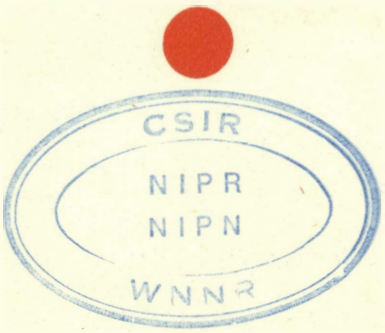
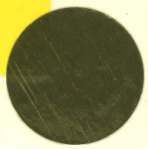
SPECIAL REPORT 5

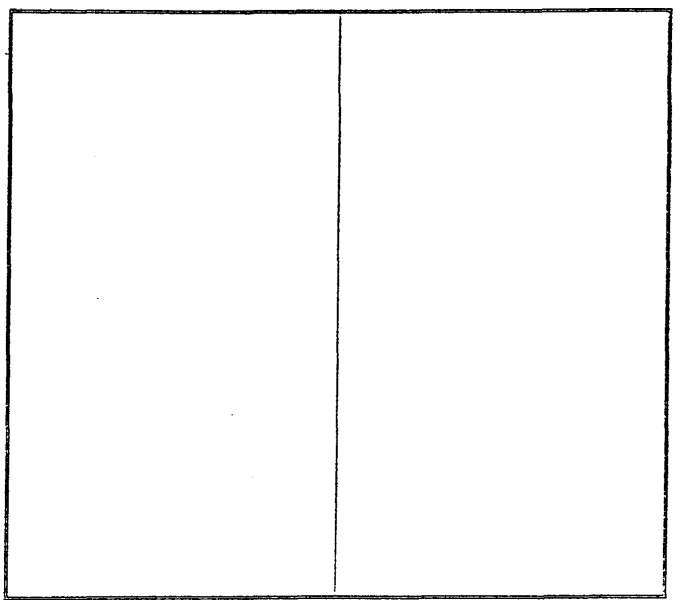
PERS 317 SUPPLEMENT TO PERS 305 :
MATHEMATICAL EXPRESSIONS
AND TABULATING



NATIONAL INSTITUTE FOR PERSONNEL RESEARCH
COUNCIL FOR SCIENTIFIC AND INDUSTRIAL RESEARCH

CSIR Special Report PERS 317 (pi - iv; 1 - 22)
UDC 519.684 : 655.25
Johannesburg, Republic of South Africa, March 1981





NATIONAL INSTITUTE FOR PERSONNEL RESEARCH
COUNCIL FOR SCIENTIFIC AND INDUSTRIAL RESEARCH

CSIR Special Report PERS 317 (pi - iv; 1 - 22)
UDC 519.684 : 655.25
Johannesburg, Republic of South Africa, March 1981

0 7988 1859 X CSIR Special Report PERS 317
Published by
National Institute for Personnel Research
Council for Scientific and Industrial Research
P O Box 32410
Braamfontein, Republic of South Africa
2017

March 1981

Printed in the Republic of South Africa by
National Institute for Personnel Research.

ACKNOWLEDGEMENTS

Director NIPR : Dr. G.K. Nelson

Mr. A. Giovannoni, AM International Ltd.;
for information and assistance and making the control tapes

Mr. S. Drijfhout, Graphic Arts, TSD, CSIR Pretoria;
for information and assistance.

Mrs. J. Gumbi; for typing this report.

Summary

This continues the work described in PERS 305 in two particular areas viz, tabulation and mathematical expressions and describes additional facilities provided in the program.

Opsomming

Hierdie veslag vervolg die werk wat in PERS 305 beskryf is,vernaamlik in die gebiede van tabulasie en wiskundige uitdrukkings. Ingesluit is 'n beskrywing van nuwe fasiliteite wat in die program aangebring is.

Foreword

The setting up of tabulations requires a different procedure to that described in PERS 305, namely the use of tabulation mode instead of hyphenless mode, which was not dealt with in the previous report. Secondly, additional facilities have been included in the program to simplify the handling of mathematical expressions. Although these operations are more difficult than straightforward text and more is required of the person setting up the Marker Table, in both cases, the underlying aim, namely that it should be possible for anyone with access to a digital computer terminal to carry out the bulk of the work of typesetting, has been met. As this can be done by a person less specialised than a typesetter operator, it should be accomplished at lower cost.

It has also been found that different disks are currently being used for 747 typesetter than those described in PERS 305. The layouts of these disks are shown in Appendix 3.

Further information is provided on certain of the commands mentioned in PERS 305.

Finally, although the methods described have been worked out for a particular machine, they clearly have wider application.

Table of Contents

I Purpose

The purpose of this report is to describe procedures for using the typesetting program for :-

(a) printing tables

(b) printing articles containing mathematical expressions

I Additional Notes on 747 Typesetter Commands

1) Unless preceded by Carriage Return, a new Point Size will be applied to whatever text is in the Typesetter's memory.

To insert characters involving a change of point size, enter the line without those characters. When the galley is available insert before (for superscripts or large symbols) or after (for subscripts) a Tape Lead to the base line for the inserts and Flush Left and Carriage Return. Then insert a line commencing with the new Point Size, new Line Length, followed by a Leading of zero, Flush Left and Carriage Return commands, then the spacing to position the character in its proper position in the text, the character itself, ending with a Tape Lead to the base line for the next line of text, Flush Left and Carriage Return. The next line must commence with a command setting the next Point Size and Line Length.

2) Add Lead, AL uses the last Leading specified whether Primary, Secondary or Tape. The paper is immediately advanced without a carrriage return.

3) Store Format, S,F, must follow the commands to be stored. 3 different formats may be stored in Hyphenless mode and 17 in Tabulation. To activate a stored format in Hyphenless mode give a Use Format, UF, command followed by a Carriage Return command. To activate it in Tabulation mode give UF within a tab column when the tabs are being set with NT commands. Note that if formats for the primary parameters and tabulation columns are all being stored, the tab format should be stored first followed by a Carriage Return command before the primary parameters are stored. This avoids inadvertent carry over of tab parameters.

4) Indenting. The Use Indent command must precede the first line to be indented.

5) Tabulation, NT, requires a control tape made up for operating in mode 1 (Tabulation) and the user is responsible for arranging the spacing of text in a line. After the printing parameters, a second initialising command string is required for setting the tab positions and there must be two more than the number of columns, the first to mark the start of the line, the last to mark the end of the table. Within each column, i.e. after the NT command for the column, there should be:- the column width in units appropriate to the point size being used, the font, and the position of the data inside the column i.e. left or right adjusted or centred, which must be locked by using a double Flush command. If no such parameters are supplied the last given will carry over to subsequent columns. The command string can be simplified by using a Use Format command in each column or using the program's repeat facility if the specifications for each column are the same. When entering data there must be an NT command for each column (including the dummies) followed by blank or data. (max.=16 cols.)

6) Horizontal and vertical lines can be drawn using the Leader and Rule commands. A Leader Character must be stored in the initial command string and a Use Leader command must be given at the start of a line (Hyphenless mode) or in each column (Tabulation mode) in which a horizontal line is required. Note that Leader characters are 9 units and unless tab columns are exactly multiples of 9 units in width, gaps will occur in the horizontal lines. A method of overcoming this is to adjust the overall width of the table to a multiple of 9 units and to specify a line length equal to the table width, an Indent left (IA) equal to the left hand margin and to give Use Leader without tabs.

Storing the Vertical Rule with a Store Rule command causes the rule to be printed on the left of the page, (Hyphenless mode) or column whenever the NT command for that column appears (Tabulation mode). The Vertical Rule character is made inoperative by storing a blank. If the Rule character requires a different font to that used for the contents of the column, the required font must be given before the NT command and the normal font restored immediately afterwards.

7) A number of commands must be preceded by a Carriage Return as their operation may otherwise affect preceding text. They also only come into effect when the current block has been processed.
The commands are :-
Point Size, Leading (Primary, Secondary or Tape), Line Length, Indents and Tabs.
It is advisable to have Flush commands (QL,QC,QR) immediately prior to the Carriage Return command to avoid conflict if there is line overflow.

8) The Indent commands operate differently in the Hyphenless and Tabulation modes.

In Hyphenless mode the operation is as described in Appendix 7 of PERS 305

In Tabulation mode, IA,IB,IC mean respectively, Indent on the left, Indent on the right and Indent both sides, and the indents apply as soon as the command is received so that the Use Indent commands fall away.

9) There is no underline character on the disk and the EBCDIC character Underline (6D) is replaced by a long hyphen.

10) Line End operates only in Hyphenless Mode and it does not implement changes in Point Size or Leading.

11) If changes of Secondary Leading or Tape Leading follow too closely, the last may replace the preceding specification even if separated by Carriage Return. This occurs because the machine reads a second line while still processing the first and commands are processed more rapidly than text. The remedy is to introduce a delay such as a string of 20 ignore characters before the second command.

12) If long strings of spacing commands are required it is better to break them into shorter segments separated by blanks so that the machine can still accommodate the line even if, by mistake, it has been made too long.

13) Line Length remains a constant number of points, whatever the Point Size so that the physical length changes with Point Size. Thus a Line Length of 0580 points occupies 100mm in Point Size 090 but only 67mm in Point Size 060. If it is necessary to use the same physical length in a smaller Point Size, the Line Length must be increased.

    Maximum physical Line Length is 180mm i.e. 1035 points at Point Size 090.

14) The Typesetter flashes the Rule character before it processes the rest of the information in a column. If a change of rule character is required it must be given in the same tabulation line.

    Also, information specified for a column in a previous tabulation line takes precedence over information specified for preceding columns in the current line.

15) Tabulation lines (lines in which the NT command is used) are not governed by the overall Line Length, but only by the width specified for individual columns. Thus a tabulation line may exceed the overall Line Length.

II Additional Facilities in Version 5 of the Program

The program is designed to shield the user from most of the pecularities of the Typesetter's operation. For example, the program supplies the required font and restores the current font when accessing a disk character that requires a font change. It also eliminates the blanks inserted by the RJE system to fill out a line after a Carriage Return

1) LT command

All blanks from this command to the next non-blank character or the end of the input line are eliminated. The command is ignored by the Typesetter.

The command has been introduced to simplify the correction of text. As the RJE system pads short lines with blanks, corrections which cause under or overflow produce additional blanks which are read as valid characters by the Typesetter. In the case of underflow of only a few characters, Ignore symbols can be used to fill out the line, but when there is an overflow or a large underflow, the string of blanks filling out the line can be eliminated by the LT command.

2) Imbedded Text Facility

The user has the option of inserting a 4th symbol, e.g. &, in the second line of the input file as a delimiter for text imbedded in command strings. The program treats all characters between & and the next command or repeat delimiter as text. Imbedded & or blanks are treated as text

An example of imbedded text arises with f(x) in which f and x are in italics but not the brackets. This can be handled by the following command string :-

ØFI&fØFM&(ØFI&xØFM&)

3) The Repeat Facility

The user has the option of inserting a 5th symbol, e.g. ?, in the second line of the file as a delimiter for either repeated @ codes in text or a repeated string within a command string. In the case of the former, ? is followed by 2 digits giving the number of repetitions (including the first occurence)

e.g. if L represents & ØØ, inserting 9 spaces in text is produced by ?03@L

For a repeated string within a command string, ? is followed by 2 digits giving the number of repetitions and another 2 digits giving the length of the string to be repeated, e.g. ?0303ØMS, will also produce 9 spaces. (width of MS = 18 points = 3 spaces)

Note : a repeated @ code may refer to a command string containing a repeated string, but other forms of nesting are not permitted

e.g. ?03?06@L or ?3007?0502&- are not valid.

4) The program will handle both Leader or Rule characters which are available on computer terminal keyboards and those which are only found on the Typesetter's disk and are retrieved using a command delimiter and a mnemonic, e.g. ØSR| or ØSRØRT

5) Lengths are no longer used for command strings. The end of a command string is signalled by two command delimiters. The @ code character and the command string must be separated by two spaces, e.g. S ØPS060ØØ

The space after ØØ may be used for comments

The length of a converted string must not exceed a tape record.

