrected procedures and finally replacing the old load module with the new one. Method (2) clearly gives the advantage of only having to compile new or changed procedures and not the whole program. To use this possibility:

1. define your load module library for inclusion of modules.

   Ex.: INCLIB='SYS1.MYSOURCE',INCV=USER01

2. define your load module library as output from the linkage editor.

   Ex.: GOSET='SYS1.MYSOURCE',GO=MYPROG,DSL=SHR,
   UNISP=3330

3. tell the linkage editor you want to include your old load module.

   ```
   Ex.: //LKED.SYSIN DD *
          INCLUDE INCLIB (MYPROG)
   ```

4. see also Execution.

The complete example (to compile part of the program, link with the old load module, substitute the new load module and execute the new load module) will be:

Ex.:

```
// EXEC PLPCLGS,INCLIB='SYS1.MYSOURCE',INCV=USER01,
// GOSET='SYS1.MYSOURCE',GO=MYPROG,DSL=SHR,
// UNISP='3330,VOL=SER=USER01',DSG=SHR
//CMP.SYSIN DD *

  PART OF THE PROGRAM TO BE COMPILED

//LKED.SYSIN DD *
 INCLUDE INCLIB (MYPROG)
//GO.SYSIN DD *

 INPUT DATA FOR THE PROGRAM
```

Execution as usual

If a user load module library is used, specify DSG=SHR, otherwise your library will be purged.
All possibilities of this catalogued procedure mau be used in the same invocation; this will necessitate writing a long list of parameters but to do the same work with separate programs you would need even more DD statements.

Ex.: using all possibilities

```
//    EXEC    PLPCLGS,MACL='SYS1.MYLIB',CS1=1,
//    CS3=1,CS4=1,CS5=1
//    UDSNP='UNIT=SYSSQ,DSN=&&PRINT',
//    DSPP=',DISP=(,PASS),SPACE=(CYL,(1,1))',
//    DCBP=',DCB=(RECFM=VBA,LRECL=125,BLKSIZE=129)',
//    DSP=',DISP=(OLD,PASS)',
//    CS6=1,GOSET='SYS1.MYSOURCE',
//    INCLIB='SYS1.MYSOURCE',DSG=SHR,
//    GO=MYPROG,DSL=SHR,UNIP='3330,VOL=SER=USER01'
//S1.SYSIN  DD  *
  MEMBER='MYMAC',MEMBER='MYMAC2';
//S4.SYSIN  DD  *
./    ADD  NAME=MEMB1,LEVEL=00,SOURCE=0,LIST=ALL
./    NUMBER  NEW1=10,INCR=10

      MEMB1  CARD  DECK

./    REPL  NAME=MEMB2,LEVEL=01,SOURCE=0,LIST=ALL
./    NUMBER  NEW1=10,INCR=10

      MEMB2  CARD  DECK

./    ENDUP
//CMP.SYSIN  DD  *

      PART  OF  THE  PROGRAM  TO  BE  COMPILED

//LKED.SYSIN  DD  *
  INCLUDE  INCLIB(MYPROG)
//GO.SYSIN  DD  *

      INPUT  DATA  FOR  THE  PROGRAM
```

For any further information or ideas to improve upon the cataloged procedure or its description:

C.L. Van den Muyzenberg

ERRATA CORRIGE: to "Structured Programming"
p. 3÷9, No. 4 Newsletter - September 1976

Corrections

Page 6	Newsletter, June 1976
Page 6	//CMP.SYSLIB OD DSN=SYS1.MACVDM, ...
Page 6	%INCLUDE STRPLI; (structured PL/I)
Page 7	structures 2 - 7
Page 10	*insert before Example:* use $ CLASS 1 to avoid trouble with big blocks
Page11	// EXEC COPYPDS,DSIN='SYS1.MYLIB',VOLIN=USER01,

References to Structures Programming

1. B.W. Boehm
   Software and its impact: a quantitative assessment
   Datamation 19, 5 (May 1973), 48-59
2. TOP-DOWN Structured Programming Techniques
   C.L. Mc Gowan, J.R. Kelly
   Petrocelli/Charter
   New York 1975