6. Default symbols for delimiters and marker are provided in the program. They are

Command delimiter	∅
@ code marker	@
Ingnore character	¢
Imbedded text delimiter	&
Repeat delimiter	✱
Invalid @ code in the text	□
Invalid text character	△

These may be overridden by supplying alternatives in the second line of the input file using the format :-
command,space,@ marker,space,ignore,space,imbedded text,space, repeat,space,invalid @ code, space,invalid text marker.

If a blank is placed in the position for a particular delimiter or marker, the default will be used. If a zero is placed there that facility will be made inoperative.

The only delimiter or markers that may <u>not</u> be used as text are the @ code marker, the ignore character and the repeat delimiter

7. The first line of the input file is used for a dual purpose

a) The first 4 characters are compared with a corresponding set of four characters stored in the program. Each time a different version of the program is created e.g. with a different set of mnemonics for a different disk, a different set of characters will be be stored. This will assist avoiding mistakes by using the wrong version of the program.

(In practice, disks 489,490,737 and 746 are so similar that a single version of the program suffices.)

b) The entire line is printed in large bold letters, 20 characters to a line, well ahead of the text. This can be used for identifying the galley and for reminding the Teleprinter operator of the disk and mode to be used

8. The format required for the input file, is :-

Line	1	check word and galley heading
	2	delimiters and markers
	3	N=number of @ codes (2digits)
	4 to N + 4	@ Code, 2 spaces, command string (up to 76 characters)
	N + 5	session identifier (not printed on the galley)
	N + 6	text
	etc.	

9. A summary of the instructions for using the program, information on its structure and the changes to be made if it requires enlarging or modifying for a different disk or machine are provided in the comments ahead of the program.

10. Any symbol on the keyboard may be used as an @ code, even @ itself, and all, even @ if it is followed by a blank, may be used in the text, except those mentioned at the end of (6).

11. In addition to the Control Tape, the Typesetter requires a Program Tape. This tape must be the Program Tape for the Keyboard with all facilities. A third tape, specific to the disk to be used is also required. The last two tapes are kept at Graphic Arts and are of no concern to the user, except that the failure of a particular function may indicate that a wrong Program Tape was used.

11) Additional Error Returns

(A complete list of Error Returns is shown in Appendix 9)

Error Number	Reason	Program Line Number of Test	Program Action	User Response
(16)	Symbol chosen for marking errors in text is not available on the Typesetter disk	1030	E	Choose another symbol
(17)	Non-EBCDIC characters in Imbedded text	805	E	Check the string
(18)	Command string overflows input buffer	many	E	Correct the length of command string
(19)	Nested repetition within a command string	777	E	Correct the command string
(20)	Length of command string (possibly containing a repeated command) exceeds a tape record	932	E	Break up the command string
(21)	A repetition used in text is nested	363	C	Correct the text
(22)	Non-numeric repetition number	372/781/786	E	In a command string: correct the command string
			C	In the text : Correct the text

Error Number	Reason	Program Line Number of Test	Program Action	User Response
(23)	An invalid mnemonic (i.e. not two upper case alphabetic characters) has been used in HXTAB or for special commands or symbols in SR. TABLES	644	S	Check the mnemonics
(24)	Errors in preliminary processing	341	S	Correct the errors
(25)	Imbedded text not available on disk	805	E	Study disk layout and HXTAB and OPTAB entries
(26)	Command not provided for in computed go to for galley heading printing parameters	997	E	Correct the command or modify the computed go to
(27)	Repeated command causes overlfow of array for converted strings	725	E	Enlarge the LCODE array
(28)	Size of Input Buffer (CH) is not a multiple of the input record (ITBREC)	251	E	Correct INCHAR and the dimension of CH
(29)	Size of Output Buffer (KCH) is not a multiple of the output record (IMBREC)	255	E	Correct KOUCHS and the dimension of KCH

III Procedure for handling Tabulations

A control tape for operating the 747 Typesetter in tabulation mode i.e. mode 1, is required - this differs from the control tape used for Hyphenless mode. For this reason tables must be produced in a separate run from normal text.

In addition to the usual primary printing parameters i.e. Point Size, Line Length, Primary Leading etc., an initial command string is required for setting the tabs for the columns. This provides information on column sizes, the positioning of text within the column and the font. Horizontal and vertical lines can be drawn using Leader and Rule commands. A point to note is that Primary Leading should be the same as Point Size, otherwise the vertical lines will either be broken or project beyond the horizontal lines. Also, table width should be a multiple of 9 otherwise gaps will appear in the horizontal lines.

Step 1

Set up the initial command string with Point Size, Line Length, Primary and Secondary Leading, Type Font, Clear Rule, set the Leader Character, LY, and end with Flush Left, Carriage Return.
e.g. S ØPS090ØPL090ØSL070ØLL0450ØFMØSR ØLDØLYØFLØCRØØ/
In this case Line Length is required only for drawing horizontal lines and extends only to the right hand margin of the table.

Step 2

Decide on column width and set up a command string for placing the tabs in which each column is specified by an NT command, Line Length, Type Font, Rule, and a QuadLock (two flush commands for placing the contents of a column left or right adjusted or centrally).

If necessary the command may be spread over two @ codes but the multiplying facility or stored formats may be used to avoid this. Specifications may be given for the first column of the table only, in which case they will apply to all subsequent columns, or the multiplying facility may be used.

e.g. T ?0519ØNTØLL0090ØFMØFCØFCØNTØCRØØ

This string is for a table of 4 columns with an initial dummy column for a left hand margin and a final dummy column to produce the right hand vertical line. The width of each column is 90 points and text is centered. The string must terminate with a Carriage Return.

Step 3

Next set up a command string for drawing the first horizontal line. This does not use tabs but requires an Indent Left for the left hand margin, Use Leader for drawing the line and a Use Secondary to drop the Base Line 7 points so that the tops of the Vertical Rules will just touch the horizontal line. This is followed by the commands for the vertical lines dropping from the horizontal line.

e.g. F ØIA090ØULØUSØFLØCRØNTØSR ?0507ØNTØSRIØCRØØ

Step 4

Next construct the command for the middle horizontal lines. In this case the command string for the descending Vertical Rules is followed by a Tape Lead to drop the Base Line for the horizontal line 3 points so that it just touches the bottom of the Rules. The rest of the string is similar to that in step 3.

e.g. M ?0603ØNTØTL030ØCRØIA090ØULØUSØFLØCR?0603ØNTØCRØØ

Step 5

Construct the command string for the last horizontal line of the table. This is similar to the first part of Step 4.

e.g. L ?0603ØNTØTL030ØCRØIA090ØULØUSØFLØCRØØ

Step 6

Set up the tabulation commands for entering the data.

e.g. t ØNTØØ,

the command for producing the right hand vertical line

e.g. e ØNTØCRØØ

and the final command for halting the Typesetter at the end of input or, in a long series of tables, to enable the operator to put a new roll of photographic paper in the Typesetter.

e.g. H ØHAØØ

Step 7

Putting this together to print the following table

	Q15	Q17	Q36
Type 6	620	12	8
Type 7	2	2	105

Marker Table

```
CSIR FOR NIPR. DISK: 490 SCHOOLBOOK. MODE: TABULATION
Ø @ : & ?
08
S ØPS090ØPL090ØSL070ØLL0450ØFMØSR ØLDØLYØFLØCRØØ
T ?0519ØNTØLL0090ØFMØFCØFCØNTØCRØØ
F ØIA0090ØULØUSØFLØCRØNTØSR ?0507ØNTØSR|ØCRØØ
M ?0603ØNTØTL030ØCRØIA090ØULØUSØFLØCR?0603ØNTØCRØØ
L ?0603ØNTØTL030ØCRØIA090ØULØUSØFLØCRØØ
t ØNTØØ
e ØNTØCRØØ
H ØHAØØ
TABULATION 1
@S@T
@F
?03@tQ15@tQ27@tQ36@e
@M
?02@tTYPE 6 @t620@t12@t8@e
@M
?02@tTYPE 7 @t2@t2@t105@e
@L
@H
```

There are a large number of ways the Marker Table can be drawn up. Some involve longer command strings and fewer @ codes in the text and vice versa, and they differ according to whether few or many tables are required and how similar they are.

Alternative Method

Tables can be entered as text in hyphenless mode using spacing commands for positioning the vertical lines and data. In this case care must be taken to see that the total number of units required for spaces and characters is less than the specified line length. Also, every line of the table must end with a Flush command, e.g. QL or QR and the Carriage Return command. Adjustments of Secondary Leading will still be necessary for joining lines. The Store Rule command is not available so the user must use the Vertical Rule Character from the keyboard, positioning it himself.

Remark

The Marker Table appears very formidable but the pattern is the same for all tables, only the number of columns and column widths change. The instructions for a typist to build up a table with @ codes are simple :-

"Type @F for the beginning of a table.
For each line of data in the table, type @t for the edge of the page, and an @t for each column followed by its data, if any.
End the line with @e.
Between lines of data type @M.
After the last line of data in the table type @L."

Headings and footnotes can be entered with the main body of the text in Hyphenless mode.

IV Procedure for handling Mathematical Expressions

The difficult aspects of typesetting mathematical text are :-

1) the changes in Point Size (size of type) e.g. small subscripts and superscripts or large brackets, square root and integral symbols

2) the frequent occurrence in text of italicised groups of characters such as f(x), or combinations involving Greek characters

3) the amount of spacing required in mathematical expressions and the necessity for preventing the typesetter from modifying this spacing

4) Typesetter peculiarities : Point Size only changes when the current line has been processed and subsequent Leadings will replace prior if they follow too closely.

The following method has been found effective for overcoming these difficulties

Step 1

Scan the text for frequently recurring combinations such as f(x) and prepare command strings and @ codes for them e.g. for f(x) use @f to stand for :-

ØFI&fØFM&(ØFI&xØFM&)ØØ or ØFI&f(x)ØFMØØ

Where there are super- or subscripts e.g. $f_n(x)$ or a^m, replace them by spaces. These will be filled on a second run

Step 2

Enter text in the usual way leaving out all smaller or larger sized characters such as subscripts or integral signs. Where there are equations or expressions standing alone, omit these entirely.

Step 3

As a change in Point Size must be preceded by a Carriage Return, dummy lines have to be created for inserting super- or subscripts or integral and summation signs or braces.

Superscripts, Subscripts and Large Symbols in Text

The galley from the first run will show where the typesetter has ended lines and where command strings for the characters still to be inserted must be placed. These will generally not be at the end of input lines and the latter will have to be broken and the second half moved to another line.

The command strings consist of a shift to the base line for the additional characters, a Carriage Return screening the preceding text from the change in Point Size, new Point Size, Line Length in units of the new Point Size, Use Secondary Leading (set to zero), and a Carriage Return to bring the new Point Size into operation. Leadings should be preceded by delays of about 20 Ignore characters to prevent overwriting. Command strings for subscripts follow and those for superscripts and large symbols precede the line they affect.

The following are examples of the command strings, assuming text PS = 9, PL = 12, SL = 0 and PS for sub- and superscripts = 6 and for large symbols = 14. Base line for superscripts 3 points above text and for subscripts 6 points lower than ordinary text.

The following @ codes suffice for all subscripts and superscripts.

@1 ?2003ØIGØTL060ØFLØCRØPS060ØLL1069ØFI?2003ØIGØUSØFLØCRØØ
@2 ?2003ØIGØTL060ØFLØCRØPS090ØLL0713ØFM?2003ØIGØUSØFLØCRØØ
@3 ?2003ØIGØTL090ØFLØCRØP060ØLL1069ØFI?2003ØIGØUSØFLØCRØØ
@4 ?2003ØIGØTL030ØFLØCRØP090ØLL0713ØFM?2003ØIGØUSØFLØCRØØ

The typing instruction are correspondingly simple. Where superscripts are required type "@ 3 spacing and superscripts @4" before the line affected and for subscripts type "@1 spacing and subscripts @2" after the line. A method for determining the spacing commands is described in Appendix 7.

If there are large symbols with a base line 6 points above the text, another command string will be required e.g.

@5 ?2003ØIGØTL060ØFLØCRØPS140ØLL0535?2003ØIGØUSØFLØCRØØ

If they have superscripts then another command string will be required e.g.

@6 ?2003ØIGØTL030ØFLØCRØPS060ØLL0535?2003ØIGØUSØFLØCRØØ

The instructions for the typists would be "type @5 spacing and @ code for the large symbol @6 spacing and the superscript @4".

Step 4

Mathematical Equations set apart from normal text

These each occupy several lines of text and because of their complexity are best handled in a separate run.

As Point Size cannot be changed within a line it is necessary to handle each collinear set of characters of a given Point Size in a separate line and to move from one base line to the next by means of a Tape Lead.

In Appendix 7 it is shown that an integral of a rational expression of the form :-

$$a_m^N(y) = \int_c \frac{f_{N,m}^{N}(y)^m}{g_{N,m}^{m}(z)^N}\, dz$$

requires only 5 base lines for its elements.

If Point Sizes of 18,9 and 6 are used for integral signs, text and subscripts respectively, then apart from the first shift from the base-line for the integral sign to that for the numerator text which depends on the positioning of the integral sign, the remaining base lines, are separated by Leadings of 6. This generates a remarkably uniform set of command strings and an even simpler system of @ codes for the typist.

The first command string required is for a switch into Point Size 18. A zero Leading can be used as the positioning of the base line can be left to the Spacing command string, e.g.

@5 ?2003ØIGØTL000ØFLØCRØPS180ØLL0356ØUSØFLØCRØØ

Command strings such as @5 are placed at the end of the line preceding the line they affect. The Carriage Return causes the blanks to the end of the line to be skipped .

Next there must be a change to Point Size 9 and a change to medium font for the text of the numerator and a Leading to position the rest of the expression in relation to the integral sign. The actual value used is a matter of taste but a Leading of 2 appears satisfactory, e.g.
@6 ?2003ØIGØTL020ØFLØCRØPS090ØLL0713ØFMØUSØFLØCRØØ

Superscripts are in Point Size 6 but use the same base line as text and are in italics e.g.
@3 ?2003ØIGØTL000ØFLØCRØPS060ØLL1069ØFIØUSØFLØCRØØ

Numerator subscripts and superscripts for fraction line text require a Leading of 6, a Point Size of 6 and italics, e.g.
@4 ?2003ØIGØLT060ØFLØCRØPS060ØLL1069ØFIØUSØFLØCRØØ

Fraction line text uses the same base line as numerator subscripts but requires Point Size, 9, e.g.
@1 ?2003ØIGØTL000ØFLØCRØPS090ØLL0713ØFMØUSØFLØCRØØ

Fraction line subscripts, denominator text and superscripts all use the same base line. Thus an initial Leading of 6 and subsequent Leadings of zero are required and can be obtained using @4, @1 and @3.

If there are no sub - or superscripts, a Leading of 6 is required for shifting to the denominator from the fraction line, e.g.
@2 ?2003ØIGØTL060ØFLØCRØPS090ØLL0713ØFMØUSØFLØCR.

These six command strings suffice for all the changes in base line and Point Size required for most mathematical expressions.

Step 5

The next step is to work out the spacing for the elements of the expression. This can be done as described in Appendix 7 where the spacing for the expression (1.2) has been worked out as an example.

Conclusion

The examples in Appendices 5,6 and 8 and in the previous report, PERS 305, show that it is perfectly feasible for anyone equipped with an ordinary computer terminal to use the program described in this report to set up text, whether it be straightforward text, tabulation or mathematical text, for typesetting on an AM International 747 Typesetter. No doubt similar programs could be written for other Typesetters which use computer compatible magnetic tapes, disks or floppy disks.

This is not to say that this method is an improvement on specialised typesetting keyboards. These are so quick, easy to use and efficient that they are obviously the preferred method for typesetting, but they are also scarce and expensive, and it would appear to be a more cost-effective approach to use the specialists and specialised equipment for the intricate and difficult cases of typesetting, and to leave the setting up of large volumes of ordinary text, which is a routine operation, to clients equipped with ordinary computer terminals. Specialists could make it even easier for clients by setting up the Marker Table and preparing the typing instructions.

Further simplification could be achieved by extending the computer program so that, when given that a subscript is to be inserted into position indicated by the number of typewriter spaces from the edge of the page, it will handle all the necessary commands. This would reduce typing instructions to "@S046n" where @S is the command invoking the subscript routine.

APPENDICES

Addition to Table of Mnemonics in Appendix 7 of PERS 305

Mnemonic	Hex Code	Name	Function
LT	C0	Line Terminate	Skips blanks until the next non-blank character, or the end of the input record.
RB	AE	Bottom Vertical Rule	Generates a vertical rule for the lower $\frac{2}{3}$ of current Point Size
RT	AF	Top Vertical Rule	Generates a vertical rule for the top $\frac{2}{3}$ of current Point Size
IS	has been changed to DA		
NS	" " "	" DN	
MS	" " "	" DM	
DE	" " "	" ER	
UA	(See Pers 305)		Only for Hyphenless Mode
UB	"		"
UC	"		"
AI	"		Tabulation Mode : Indent left
IB	"		" " right
IC	"		" " both
DB	1F	2 space	Insert a space of 2 points
DC	2C	3 "	" " " " 3 "
DD	9A	4 "	" " " " 4 "
DE	2D	5 "	" " " " 5 "
DF	9B	6 "	" " " " 6 "
DG	9C	7 "	" " " " 7 "
DH	9D	8 "	" " " " 8 "
DI	9E	10 "	" " " " 10 "
DJ	9F	11 "	" " " " 11 "
DK	2E	12 "	" " " " 12 "
DL	2F	15 "	" " " " 15 "

Changes in Tables of Appendix 8 of PERS 305

P. (ii) Hexadecimal Codes for Typesetting Computer Functions

Cell		
	AE	from blank to Bottom Vertical Rule
	AF	---------- Top Vertical Rule
	80	---------- Font Store Flag
	90	---------- Restore Font Flog
	A0	---------- Front Flag
	C0	---------- Line Terminate
	1F	---------- 2 spaces
	2C	---------- 3 "
	9A	---------- 4 "
	2D	---------- 5 "
	9B	---------- 6 "
	9C	---------- 7 "
	9D	---------- 8 "
	9E	----------10 "
	9F	----------11 "
	2E	----------12 "
	2F	----------15 "

P. (iii) Hexadecimal Codes for Mnemonics

Cell		
	AE	from blank to mnemonic RB[2]
	AF	---------- RT[3]
	80	---------- FS
	90	---------- RF
	A0	---------- FF
	C0	---------- LT
	1F	---------- DB
	2C	---------- DC
	9A	---------- DD
	2D	---------- DE
	9B	---------- DF
	9C	---------- DG
	9D	---------- DH
	9E	---------- DI
	9F	---------- DJ

Hexadecimal Codes for Mnemonics (cont.)

2E ---------- DK

2F ---------- DL

0D from IS to DA

0E " NS " DN

0F " MS " DM

29 " DE " ER

6E should have a font code = 1

P. (iv) Operation Table

Cell	05	from	19	to 1
	09	"	1	" 7
	0B	"	1	" 7
	0C	"	19	" 1
	1B	"	19	" 18
	1F	"	19	" 1
	2C	"	19	" 1
	2D	"	19	" 1
	2E	"	19	" 1
	2F	"	19	" 1
	9A	"	19	" 1
	9B	"	19	" 1
	9C	"	19	" 1
	9D	"	19	" 1
	9E	"	19	" 1
	9F	"	19	" 1
	AE	"	19	" 3
	AF	"	19	" 4
	C0	"	19	" 7
	5A	"	1	" 15

Characters Available on 747 Disks at Graphic Arts CSIR

1) South African Mathematical Format : Disk PT490 School book

a) all three fonts : medium, italic, bold.

1. alphabet : lower case a to z, upper case A to Z
2. linguistic characters : ń ê ë ï
3. digits : 0 to 9
4. punctuation ; : ‘ ’ ? ! and both lower and upper case , . -
5. special characters : ( ) [] % / &

b) only one font

1. accents : lower case : ´ ` ¨ ^ ˘ , upper case ´ ` ¨ ^ ˘
2. subscripts : digits 0 to 9 ( )
3. superscripts : digits 0 to 9 ( ) + -
4. mathematical symbols : + − × ÷ = ≡ > < ± . * • ∫ √ —
5. measures: □ Δ ℓ £ °
6. Greek alphabet : α β γ δ Δ ι λ μ ν ω Ω π σ Σ θ
7. special characters : † *
8. special printing characters : vertical rules | | leaders . .. _ —

The following disks have similar character sets with minor differences.

Instead of the vertical rules | | on PT490

PT 489 Times Roman has ! in all three fonts

PT 746 Times Roman has ρ → *

PT 737 Megaron has A° → ≤

PT 746 also has ~. A° instead of £ √ ≡

The actual character positions on PT490 are as shown in Appendix 9, p.(i) of PERS 305.

2) South African Library Format : Disk PT488 Times Roman

a) all three fonts : medium, italic, bold,

1. alphabet : lower case a to z, upper case A to Z
2. linguistic characters : lower case æ œ, upper case Æ Œ
3. digits : 0 to 9
4. punctuation ; : ? ! and both lower and upper case, . -
5. special characters: ( ) [] % / — 1

b) medium and bold fonts,

1. upper lower case : ´ ` ¨ ^ ˘ ~ ° ///
2. other special characters : ¸ · ¯ - ˇ ^

c) medium font only

1. arithmetic characters : + ± = × ¼ ½ ¾ ²³
2. other special characters: α β & @ © £ ∂ * # ≠ ‡ † → * || | !

The positions of the characters on the disk are shown on P. (ii) Appendix 9 in PERS 305

3) South African Format - Roman Faces : Disk PT482 Megaron.

a) all three fonts : medium, italic and bold.

1. alphabet : lower case a to z, upper case A to Z, a g
2. linguistic characters : ń ê ë ï
3. digits : 0 to 9
4. punctuation: ; : ‘ ’ ? ! and both lower and upper case , . -
5. special characters : ( ) / % &

b) medium and bold

1. — ℓ

c) medium font only

1. accents : lower and upper case ` ´ ¨ ^ ˇ
2. subscripts : 0 to 9 ( )
3. superscripts : 0 to 9 + - / *
4. mathematical : + × ÷ = <> ± $\frac{1}{8}$ $\frac{1}{4}$ $\frac{3}{8}$ $\frac{1}{2}$ $\frac{5}{8}$ $\frac{3}{4}$ $\frac{7}{8}$ {} ± = - μ .
5. measures : □ △ £ ⌀ °
6. special characters : ◇ ◻ † ® © @ - - ™
7. special printing characters : leaders . .. _ — vertical rule |

Details of the Contents of the Disks at the CSIR

PT489 Times Roman

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ^ ' ¨ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z . ' ? ï & ´ (/ -)¨ / ê ! - : , . '

super shift
△ ' × ω ι θ ' δ ÷ ° > = £ ≡ , σ Σ

unshift
*a b c d e f g h i j k l m n o p q r s t u v w x y z * ` β [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '*

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ¯ _ ' ? ï & ` (μ —)^ / ê ! - : , . '

super shift
λ α † Ω ∫ γ ' Δ ° π < ± \ ' , ν

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ⁶ ² ³ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z + _ ' ? ï & • (· —) □ / ê ! - : , . '

super shift
¹ ¹ ² ³ ⁴ ⁵ ⁴ ⁷ – ⁵ ⁶ ⁹ ⁷ ⁸ ⁹ ⁰ ⁰

PT490 School-book

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ^ ' ¨ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z | ? ï & ´ (/ -)¨ / ê ! - : , . '

super shift
△ ' × ω ι θ ' δ ÷ ° > = £ ≡ , σ Σ

unshift
*a b c d e f g h i j k l m n o p q r s t u v w x y z * ` β [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '*

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ¯ _ | ? ï & ` (μ —)^ / ê ! - : , . '

super shift
λ α † Ω ∫ γ ' Δ ° π < ± \ ' , ν

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ⁶ ² ³ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z + _ ' ? ï & • (· —) □ / ê ! - : , . '

super shift
¹ ¹ ² ³ ⁴ ⁵ ⁴ ⁷ – ⁵ ⁶ ⁹ ⁷ ⁸ ⁹ ⁰ ⁰

Details of the Contents of the Disks at the CSIR

PT737 Megaron

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ˆ ¨ [ ] 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Ä ? ï & ˜ (/ —) ¨ / ê ! - : , .'
super shift
△ ' × ω ι θ ' δ – ° > = £ ≡ ¸ σ Σ

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ˆ ¨ β [ ] 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ˜ _ → ? ï & ` (μ —) ˆ / ê ! - : , .'
super shift
λ α † Ω ∫ γ ' Δ ® π · + \ ' ¸ ν

PT482 Megaron (no width tape)

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ⁶ ² ³ [ ] 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z + _ ≦ ? ï & • (· —) □ / ê ! - : , .'
super shift
₁ ¹ ₂ ₃ ⁴ ⁵ ₄ ⁷ – ₅ ₆ ⁹ ₇ ₈ ₉ ⁰ ₀

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ˆ ¨ α g 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ¨ | ? ï & ˜ (/ —) ¨ / ê ! - : , .'
super shift
△ ½ × ® ¾ ˜ ¦ _ ÷ ° > = £ © ¸ ½ ™

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ˆ ¨ @ α g 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ˜ _ ± ? ï & ` (μ -) ˆ ¨ ê ! - : , .'
super shift
▽ ⅛ † ◻ ⅜ ⅞ | $ ⁸ < ± ◇ ' ¸ ⅛

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ⁶ ² ³ α g 1 2 3 4 5 6 7 8 9 0 % ë ' n - ; , .'