Statistics of computing installation utilization

Report of computing installation exploitation for the month of August

	YEAR 1976	YEAR 1975
Number of working days	22 d	20 d
Work hours from 8.00 to 24.00for	16.00 h	9.25 h
Duration of scheduled maintenance	25.40 h	15.00 h
Duration of unexpected maintenance	18.76 h	2.75 h
Total maintenance time	44.16 h	17.75 h
Total exploitation time	307.84 h	167.25 h
CPU time in problem mode	107.62 h	59.96 h
Teleprocessing:		
CPU time	1.72 h	0.72 h
I/O number	335.000	598.000
Equivalent time	4.06 h	4.90 h
Elapsed time	152 h	100 h
Batch processing:		
Number of jobs	7468	6328
Number of cards read	2388000	1746000
Number of cards punched	161000	116000
Number of lines printed	23346000	17897000
Number of pages printed	535000	399000

BATCH PROCESSING DISTRIBUTION BY CLASS

	A	1	2	3	4	5	D	TOTAL
Number of jobs	1107	2356	912	1520	399	91	398	6783
Elapsed time (hrs)	22	86	66	135	89	21	39	458
CPU time (hrs)	0.9	11.1	13.1	24.6	39.7	9.1	5.8	104.3
Equivalent time (hrs)	8.6	32.6	31.7	72.9	52.5	13.9	21.9	234.1
Turn around time (hrs)	0.6	0.8	1.3	0.7	1.5	1.7	1.2	0.8

PERCENTAGE OF JOBS FINISHED IN LESS THAN

TIME	15'	30'	1^h	2^h	4^h	8^h	1^D	2^D	3^D	6^D
% year 1975	29.3	47.0	65.0	78.7	84.9	87.2	96.8	97.4	98.7	100
% year 1976	50.2	68.8	83.0	93.8	97.9	98.9	99.3	99.4	99.5	100

Utilization of the computer center by the objectives and appropriation accounts for the month of August

		IBM 370/165 equivalent time in hours
120	General Infrastructure	54.0793
130	Scientific and Technical Support	2,7576
143	ESSOR Reactor	0.3928
145	Medium Activity Laboratory	0.0445
146	Central Bureau for Nuclear Measurements (CBNM)	–
191	Technical Support to Commission Activities	2.2716
193	Technical Support to Power Stations	1.8371
211	Waste Disposal	1.0344
213	Materials Science and Basic Research on Materials	0.7655
214	Hydrogen	1.2879
221	Reactor Safety	61.2168
222	Applied Informatics	27.7277
223	Information Analysis Services	36.4115
230	European Informatics Network	1.6020
251	Standards and Reference Materials	2.0435
252	Protection of the Environment	23.9332
253	Remote Sensing of Earth's Resources	1.3113
254	New Technologies	–
412	Fissile Materials Control	4.6997
	TOTAL	223.4164
190	Services to external Users	11.2653
	TOTAL	234.6817

Statistics of computing installation utilization

Report of computing installation exploitation for the month of **September**

	YEAR 1976	YEAR 1975
Number of working days	22 d	22 d
Work hours from 8.00 to 24.00for	16.00 h	9.25 h
Duration of scheduled maintenance	21.08 h	22.83 h
Duration of unexpected maintenance	6.72 h	4.08 h
Total maintenance time	27.80 h	26.91 h
Total exploitation time	324.20 h	177.83 h
CPU time in problem mode	114.12 h	61.41 h
Teleprocessing:		
CPU time	2.00 h	0.80 h
I/O number	333000	574000
Equivalent time	4.33 h	4.81 h
Elapsed time	143.00 h	93.00 h
Batch processing:		
Number of jobs	10,460	7,867
Number of cards read	3,368,000	2,628,000
Number of cards punched	212,000	188,000
Number of lines printed	27,838,000	20,609,000
Number of pages printed	634,000	467,000

BATCH PROCESSING DISTRIBUTION BY CLASS

	A	1	2	3	4	5	D	TOTAL
Number of jobs	1519	3652	1374	1981	640	159	414	9739
Elapsed time (hrs)	28	96	94	163	103	29	47	560
CPU time (hrs)	1,0	8.1	15.6	35.2	35.9	9.2	6.4	111.4
Equivalent time (hrs)	11.9	42.7	38.6	74.6	53.9	16.9	26.5	265.1
Turn around time (hrs)	0.2	0.4	0.6	0.8	1.2	1.8	1.2	0.6

PERCENTAGE OF JOBS FINISHED IN LESS THAN

TIME	15′	30′	1^h	2^h	4^h	8^h	1^D	2^D	3^D	6^D
% year 1975	25.0	42.5	59.6	73.8	82.0	84.1	96.3	96.9	99.1	100
% year 1976	51.6	69.2	83.1	93.3	98.2	99.4	99.6	99.9	99.9	100

Utilization of the computer center by the objectives and appropriation accounts for the month of **September**