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z + _ = ? ï & • (/ —) □ / ê ! - : , .'
super shift
₁ ¹ ₂ ₃ ⁴ ⁵ ₄ ⁷ – ₅ ₆ ⁹ ₇ ₈ ₉ ⁰ ₀

Details of the Contents of the Disks at the CSIR

PT488 Times Roman

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ` ˛ ˆ œ æ 1 2 3 4 5 6 7 8 9 0 % ˜ - : , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Œ Æ ˚ ? ı & [(¯ —)] / ¨ ! - : , . '
super shift
· ˙ ~ • ` ˆ ˍ ˆ ˇ / ¸ ˜ ˇ / ˘ ¨ .

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ‡ © | œ æ 1 2 3 4 5 6 7 8 9 0 % ¹ ² - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Œ Æ ¡ ? ı [(} —)] / ¦ ' - : ,
super shift
*β ▮ α ± ! $ < = £ + > * † # → @ ‡*

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ` ˛ ˆ œ æ 1 2 3 4 5 6 7 8 9 0 % ˜ - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z Œ Æ ˚ ? ı [(¯ —)] / ¨ ! - : , . '
super shift
· ˙ ~ • ` ˆ ˍ ˆ ˇ / ¸ ˜ ˇ / ˘ ¨ .

PT746 Times Roman

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ˆ ˊ ¨ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ρ ? ï & ´ (/ —) ¨ ê ! - : , . '
super shift
△ ' × ω ι θ ' δ ÷ ° > = ~ Å ˌ σ Σ

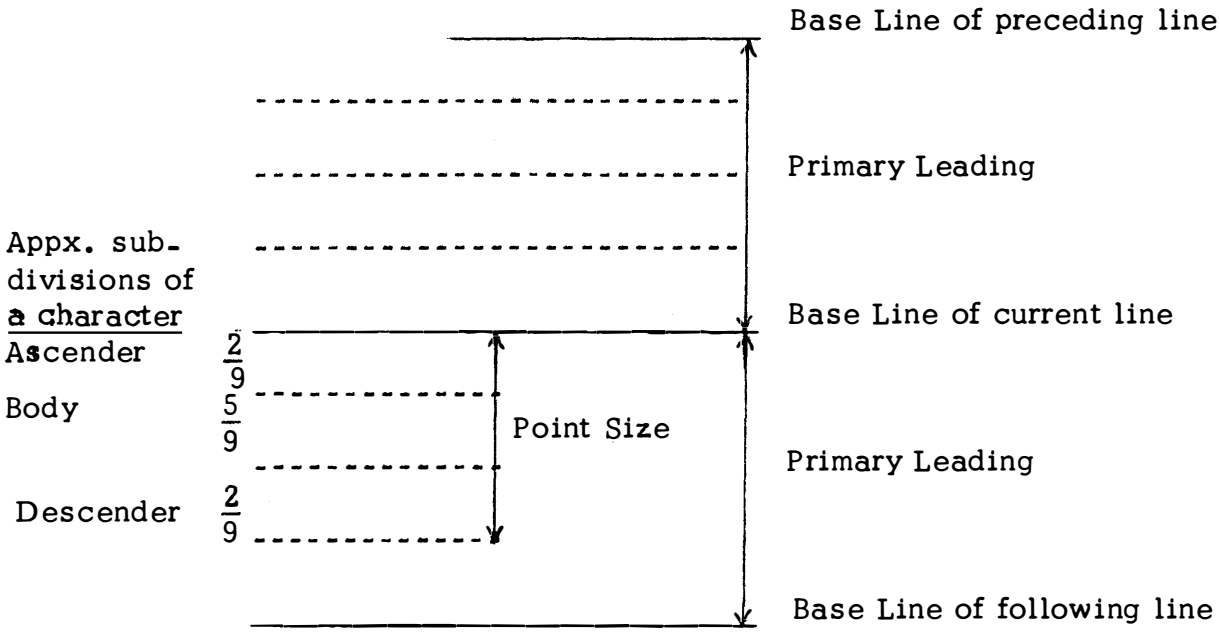
unshift
*a b c d e f g h i j k l m n o p q r s t u v w x y z * ` β [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '*

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ˇ _ → ? ï & ` (μ —) ˆ ê ! - : , . '
super shift
*λ α † Ω ∫ γ ' Δ * π < ± ' ˌ ν*

unshift
a b c d e f g h i j k l m n o p q r s t u v w x y z ° ² ³ [ ] 1 2 3 4 5 6 7 8 9 0 % ë 'n - ; , . '

SHIFT
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z + _ * ? ï & • (· —) □ / ê ! - : , . '
super shift
ı ¹ ₂ ₃ ⁴ ⁵ ₄ ˋ ˉ ₆ ₆ ⁹ ˑ ₉ ⁰ ₀

Relation of Point Size and Leading



Note : The base line for a character is conventionally at the foot of the ascender i.e. $\frac{2}{9}$ down from the top of a tall character such as h. The convention used here, namely putting the base line at the top of the ascender conforms with the current setting of the CSIR machine.

Table for Calculating the Number of M Spaces

M x 1	= 18	M x 31	= 558
2	= 36	32	= 576
3	= 54	33	= 594
4	= 72	34	= 612
5	= 90	35	= 630
6	= 108	36	= 648
7	= 126	37	= 666
8	= 144	38	= 684
9	= 162	39	= 702
10	= 180	40	= 720
11	= 198	41	= 738
12	= 216	42	= 756
13	= 234	43	= 774
14	= 252	44	= 792
15	= 270	45	= 810
16	= 288	46	= 828
17	= 306	47	= 846
18	= 324	48	= 864
19	= 342	49	= 882
20	= 360	50	= 900
21	= 378	51	= 918
22	= 396	52	= 936
23	= 414	53	= 954
24	= 432	54	= 972
25	= 450	55	= 990
26	= 468	56	= 1008
27	= 486	57	= 1026
28	= 504	58	= 1044
29	= 522	59	= 1062
30	= 540	60	= 1080

Relationships between Distances in Various Point Sizes

First \ Second	6	7	8	9	10	11	12	14	18
6	1	.86	.75	.67	.60	.55	.50	.43	.34
7	1.17	1	.87	.78	.70	.64	.58	.50	.39
8	1.33	1.14	1	.89	.80	.73	.67	.57	.44
9	1.50	1.29	1.13	1	.90	.82	.75	.64	.50
10	1.67	1.43	1.25	1.11	1	.91	.83	.71	.56
11	1.83	1.57	1.38	1.22	1.10	1	.92	.79	.61
12	2.00	1.71	1.50	1.33	1.20	1.09	1	.86	.67
14	2.33	2.00	1.75	1.56	1.40	1.27	1.17	1	.78
18	3.00	2.57	2.25	2.00	1.80	1.64	1.50	1.29	1

Given a distance in First Point Size, what is the corresponding distance in the Second Point Size ?

Multiply the distance in the First Point Size by the factor in the corresponding cell e.g. 10 in PS6 corresponding to 10 x .5 in PS18

Note : Only the Point Sizes shown in the table are available on the 747 machine

Appendix 5

Tabulation

Data

	Q15	Q27	Q36
TYPE 6	620	12	8
TYPE 7	2	2	105

Tabulation

Marker Table and Data Input

```
L.001 CSIR-FOR NIPR.        DISK:400 SCHOOLBOOK.MODE:TABULATION
L.002 $ @ ! % ?
L.003 08
L.004 S   $PS090$PL090$SL070$LL0450$SR $LD$LY$EM$EL$CR$$      PRIMARY
L.005 T   ?0519$NT$LL0090$EM$EC$EC$NT$CR$$                    TAB POSNS.
L.006 F   $IA090$UL$US$EL$CR$NT$SR ?0507$NT$SR|$CR$$          TOP LINE
L.007 M   ?0603$NT$TL030$CR$IA090$UL$US$EL$CR?0603$NT$CR$$    MIDDLE LINE
L.008 L   ?0603$NT$TL030$CR$IA090$UL$US$EL$CR$$               BOTTOM LINE
L.009 t   $NT$$        DATA LINE
L.010 e   $NT$CR$$     END OF DATA LINE
L.011 H   $HA$$        HALT
L.012 TABULATION 1
L.013 @S@T
L.014 @F
L.015 ?03@t@15@t@27@t@36@e
L.016 @M
L.017 ?02@tTYPE 6@t@20@t12@t8@e
L.018 @M
L.019 ?02@tTYPE 7@t2@t20t105@e
L.020 @L@H
?
```

```
 N.I.P.R. TYPESETTING PROGRAM (5.2)

INITIALISATION COMPLETED


CODES USED
N=   8
CD:LEN :FLAGS :TYPESETTING CODES IN HEX
S    32 A0A0A0  11F0F9F012F0F9F013F0F7F010F0F4F5F0401B40021AFC900180060A00
T    56 00A0A0  0510F0F0F9F0018007070510F0F0F9F0018007070510F0F0F9F0018007070510F0F0F9F0018007070510
                F0F0F9F0018007070500A00
F    45 A000A0  15F0F9F00227902006 0A0005401B400501184F900501184F900501184F900501184F900501184F900A00
M    34 A000A0  0505050505051 4F0F3F00A0015F0F9F00227902006 0A0005050505050500A00
L    26 A000A0  0505050505051 4F0F3F00A0015F0F9F00227902006 0A00
T     4 000000  05
E     6 0000A0  050A00
H     5 0000A0  0800

DELIMITERS AND MARKERS FOR THIS RUN
COMMAND DELIMITER          $    5B
TYPESETTER CODE MARKER     @    7C
IGNORE                          5A
TEXT DELIMITER             &    50
REPEAT DELIMITER           ?    6F
ERROR MARKERS:CODES        J    51
              TEXT         K    52


  208 USED OUT OF    800 AVAILABLE IN LCODE ARRAY

BATCH IDENTIFIER: TABULATION 1



END OF CONVERSION
-----------

     8 LINES ENTERED


     0 ERRORS

     4 RECORDS WRITTEN
```

Tabulation

Output on Magnetic Tape

```
REC.LENGTH=  120
  PS18    LL300    PL24     SL50  FBCIUSFLCRC S I R - F O R
11F1F8F010F0F3F0F012F2F4F013F5F0F003262006 0AC3E2C9D960C6D6D9
  N I P R .                CR D I S K ; 4 9 0   S C H O O L B
40D5C9D7D94B4040404040400 60AC4C9E2D27AF4F9F040E2C3C8D6D6D3C2
  O O K .  CR M O D E ; T A B U L A T I O N              CR
D6D6D24B060AD4D6C4C57AE3C1C2E4D3C1E3C9D6D5404040404040060A4040
                                                CR US  CR
4040404040404040404040404040404040404040060A20060A000000000000001
  P59   PL9     SL7     LL450      SR     LD - FM   CR    NT
11F0F9F012F0F9F013F0F7F010F0F4F5F0401B40021AFC0101060A000510
  LL90  FM FC FL  NT      ditto      NT      ditto      NT   ditto
F0F0F9F00107070510F0F0F9F00107070510F0F0F9F00107070510F0F0F9
        NT      ditto          NTCR   IA90    UL  US  CR  NT  SR
F00107070510F0F0F9F00107070 50A0015F0F9F00227012006 0A0005401B
  NT  SR]  NT      NT      NT      NT      CR  NTNT
4005011B4F0105011B4F0105011B4F0105011B4F0105011B4F010A0005052
  NT Q15  NT Q27  NT Q36  NTCR    6XNT      TL3   CR  IA90
0005D8F1F505D8F2F705D8F3F6050A000505050505 14F0F3F00A0015F0
      UL  US  CR    6XNT    CR  NTNT TYPE   6NT 620 NT
F9F00227012006 0A000505050505 0A000505E3E8D7C540F605F6F2F005
  12 NT 8 NTCR      6XNT      TL3   CR  IA90  UL  US  CR
F1F205F805 0A000505050505 14F0F3F00A0015F0F9F00227012006 0A00
   6XNT    CR  NTNT TYPE   7 NT 2 NT 2 NT 105  NTCR
0505050505050A000505E3E8D7C540F705F205F205F1F0F5050A00000003
   6XNT      TL3   CR  IA90  UL  US  CR  HA
0505050505 14F0F3F00A0015F0F9F00227012006 0A000B004040404040

4040404040404040404040404040404040404040404040404040404040

4040404040404040404040404040404040404040404040404040404040

4040404040404040404040404040404040404040404040404040404004

END OF INPUT.     4 RECORDS READ
```

Tabulation

Final Galley

CSIR-FOR NIPR.
DISK:490 SCHOOLBOOK.
MODE:TABULATION

	Q15	Q27	Q36
TYPE 6	620	12	8
TYPE 7	2	2	105

Mathematical Text
Data

We find *a priori* the degree of the polynomial $\Omega_N(x)$ or $\Delta_N(x)$ approximating a function $f(x)$, so that the maximum truncation error is less than explicit forms for $\omega_N(x)$ and $\delta_N(x)$ as contour integrals.

Let $f(x)$ be defined for $-1 \leq x \leq 1$ in the complex plane and $f(z)$ be defined for all z, such that $f(z) = f(x)$ in the interval $[-1,1]$. Let the truncation error $p_N(x) = f(x) - P_N(x)$.
Given

$$(1.1) \qquad a_{n,m} = \frac{1}{\pi i} \int_C \frac{f^m(z)\, dz}{\sqrt{(z^2 - 1)}(z + \sqrt{(z^2 - 1)})^n},$$

where C is a contour enclosing $-1 \leq \mathrm{Re}\ z \leq 1, \mathrm{Im}\ z = 0$, on and within which $f(z)$ is regular, and $| z + \sqrt{(z^2 - 1)} | > 1$ for all z except $-1 \leq \mathrm{Re}\ z \leq 1, \mathrm{Im}\ z = 0$.
Substituting for $a_{n,m}$ in $p_N(x) = \Sigma_{n=N+1}^{\infty} a_{n,m} L_n(x)$ and interchanging integral and sum,

$$(1.2) \qquad p_N^m(x) = \frac{1}{2\pi i} \int_C \frac{f^m(z)}{(z - x)\sqrt{(z^2 - 1)}} \left\{ \frac{L_{N+1}(x)}{(z + \sqrt{(z^2 - 1)})^m} - \frac{L_m(x)}{(z + \sqrt{(z^2 - 1)})^{N+1}} \right\} dz.$$

To obtain $\omega_N(x)$, we derive an expression for $A_{n,N}$ as a contour integral. If a function $f(z)$ is regular on and within C, then

$$(1.3) \qquad f(x) = \frac{1}{2\pi i} \int_C \frac{f(z)}{z - x}\, dz,$$

where x is any point within C.

Mathematical Text : Text without Equations

Marker Table for the First Run

```
L.001 CSIR_FOR NIPR.        DISK:490 SCHOOLBOOK.MODE:HYPHENLESS
L.002   @ ! & ?
L.003 46
L.004 0   $PS090$PL120$LL0713$SL000$FM$FL$CR$$              PRIMARY
L.005 1   ?2003$IG$TL060$FL$CR$PS060$LL1069$FI?2003$IG$US$FL$CR$$
L.006 2   ?2003$IG$TL060$FL$CR$PS090$LL0713$FM?2003$IG$US$FL$CR$$
L.007 3   ?2003$IG$TL030$FL$CR$PS060$LL1069$FI?2003$IG$US$FL$CR$$
L.008 4   ?2003$IG$TL030$FL$CR$PS090$LL0713$FM?2003$IG$US$FL$CR$$
L.009 5   ?2003$IG$TL060$FL$CR$PS140$LL0535$FM?2003$IG$US$FL$CR$$
L.010 m   $FM$$      medium font
L.011 i   $FI$$      italic
L.012 b   $FB$$      bold
L.013 x   $FI&x$FM$$                                        x
L.014 X   $FI&f$FM&($FI&x$FM&)$$                            f(x)
L.015 z   $FI&z$FM$$                                        z
L.016 Z   $FI&f$FM&($FI&z$FM&)$$                            f(z)
L.017 p   $FI&p$DK$FM&($FI&x$FM&)$$                         p (x)
L.018 P   $FI&P$DK$FM&($FI&x$FM&)$$                         P (x)
L.019 a   $FI&a?0203$DM$FM$$                                a
L.020 l   $FI&L$DK$FM&($FI&x$FM&)$$                         L (x)
L.021 S   $GZ$$                                             Σ
L.022 O   $GO$DK&($FI&x$FM&)$$                              Ω (x)
L.023 o   $GW$DK&($FI&x$FM&)$$                              ω (x)
L.024 Y   $GC$DK&($FI&x$FM&)$$                              Δ (x)
L.025 y   $GD$DK&($FI&x$FM&)$$                              δ (x)
L.026 u   $FI&z$FM& + $MR&($FI&z$TC$FM& _ 1)$$              z+√(z²_1)
L.027 W   $FI&A?0303$DN$FM$$                                A
L.028 s   $MO$$                                             [
L.029 t   $MC$$                                             ]
L.030 q   $OC?0503$MT$QC$$                                  ∞
L.031 A   $DA$$                    1 space
L.032 B   $DB$$                    2
L.033 C   $DC$$                    3
L.034 D   $DD$$                    4
L.035 E   $DE$$                    5
L.036 F   $DF$$                    6
L.037 G   $DG$$                    7
L.038 H   $DH$$                    8
L.039 N   $DN$$                    9
L.040 I   $DI$$                    10
L.041 J   $DJ$$                    11
L.042 K   $DK$$                    12
L.043 L   $DL$$                    15
L.044 M   $DM$$                    18
L.045 *   ?3003$IG$LT$$                  DELAY
L.046 T   $LT$$                          SKIP PADDING
L.047 f   ?2003$IG$TL500$FL$CR$$         SPACE
L.048 e   $FL$CR$$                       PARA END
L.049 h   $FL$CR$HA$$                    HALT
```

Mathematical Text : Text without Equations

Text Input for the First Run

```
L.049 PRINTING MATHS 7 - TEXT
L.050 @@
L.051 We find @ia priori@m the degree of the polynomial @Q or @Y approximating@T
L.052  a function @X, so that the maximum truncation error is less than exp@T
L.053 licit forms for @o and @y as contour integrals.@e
L.054    Let @X be defined for -1 <= @x <= 1 in the complex plane and @Z be@T
L.055  defined for all @z, such that @Z = @X in the interval @s-1,1@t. Let the@T
L.056  truncation error @p = @X - @P.@e
L.057    Given@f
L.058 (1)@f
L.059 where C is a contour enclosing -1 <= Re @z <= 1, Im @z = 0, on and with@T
L.060 in which @Z is regular, and |@u| > 1 for all @z except -1 <= Re @z <= 1,@T
L.061  Im @z = 0. Substituting for @a in @p = ?@4@M@a @1 and interchanging@T
L.062  the integral and sum@f
L.063 (2)@f
L.064    To obtain @o we derive an expression for @' as a contour integral.@T
L.065  If a function @Z is regular on and within C, then@f
L.066 (3)@f
L.067 where @x is any point within C.@h
?
```

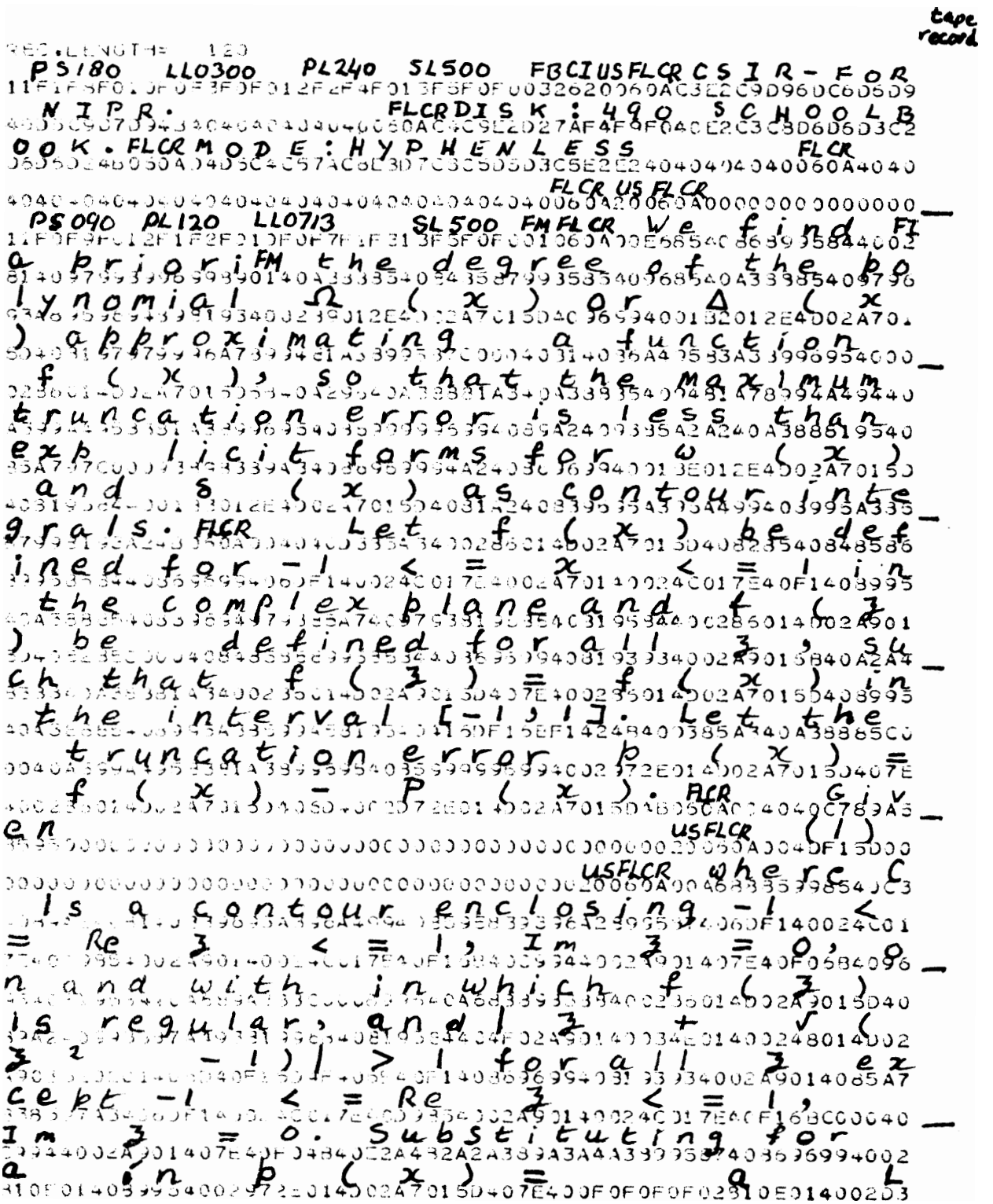
```
N.I.P.R. TYPESETTING PROGRAM (5.2)

INITIALISATION COMPLETED

CODES USED
N= 46
CD:LEN :FLAGS :TYPESETTING CODES IN HEX
0    25 00A0A0  11F0F9F012F1F2F010F0F7F1F313F0F0F00180060A00
1    65 00A0A0  000000000000000000000000000000000000014F0F6F0060AD011F0F6F010F1F0F6F90280000000
        000000000020060A00
2    65 00A0A0  000000000000000000000000000000000000014F0F6F0060AD011F0F9F010F0F7F1F3018000000
        000000000020060A00
3    65 00A0A0  000000000000000000000000000000000000014F0F3F0060AD011F0F6F010F1F0F6F90280000000
        000000000020060A00
4    65 00A0A0  000000000000000000000000000000000000014F0F3F0060AD011F0F9F010F0F7F1F3018000000
        000000000020060A00
5    65 00A0A0  000000000000000000000000000000000000014F0F6F0060AD011F1F4F010F0F5F3F5018000000
        000000000020060A00
M     5 00A000  0180
I     5 00A000  0280
B     5 00A000  0380
X     8 00A000  0280A70180
X    15 00A000  02808601804D0280A7018050
Z     8 00A000  0280A90180
Z    15 00A000  02808601804D0280A9018050
P    16 00A000  0280972E01804D0280A7018050
P    16 00A000  0280D72E01804D0280A7018050
A    10 00A000  0280810F0F0180
L    16 00A000  0280D32E01804D0280A7018050
S     6 A00000  018F90
O    14 A0A000  0289902E4D0280A701805D
D    14 A0A000  018E902E4D0280A701805D
Y    14 A0A000  0182902E4D0280A701805D
Y    14 A0A000  0183902E4D0280A701805D
U    30 A0A000  0280A9018040034E9040024890 4D0280A90332900180406D40F15D
W    11 00A000  0280C10E0E0E0180
S     4 000000  41
T     4 000000  42
Q    14 A00000  015590282828282801559 0
A     4 000000  0D
B     4 000000  1F
C     4 000000  2C
D     4 000000  9A
E     4 000000  2D
F     4 000000  9B
G     4 000000  9C
H     4 000000  9D
N     4 000000  0E
I     4 000000  9E
J     4 000000  9F
K     4 000000  2E
L     4 000000  2F
M     4 000000  0F
```