		IBM 370/165 equivalent time in hours
120	General Infrastructure	61.0438
130	Scientific and Technical Support	1.6751
143	ESSOR Reactor	7.6533
145	Medium Activity Laboratory	0.0188
146	Central Bureau for Nuclear Measurements (CBNM)	–
191	Technical Support to Commission Activities	1.8375
193	Technical Support to Power Stations	3.6610
211	Waste disposal	2.0172
213	Materials Science and Basic Research on Materials	0.7116
214	Hydrogen	2.0286
221	Reactor Safety	79.0611
222	Applied Informatics	31.1711
223	Information Analysis Services	21.3411
230	European Informatics Network	4.7086
251	Standards and Reference Materials	4.1770
252	Protection of the Environment	7.2638
253	Remote Sensing of Earth's Resources	3.5041
254	New Technologies	–
412	Fissile Materials Control	0.8441
	TOTAL	232.7178
190	Services to external Users	27.0782
	TOTAL	259.7960

EQUIVALENT TIME TABLE FOR ALL JOBS OF THE ADMINISTRATION – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	64	55	62	73	62	61	94	52	51	59	74	70
accumulation	64	119	181	254	316	377	471	523	574	633	707	777
Year 1976	84	82	101	77	57	64	73	54	61			
accumulation	84	166	267	344	401	465	538	592	653			

EQUIVALENT TIME TABLE FOR THE JOBS OF ALL THE OBJECTIVES – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	178	171	168	166	142	166	228	137	152	170	190	176
accumulation	178	349	517	683	825	991	1219	1356	1508	1678	1868	2044
Year 1976	206	237	270	241	229	248	249	223	233			
accumulation	206	443	713	954	1183	1431	1680	1903	2136			

EQUIVALENT TIME TABLE FOR THE JOBS OF THE EXTERNAL USERS – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	16	28	24	28	32	31	26	15	18	19	12	18
accumulation	16	44	68	96	128	159	185	200	218	237	249	267
Year 1976	18	19	28	16	25	32	14	11	27			
accumulation	18	37	65	81	106	138	152	163	190			

EQUIVALENT TIME TABLE FOR ALL JOBS OF ALL USERS – MONTHLY AND CUMULATIVE STATISTICS

	January	February	March	April	May	June	July	August	September	October	November	December
Year 1975	214	216	208	215	190	222	266	166	181	202	219	208
accumulation	214	430	638	853	1043	1265	1531	1697	1878	2080	2299	2507
Year 1976	233	271	313	280	277	281	260	245	273			
accumulation	233	504	817	1097	1374	1655	1915	2160	2433			

A Programming Language for you

A.A. Pollicini

Normally, not experienced people are troubled, when they approach Programming.
Why this ?
Perhaps because people usually communicate each other by means of natural expression media that are the living languages, while programming languages are rather unnatural ones. Indeed programming languages are not used to communicate with human individuals, rather with strange and icy machines: the computers ! Thus the myth of Computer arose.

It is time to explode this myth and regard computers as natural tools in our hands. But to do this we need natural programming languages.

A great merit of Dr. Kenneth E. Iverson, maybe the greatest merit, was to consider that scientists are quite familiar with mathematical language, i.e. the formalisation of a logical and rigorous way of thinking, and then he developed a generalized mathematical notation on which he based the design of the programming language APL.
He certainly thought that scientists could have communicate with computers, without trouble of programming disciplines as:

- declare type and precision of variables
- specify format rules to enter data or print results
- code instructions at fixed columns
- etc.

only following the layout of mathematical expressions, therefore he enforced himself to provide APL with all these facilities.
That is why in the environment of Ispra JRC, APL must be a very suitable tool !

The system APL 360 is implemented at the Ispra Computing Centre since four months; an APL course was held last June and a little APL community began to approach the language.
In order that a great many of Establishment personnel may take advantage of such a powerful tool, an outline of the features of APL and a sample of use are presented below.

Main Features of APL 360 System

APL 360 system is based on a conversational language, therefore it is available for use through a connected terminal which is the only external evidence of the computing installation the user must approach (no punched cards, no submission card, no Job Control Language, no listing inside a box, etc.).

The Workspace Concept

To discharge users from storage allocation, the system assigns to each connected terminal a fixed amount of storage locations, that is a workspace.
When a terminal is working, its workspace is active.
The active workspace contains all variables and instructions generated during the working session.
The active workspace can be saved for use during further sessions, for instance to get data from a table or to call a pre-defined function. When saved, a workspace is stored into a System Library, where it is identified by a name.
The APL 360 system, provides users with **private** and **public** libraries. The relationship user-workspace can be of two forms:

- make computations applying the language;
- perform control and management operations by means of system commands.