Lines continue with a further 24 zeros
(12 Ignore characters)

```
*    35 0000A0  0000000000000000000000000000000000000000000000000000C000
T     5 0000A0  C000
F    30 0000A0  0000000000000000000000000000000000000014F5F0F0060A00
E     6 0000A0  060A00
H     8 0000A0  060A000800

DELIMITERS AND MARKERS FOR THIS RUN
COMMAND DELIMITER          $    5B
TYPESETTER CODE MARKER     @    7C
IGNORE                          5A
TEXT DELIMITER             &    50
REPEAT DELIMITER           ?    6F
ERROR MARKERS:CODES        J    51
              TEXT         K    52

  734 USED OUT OF   800 AVAILABLE IN LCODE ARRAY

BATCH IDENTIFIER: PRINTING MATHS 7 _ TEXT


END OF CONVERSION
-----------

   21 LINES ENTERED

    0 ERRORS

   26 RECORDS WRITTEN
```

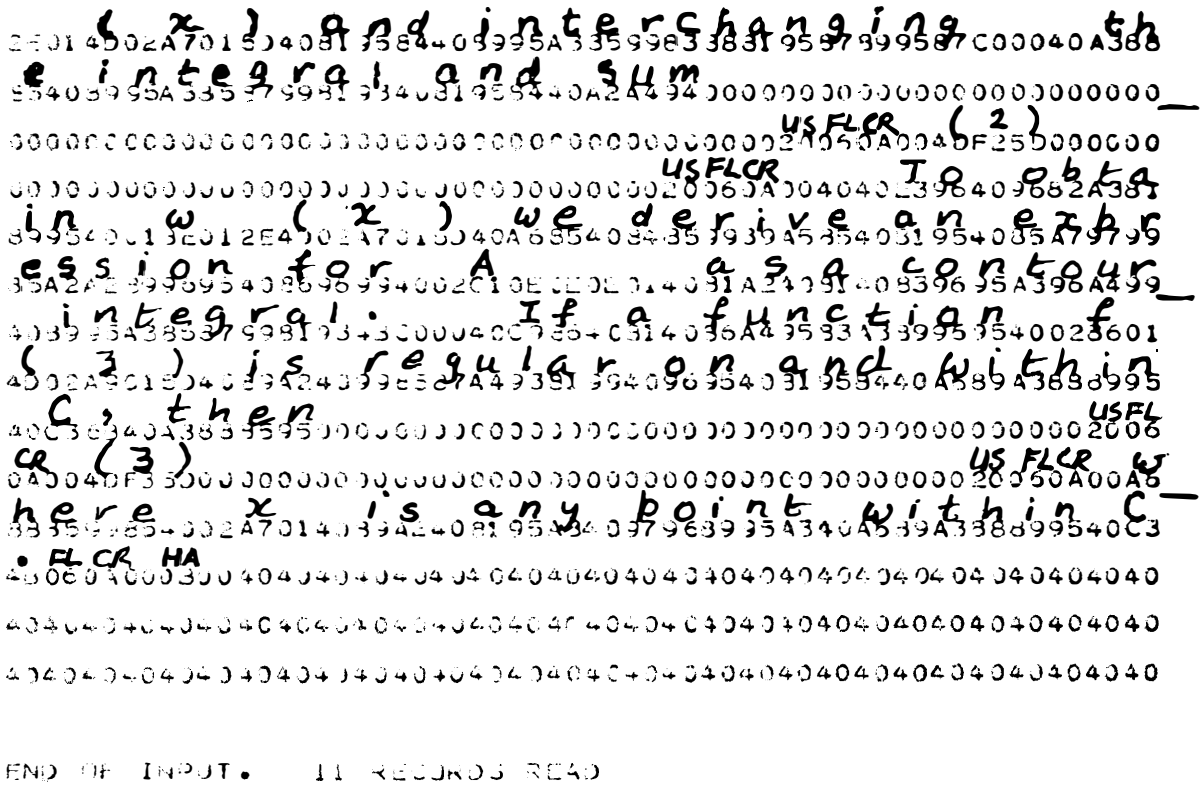
Mathematical Text : Text without Equations

Output on Magnetic Tape



Mathematical Text : Text without Equations

Output on Magnetic Tape - continued



Mathematical Text : Text without Equations

Galley Proof from First Run

CSIR—FOR NIPR.
DISK:490 SCHOOLBOOK.
MODE:HYPHENLESS

We find *a priori* the degree of the polynomial $\Omega\ (x)$ or $\triangle\ (x)$ approximating a function $f(x)$, so that the maximum truncation error is less than explicit forms for $\omega\ (x)$ and $\delta\ (x)$ as contour integrals.

Let $f(x)$ be defined for $-1 <= x <= 1$ in the complex plane and $f(z)$ be defined for all z, such that $f(z) = f(x)$ in the interval $[-1,1]$. Let the truncation error $p\ (x) = f(x) - P\ (x)$.

Given

(1)

where C is a contour enclosing $-1 <= \mathrm{Re}\, z <= 1$, $\mathrm{Im}\, z = 0$, on and within which $f(z)$ is regular, and $|z + \sqrt{(z^2 - 1)}| > 1$ for all z except $-1 <= \mathrm{Re}\, z <= 1$, $\mathrm{Im}\, z = 0$. Substituting for a in $p\ (x) =$ $a\ L\ (x)$ and interchanging the integral and sum

(2)

To obtain $\omega\ (x)$ we derive an expression for A as a contour integral. If a function $f(z)$ is regular on and within C, then

(3)

where x is any point within C.

```
L.050 PRINTING MATHS 7 _ TEXT
L.051 @0
L.052 We find @ia priori@m the degree of the polynomial @0 or @Y approximating a@T
L.053 @1?35@M@DN?06@MN@2
L.054  function @X, so that the maximum truncation error is less than exp@T
L.055 licit forms for @o and @y as contour integrals.@1@M@CN?06@M@IN@2@*@e
L.056   Let @X be defined for _1 <= @x <= 1 in the complex plane and @Z be@T
L.057  defined for all @z, such that @Z = @X in the interval @s_1,1@t. Let the@T
L.058  truncation error @p = @X@1?51@M@GN@2 _ @P.@1?03@M@CN@2@*@e
L.059   Given@f
L.060 (1)@f@*
L.061 where C is a contour enclosing _1 <= Re @z <= 1, Im @z = 0, on and with@T
L.062 in which @Z is regular, and |@u| > 1 for all @z except _1 <= Re @z <= 1,@T
L.063  Im @z = 0.@5?10@M@N@S@3?26@M@H@q@4
L.064  Substituting for @a in @p = ?03@M@L@a@1 and interchanging integral@T
L.065 @1?13@M@Jn,m?04@M@EN?05@M@Kn=N@m+1@i@M@Ln,m?02@Mn@2
L.066  and sum@f
L.067 (2)@f@*
L.068   To obtain @o we derive an expression for @W as a contour integral.@T
L.069  If a function@1?08@M@L@AN?23@M@Bn,N@2 @Z is regular on and within C, then@f
L.070 (3)@f@*
L.071 where @x is any point within C.@h
```

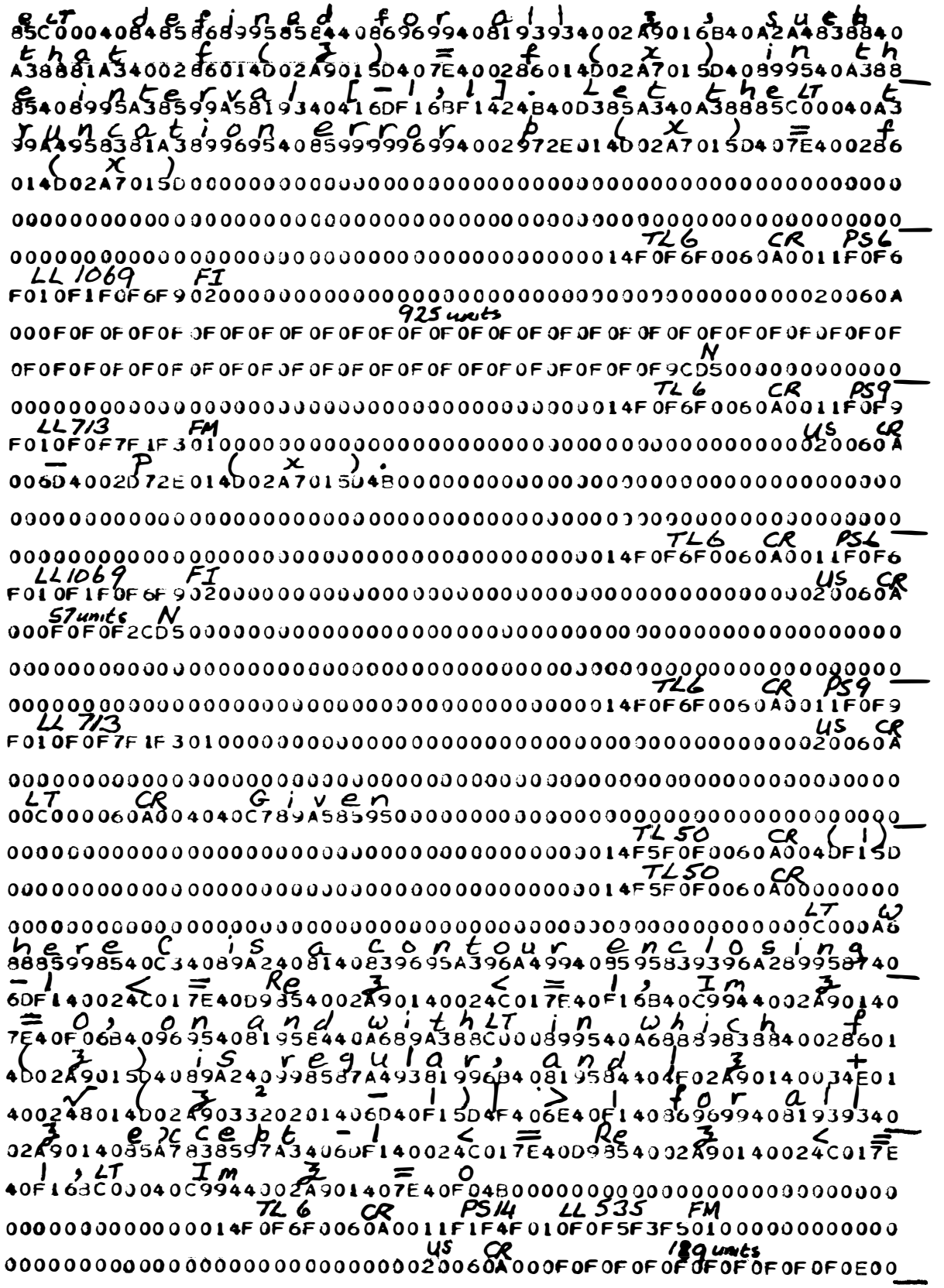
Mathematical Text : Text without Equations

Output (with Subscripts) on Magnetic Tape

```
REC.LENGTH=  120                                                    tape
  PS18   LL 300     PL24    SL50   FBCIUSFLCR C S I R - F O R       record
11F1F8F010F0F3F0F012F2F4F013F5F0F0032620060AC3E2C9D96DC6D6D9
  N I P R .                CR D I S K : 490      S C H O O L B
40D5C9D7D94B4040404040400060AC4C9E2D27AF4F9F040E2C3C8D6D6D3C2
 O O K .  CR M O D E : H Y P H E N L E S S                CR
D6D6D24B060AD4D6C4C57AC8E8D7C8C5D5D3C5E2E2404040404000060A4040
                                                CR US CR
4040404040404040404040404040404040404040400060A20060A000000000000000
  PS9    PL12    LL713      SL50   FM  CR We find   FI
11F0F9F012F1F2F010F0F7F1F313F0F0F001060A00E685408689958440002
a priori FM the degree of the po
814097998996998901400A38885400485997993585409688640A38885409796
lynomial Ω(x) or Δ(x)
93A8959694898193400289012E4D02A7015D4096994082 2E4D02A7015D40
approximating a LT
819797996A7899481A38995874081C000000000000000000000000000000
                                                TL6   CR   PS6
000000000000000000000000000000000000000000014F0F6F0060A0011F0F6
 LL1069    FI                                              US  CR
F010F1F0F6F9020000000000000000000000000000000000000020060A
                          634 units
000F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F
          N  108 units     N
0F0F0F0F0F0F9AD50F0F0F0F0F0F0D500000000000000000000000000000000
                                                TL6   CR   PS9
000000000000000000000000000000000000000000014F0F6F0060A0011F0F9
 LL713     FM                                              US  CR
F010F0F7F1F3010000000000000000000000000000000000000020060A
   function f(x), so that t
004086A49583A3899695400286014D02A7015D6B40A29640A38881A340A3
he maximum truncation error is
888540948 1A73994A49440A399A4958381A33996954085999996994089A2
 less than exp LT licit forms fo
409385A2A240A388819540 85A797C0009389838 9A3408696999 4A2408696
r ω(x) and δ(x) as conto
99408E2E4D02A7015D40819584408 32E4D02A7015D4091A2403396 95A396
ur integrals.
A4994089 95A3858799819 3A24B000000000000000000000000000000000

000000000000000000000000000000000000000000000000000000000000
                                                TL6   CR   PS6
000000000000000000000000000000000000000000014F0F6F0060A0011F0F6
  LL1069    FI                                              US  CR
F010F1F0F6F9020000000000000000000000000000000000000020060A
 21units N      118 units   N
000F2CD50F0F0F0F0F0F0F9ED500000000000000000000000000000000000

000000000000000000000000000000000000000000000000000000000000
                                                TL6   CR   PS9
000000000000000000000000000000000000000000014F0F6F0060A0011F0F9
   LL713     FM                                              US  CR
F010F0F7F1F3010000000000000000000000000000000000000020060A

000000000000000000000000000000000000000000000000000000000000
 LT    CR    Let f(x) be define
00C000060A0040400385A3400286014D02A7015D4092854084958689958 5
d for -1 < = x < = 1 in th
8440869699406DF140024C017E4002A70140024C017E40F1409995 40A388
e complex plane and f(z) b
8540839694977938 5A74097938195854081958440028601 4D02A9015D4082
```

Mathematical Text : Text without Equations

Output (with Subscripts) on Magnetic Tape Continued



Mathematical Text: Text without Equations

Output (with Subscripts) on Magnetic Tape Continued

Mathematical Text : Text without Equations

Text Input with Subscripts

CSIR—FOR NIPR.
DISK:490 SCHOOLBOOK.
MODE:HYPHENLESS

We find *a priori* the degree of the polynomial $\Omega_N(x)$ or $\triangle_N(x)$ approximating a function $f(x)$, so that the maximum truncation error is less than explicit forms for $\omega_N(x)$ and $\delta_N(x)$ as contour integrals.

Let $f(x)$ be defined for $-1 <= x <= 1$ in the complex plane and $f(z)$ be defined for all z, such that $f(z) = f(x)$ in the interval $[-1,1]$. Let the truncation error $p_N(x) = f(x) - P_N(x)$.

Given

(1)

where C is a contour enclosing $-1 <= \mathrm{Re}\, z <= 1, \mathrm{Im}\, z = 0$, on and within which $f(z)$ is regular, and $|z + \sqrt{(z^2 - 1)}| > 1$ for all z except $-1 <= \mathrm{Re}\, z <= 1, \mathrm{Im}\, z = 0$. Substituting for $a_{n,m}$ in $p_N(x) = \sum_{n=N+1}^{\infty} a_{n,m} L_n(x)$ and interchanging integral and sum

(2)

To obtain $\omega_N(x)$ we derive an expression for $A_{n,N}$ as a contour integral. If a function $f(z)$ is regular on and within C, then

(3)

where x is any point within C.

Determining Spacing Codes

As it is difficult to measure distances exactly with the transparency, two or three runs with small and often opposed adjustments in spacing may be necessary before an aesthetically pleasing result is obtained.

In order to keep track of these adjustments some systematic layout showing the spacing used and subsequent changes is very useful. Two such layouts are shown on the following pages - one for subscripts in text and the other for mathematical expressions.

The method is quite simple. A line is used for each set of characters which have a common Point Size and base line. Distances between characters are measured on the original manuscript or from the galley and written down in brackets and then converted to M spaces and a remainder using the table in Appendix 4 p. (ii). In the case of mathematical expressions it is more useful to measure all distances in terms of the Point Size used for the text and to convert to the actual Point Size used for characters such as integral signs and subscripts. Having all spacing in a common unit greatly assists the placing of different elements of the expression.

When a space is obtained in M space units, the coding required is easily derived.

For example; if the Marker Table shows

@D	ØDDØØ	4 spaces
@M	ØDMØØ	18 spaces

a space of 35 M+4 would be written as ?35@M@D

Spacing for Sub - and Superscripts and Other Symbols in Text

(See Appendix 6 P. (i) and (viii))

Char. widths: n(12), N(16), m(17), 9(5), Σ (14), + (10), = (10)

1.

35M+4 (634) 634 650 N

6M (108) 758 N

2.

M+3 (21) 21 37 N

6M+10 (118) 155 N

3.

51M+7 (925) 925 N

4.

3M+3 (57) 57 N

5.

10M+9 (189) 189 203 Σ

(PS 14)

(PS6) 440 476 Σ

6.

26M+8 (476) 476 ∞

7.

13M+11 (245) 245 (34) 279 n,m

4M+5 (76) 355 371 N

5M+12 (102) 473 (66) 539 n=N+1

M+15 (33) 572 606 n,m

2M (36) 642 n

8.

8M+15 (159) 159 175 N

23M+2 (416) 591 n,m

Spacing for a Mathematical Expression

(See Appendix 8 pp. (ii) and (ix))

Starting at line 61.

									Total
Spacing	9 (132)	18 (66) = 3M+12	9 9 132 150	9 (142)	15 (71) = 3M+17	9 (332)	18 (166) = 9M+4		644
widths			9 18	1 10				1 10	
PS18 : large symbols			∫	[				]	
Spacing	(157) (236) = 18M + 2								
widths			10						
PS6 : superscripts			m						
Spacing	(92) = 5M+2		(104) = 5M+14		(96) 5M+6		(102) = 5M+12		566
widths		10 10		58 40	39 66		54 54		
PS9 numerator		1		f (z)	L (x)		L(x)		
Spacing		9 (358)	6 (537) = 29M + 15		9 (138)	6 (207) = 11M+9			782
widths					28		10		
PS9 subscripts					N+1		m		
Spacing	(18)	(12)	(36)	(12)	(6) (6)		(12)		651
widths	36	10	30 = 5V	132 = 22V	138 = 23V	10	168 = 28V	20 5	
PS9 : fraction line	(1·2)	=	———	———	———	–	———	dz9	
Spacing			6 (651) = 36M + 3				6 (234) = 13M		923
widths					10		28		
PS6 : superscripts					m		N+1		
Spacing	(84) = 4M+12	(39) = 2M+3	(10)		(35)				614
widths	2 26	1 154	1 133		1 133				
PS9 : denominator	2πi	(z-x)√(z²-1)	(z+√(z²-1))		(z+√(z²-1))				
Spacing	6 (211) = 10M+13								
width	10								
PS6 lower limit	C								

Determining the Command Strings for Changing the Point Size and Base Line

The pro-forma on the following page provides a reasonable representation with a small number of command strings. Five base lines suffice for all parts of a rational expression which means a corresponding limited number of command strings for changing base line and Point Size.

Points to note are :-

a) A new Point Size must be screened from preceding text by a Carriage Return and must be followed by a Carriage Return to bring it into operation otherwise part of the following text may be processed at the previous Point Size. The Line Length must at the same time be converted into the new units.

b) It has been found in practice that the sequence that appears to be the least subject to the overwriting of successive Leadings is to have the Tape Leading to the new Base Line, Carriage Return, new Point Size, a Secondary Leading of zero and Carriage Return. It is advisable to precede Leadings by a delay of 20 Ignore characters to minimise the possibility of overwriting. Further delays should be inserted using the Delay command string when the line is very short.

A typical command string is thus :-

?2003ØIGØTL060ØFLØCRØPS060ØLL1069ØFLØUSØCR

Base Line

0 —
1 —
2 —
3 —
4 —

$$a^{N}_{m}(y) = \int_{C} \frac{f^{N}_{N,m}(y)^{m}}{g^{m}_{N,m}(z)^{N}}\, dz$$

On Base Line 0 : Integral Signs, upper limits of integration.
Large brackets

1 : Numerator character and numerator superscripts

2 : Numerator subscripts
Fraction line, fraction line characters, superscripts for fraction line characters.