Outline of the APL Language

The laguage is based on an extended character set, which consists of the 26 capital letters of the alphabet, the 10 decimal digits and 52 special characters. In addition some symbols may be composed overstriking two special characters.
Text-lines typed from terminal keyboard are interpreted by the system according to the principle of "visual fidelity", therefore backspaces can freely be used, but overstrikes is only permitted for composed symbols.
The fundamental elements of the language are:

- constants
- names
- operators

which concur to form statements.

Assignment Statement

The basic statement is assignment statement, which follows the syntax:

< assignment statement > ::= < identifier >← < expression >

As any APL name, < identifier > may be formed of any sequence of alphabetic, underlined alphabetic and numeric characters that starts with an alphabetic and contains no blank. The length of the sequence is formally unlimited.

< expressions > may be built recursively of operands and operators which come together to form elementary expressions of two types:

- monadic expressions in prefixed notation
- dyadic expressions in interposed notation.

Operands may be constants, variables or expressions. Operators may be primitive functions or defined functions. Because of the large number of operators, no hierarchy exists among them. Thus, in absence of parentheses, functions are evaluated from rightmost to leftmost.

Function Definition

While primitive functions, quickly represented by special characters, are provided by the system, defined functions must be provided by the user, entering their definition from the terminal, following the general syntax:

```
< function definition >:: =< function leader >< body >
                             < end definition >
< function leader >:: =∇< function referencing >
< body >              :: =< statements >
< end definition > :: =∇
```

The < Function referencing > contains the function name and optionally explicit arguments and can involve explicit results.
Notice that defined functions may be niladic when they are interactive or operate on global variables.
Control of execution of the < statements > within the < body > of the function, normally is sequential, but can be altered by statements of the type:

```
        < branching statements >::= →< expression >
```

System Commands

Each command is recognized by the system for its first character, that is a right parenthesis which contrasts with formal correctness of expressions and invalidates the text-line as statement.
Commands are grouped into five classes depending on their purpose.
The system commands are presented in the below list, where:

- items in brackets are optional,
- ***key*** and ***lock*** are passwords for information security,
- ***wsname*** is either library number and workspace name, or workspace name alone, as required.

Terminal Control

)user account number [:key]	Sign on designated user and start a work session.
)***OFF*** [:lock]	End work session.
)***CONTINUE*** [:lock]	End work session and store active workspaces.

Workspace Control

)***CLEAR***	Activate a clear workspace.
)***SYMBOLS*** n	Changes number of permitted names in a workspace.
)***LOAD*** wsname [:key]	Activate a copy of a stored workspace.
)***COPY*** wsname [:key] name	Copy a global object from a stored workspace.
)COPY wsname [:key]	Copy all global objects from a stored workspace.
)***PCOPY*** wsname [:key] name	Copy a global object from a stored workspace, protecting active workspace.
)***PCOPY*** wsname [:key]	Copy all global objects from a stored workspace, protecting active workspace.
)***GROUP*** names	Gather objects into a group.
)***ERASE*** names	Erase global objects.
)***ORIGIN*** 0 or 1	Set index origin for array operations.
)***DIGITS*** 1 to 16	Set maximum for significant digits in output.
)***WIDTH*** 30 to 130	Set maximum for an output line.
)***WSID*** wsname	Change active workspace identification.

Library Control

)***SAVE***	Save a copy of active workspace using its current name.
)***SAVE*** wsname [:lock]	Save a copy of the active workspace assigning a new name.
)***DROP*** wsname	Erase a stored workspace.

Inquiry

)***FNS*** [letter]	List names of defined functions.
)***VARS*** [letter]	List names of global variables.
)***GRPS*** [letter]	List names of groups.
)***GRP*** name	List membership of designated group.
)***SI***	List halted functions (state indicator)

)*SIV*	List halted functions and associated local variables (augmented state indicator)
)*WSID*	Give identification of active workspace.
)*LIB* [number]	List names of workspaces in the specified library.
)*PORTS*	List ports in use and codes of connected users.
)*PORTS* code	List port numbers associated with designated user code.

Communications

)*MSGN* port [text]	Address text to designated port. No reply expected.
)*MSG* port [text]	Address text to designated port and lock sender's keyboard. Reply expected.
)*OPRN* [text]	Address text to recording terminal (APL Operator). No reply expected.
)*OPR* [text]	Address text to recording terminal (APL Operator), and lock sender's keyboard. Reply expected.

An Example

Literature is rich of didactic examples of increasing complexity, and public libraries contain APL courses to guide progressively the user in learning APL, so that interested people can easily acquire a basic knowledge. Nevertheless this presentation includes a simple example which is a very little attemp to show the development of an APL session and to interpret each user operation and the consequent action of the system.

The Problem

It is to compute the distance D between adjacent planes of an hexagonal crystal, as function of five parameters:

– A, C	constants for a given compound,
– H, K, L	integers which identify the set of planes under consideration,

following the relationship expressed by the formula:

$$\frac{1}{D^2} = \frac{4}{3}\left(\frac{H^2 + HK + K^2}{A^2}\right) + \frac{L^2}{C^2}$$

The problem will be solved by a function definition followed by a series of executions of the pre-defined function with actual values.
The conversation' user-system is represented by the left side (in APL printing characters) in which the leftmost lines are texts printed by the system and the others are the text-lines typed by the user. The right side (in current block letters) explains the operation that appears on the same line.

Session Sample

```
       )131790                                   Sign-on to open the session.
007)    10.58.03   09/14/1976   TRAVET     Terminal (port) identifi-
                                                 cation; hour, date and user
                                                 identification.
APL/360                                          System ready to operate.
                                    *** Begin    the definition phase.
    ∇ HEXAGONAL                                  Define niladic function named
                                                 HEXAGONAL.
[1]      'SPECIFY A AND C (IN  ANGSTROMS)'
                                                 Send a message to ask for data.
[2]     AC ← □                                   Assign typed input to arr,ay AC.
[3]      'SPECIFY H K L'                         Send a message to ask for data.
[4]      HKL←□                                   Assign typed data tọ array HKL.
[5]      →(0=ρρHKL)/0                            End execution if typed data is
                                                 a  scalar instead of an array.
[6]      D←÷(+/ 4  4  4  3  xHKL[1  2  2  3]xHKL[1  1  2  3]
                ÷3xAC[1  1  1  2]*2)*0.5         compute the distance  D.
[7]      'D  IS  ';D;' ANGSTROMS'                Print  the  result.
[8]      →4                                      Branch to statement [4]  to
                                                 process a new case.
     ∇                                           End  definition.
                                    *** Begin    the  execution  phase.
         GE02← 4.987     5.652                   User assigns the values of con-
                                                 stants A and C for germanium
                                                 dioxyde to an array.
         END ← 0                                 User  assigns a scalar to a mnemo-
                                                 nic name to be used to stop
                                                 execution.
         HEXAGONAL                               User calls the function for
                                                 execution.
```

```
SPECIFY A AND C (IN ANGSTROMS)   System asks for input data
□:                               to be assigned to A and C.
  GE02                           User supplies array GE02.
SPECIFY H K L                    System asks for input data to be
□:                               assigned to H,K and L.
  1 0 0                          User supplies a set of planes.
D IS 4.318868689 ANGSTROMS       System replies the result.
□:                               System asks for new input data.
  1 1 0                          User supplies a new set of planes.
D IS 2.4935 ANGSTROMS            System replies the result.
  . . . . . . . . . . . .   *** As many iterations as the user desires.
  . . . . . . . . . . . .
□:                               System asks again for input.
  END                            User causes the end of execution.
  )SAVE WORK                     User saves the workspace contain-
                                 ing the function under the name
                                 WORK.
11.08.41 09/14/76                System declares hour and date of
                                 saving.
  )OFF                           User closes the session.
```

Bibliography

For the language description the following IBM manuals:
- APL/360 OS User's Manual GH20-0906
- APL/360 OS General Information Manual GH20-0850

Source of example application:
- APL/360 PRIMER GH20-0689-IBM

The Newsletter is available at:

Mrs R. Porta
Program's Library
Bldg. 36 - Tel. 760

Des exemplaires du Bulletin sont disponibles chez:

Mme R. Porta
Bibliothèque des Programmes
Bât. 36 - Tel. 760

Les personnes intéressées et désireuses de recevoir régulièrement "Computing Centre Newsletter" sont priées de remplir le bulletin suivant et de l'envoyer à:

Mme R. Porta
Bibliothèque des Programmes
Bât. 36, Tel. 760

Nom . . . .

Adresse . . . .

. . . .

Tel. . . . .

The Persons interested in receiving regularly the "Computing Centre Newsletter" are requested to fill out the following form and to send it to:

Mrs R. Porta
Program Library
Building 36, Tel. 760

Name . . . .

Address . . . .

. . . .

Tel. . . . .