3 : Subscripts for fraction line characters
Denominator and denominator superscripts

4 : Subscripts for denominator
Lower limits of integration

Mathematical Text : Equations

Marker Table

```
L.001 CSIR-FOR NIPR.        DISK:490 SCHOOLBOOK.MODE:HYPHENLESS
L.002   0 ! & ?
L.003 45
L.004 0   $PS090$PL120$LL0713$SL000$FM$FL$CR$$          PRIMARY
L.005 1   ?2003$IG$TL000$FL$CR$PS090$LL0713$FM$US$FL$CR$$   LEAD 0,PS 9
L.006 2   ?2003$IG$TL060$FL$CR$PS090$LL0713$FM$US$FL$CR$$   LEAD 6,PS 9
L.007 3   ?2003$IG$TL000$FL$CR$PS060$LL1060$FI$US$FL$CR$$   LEAD 0,PS 6
L.008 4   ?2003$IG$TL060$FL$CR$PS060$LL1069$FM$US$FL$CR$$   LEAD 6,PS 6
L.009 5   ?2003$IG$TL000$FL$CR$PS180$LL0355$FM$US$FL$CR$$   LEAD 0,PS 18
L.010 6   ?2003$IG$TL020$FL$CR$PS090$LL0713$FM$US$FL$CR$$   LEAD 2,PS 9
L.011 m   $FM$$      medium font
L.012 i   $FI$$      italic
L.013 b   $FB$$      bold
L.014 x   $FI&x$FM$$                                  x
L.015 X   $FI&f$FM&($FI&x$FM&)$$                      f(x)
L.016 z   $FI&z$FM$$                                  z
L.017 Z   $FI&f$FM&($FI&z$FM&)$$                      f(z)
L.018 Y   $FI&f$DK$FM&($FI&z$FM&)$$                   f (z)
L.019 d   $FI&dz$FM$$                                 dz
L.020 p   $FI&p$DK$FM&($FI&x$FM&)$$                   p (x)
L.021 a   $FI&a$DM$DI$FM$$                            a
L.022 l   $FI&L$DK$FM&($FI&x$FM&)$$                   L (x)
L.023 k   $FI&L$DM$DK$FM&($FI&x$FM&)$$                L  (x)
L.024 q   $FI$GP&i$FM$$                               πi
L.025 r   $MR&($FI&z$TC& $LY$FM& 1)$$                 √(z² - 1)
L.026 u   $FI&z$FM& + $MR&($FI&z$TC$FM& _ 1)$$        z+√(z²-1)
L.027 s   $MO$$          [
L.028 t   $MC$$          ]
L.029 j   $MI$$          ∫
L.030 v   $MV$$
L.031 A   $DA$$          1 space
L.032 B   $DB$$          2
L.033 C   $DC$$          3
L.034 D   $DD$$          4
L.035 E   $DE$$          5
L.036 F   $DF$$          6
L.037 G   $DG$$          7
L.038 H   $DH$$          8
L.039 N   $DN$$          9
L.040 I   $DI$$          10
L.041 J   $DJ$$          11
L.042 K   $DK$$          12
L.043 L   $DL$$          13
L.044 M   $DM$$          18
L.045 *   ?3003$IG$LT$$                DELAY
L.046 T   $LT$$                        SKIP PADDING
L.047 f   ?2003$IG$TL500$FL$CR$$       SPACE
L.048 h   $FL$CR$HA$$                  HALT
```

Mathematical Text : Equations

Data Input

```
L.040 PRINTING MATHS 6b- EQUATIONS
L.050 @0
L.051 (1)@f@*@5
L.052 ?07@M@K@j@*@6
L.053 ?13@M@F1?07@M@I@Y@d@3
L.054 ?33@Mm@*@2
L.055 (1.1)?06@M@H@a@K=@K?04@v?02@M@D?37@v,@4
L.056 ?14@Mn,m?26@M@En@*@1
L.057 ?13@M@C@q?02@M@E@r(@u)@*@4
L.058 ?23@M@I@CC@f@*@1
L.059 (2)@f
L.060 ?03@M@p@*@3?05@M@K@Am@*@4?05@M@K@AN@*@2@*@5
L.061 ?03@M@K@j?03@M@L@B@s?09@M@D@t@6
L.062 ?05@M@B1?05@M@D@I@Y?05@M@F@k?05@M@K@1@3
L.063 ?18@M@Bm@*@4
L.064 ?29@M@LN+1?11@M@Nm@*@1
L.065 (1.2)@M=@K?05@v?02@M?22@v@K?23@v _ ?28@v@K@d,@2
L.066 ?04@M@K2@q?02@M@C(@z _ @x)@r@I(@u)@M@L(@u)@3
L.067 ?36@M@Cm?13@MN+1@*@4
L.068 ?11@M@I@CC@f@*@1
L.069 (3)@f@*@5
L.070 ?10@M@F@j@*@6
L.071 ?10@M1?03@M@Z@*@2
L.072 (1.3)?12@M@C@X@K=@K?05@v?02@M?08@v@C@d,@2
L.073 ?18@M@C2@q?02@M@D@z _ @x@*@4
L.074 ?31@M@I@DC@*@h
```

```
N.I.P.R. TYPESETTING PROGRAM (5.2)

INITIALISATION COMPLETED

CODES USED
N= 45
CD:LEN :FLAGS :TYPESETTING CODES IN HEX
0    25 00A0A0  11F0F9F012F1F2F010F0F7F1F313F0F0F00180060A00
1    45 00A0A0  00000000000000000000000000000000000000000014F0F0F0060A0011F0F9F010F0F7F1F3018020060A00
2    45 00A0A0  00000000000000000000000000000000000000000014F0F6F0060A0011F0F9F010F0F7F1F3018020060A00
3    45 00A0A0  00000000000000000000000000000000000000000014F0F0F0060A0011F0F6F010F1F0F6F9028020060A00
4    45 00A0A0  00000000000000000000000000000000000000000014F0F6F0060A0011F0F6F010F1F0F6F9018020060A00
5    45 00A0A0  00000000000000000000000000000000000000000014F0F0F0060A0011F1F8F010F0F3F5F6018020060A00
6    45 00A0A0  00000000000000000000000000000000000000000014F0F2F0060A0011F0F9F010F0F7F1F3018020060A00
M     5 00A000  0180
I     5 00A000  0280
B     5 00A000  0380
X     8 00A000  0280A70180
X    15 00A000  02808601804D0280A701805D
Z     8 00A000  0280A90180
Z    15 00A000  02808601804D0280A901805D
Y    16 00A000  0280862E01804D0280A901805D
D     9 00A000  028084A90180
P    16 00A000  0280972E01804D0280A701805D
A    10 00A000  0280810F9D0180
L    16 00A000  0280D32E01804D0280A701805D
K    17 00A000  0280D30F2E01804D0280A701805D
Q    11 A0A000  0280028A90890180
R    22 A0A000  0248904D0280A90332904002FC90018040F15D
U    30 A0A000  0280A901804003 4E904002489 04D0280A90332900180406D40F15D
S     4 000000  41
T     4 000000  42
J     6 A00000  024490
V     6 A00000  034990
A     4 000000  0D
B     4 000000  1F
C     4 000000  2C
D     4 000000  9A
E     4 000000  2D
F     4 000000  9B
G     4 000000  9C
H     4 000000  9D
N     4 000000  0E
I     4 000000  9E
J     4 000000  9F
K     4 000000  2E
L     4 000000  2F
M     4 000000  0F
*    35 0000A0  00000000000000000000000000000000000000000000000000000000C000
T     5 0000A0  C000
H    30 0000A0  00000000000000000000000000000000000000000014F5F0F0060A00
      8 0000A0  060A000B00
```

Mathematical Text : Equations

Computer Output - Continued

```
DELIMITERS AND MARKERS FOR THIS RUN
COMMAND DELIMITER          $    5B
TYPESETTER CODE MARKER     @    7C
IGNORE                          5A
TEXT DELIMITER             &    50
REPEAT DELIMITER           ?    6F
ERROR MARKERS:CODES        J    51
              TEXT         K    52

  657 USED OUT OF   800 AVAILABLE IN LCODE ARRAY

BATCH IDENTIFIER: PRINTING MATHS 6B- EQUATIONS


END OF CONVERSION
-----------

   25 LINES ENTERED

    0 ERRORS
   27 RECORDS WRITTEN
```

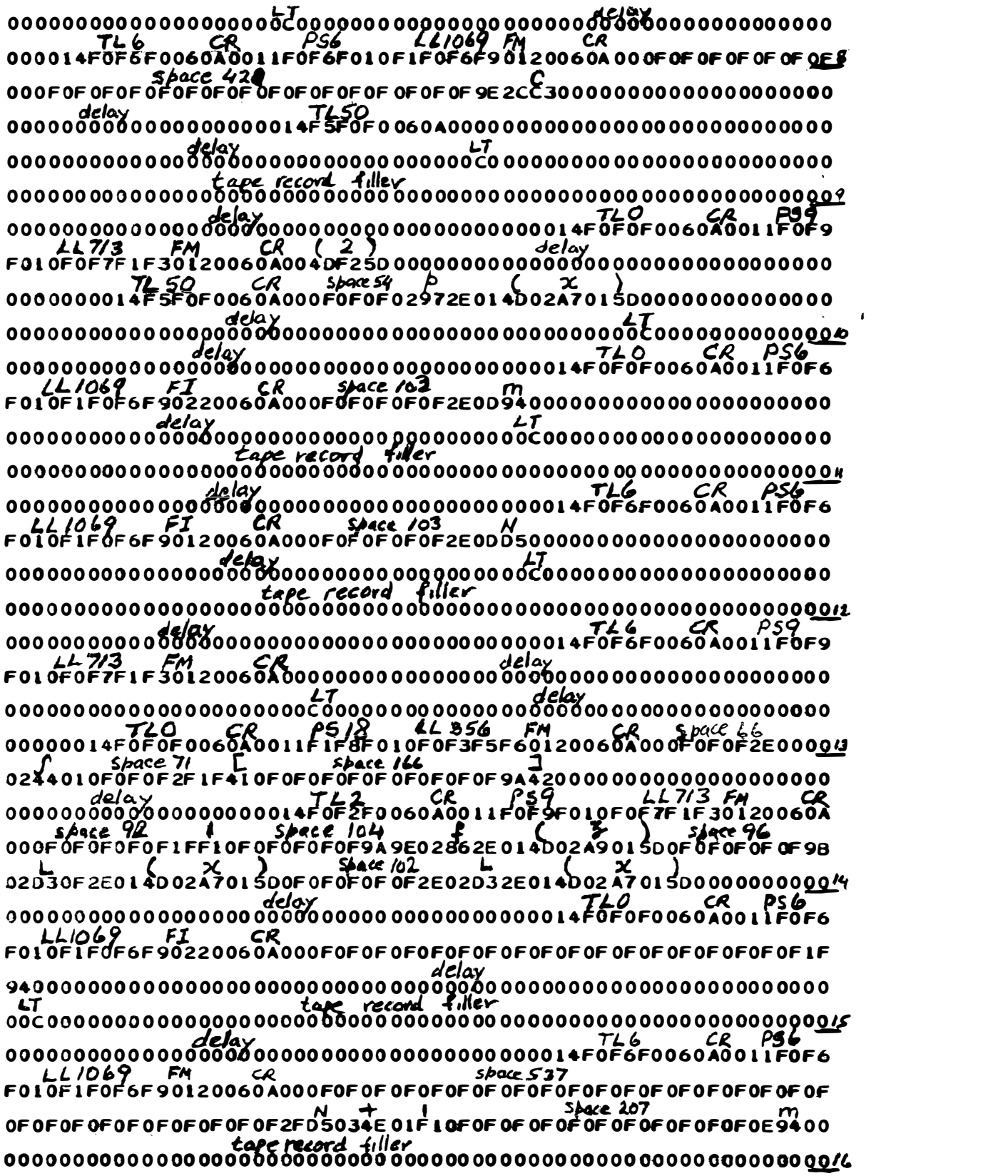
Mathematical Text : Equations

Output on Magnetic Tape

```
                                                                        Tape
                                                                        Records
REC.LENGTH=   120
  PS18      LL300       PL24      SL50   FB CIUSFLCR C S I R - F O R
11F1F8F010F0F3F0F012F2F4F013F5F0F0032620060AC3E2C9D960C6D6D9
  N I P R .                    CR D I S K : 490     S C H O O L B
40D5C9D7D94B40404040404040060AC4C9E2D27AF4F9F040E2C3C8D6D6D3C2
 O O K .   CR M O D E : H Y P H E N L E S S                 CR
D6D6D24B060AD4D6C4C57AC8E8D7C8C5D5D3C5E2E2404040404040060A4040
                                              CR US  CR
404040404040404040404040404040404040404040060A20060A0000000000000000   1
  PS9   PL12   LL7/3              SL0    FM    CR  ( 1 )
11F0F9F012F1F2F010F0F7F1F313F0F0F001060A004DF15D000000000000000
          delay                        TL50   CR
0000000000000000000000000014F5F0F0060A0000000000000000000000000
                delay                              LT
000000000000000000000000000000000000000000000000000C0000000000000000000
                   tape record  filler
00000000000000000000000000000000000000000000000000000000000000000000000   2
                delay                              TL0    CR   PS18
00000000000000000000000000000000000000000000014F0F0F0060A0011F1F8
   LL356   FM    CR            space 136      ∫
F010F0F3F5F60120060A000F0F0F0F0F0F0F2E024401000000000000000000
                 delay                               LT
0000000000000000000000000000000000000000000000000000C00000000000000000
                 tape  record  filler
00000000000000000000000000000000000000000000000000000000000000000000000   3
               delay                                 TL2    CR   PS9
0000000000000000000000000000000000000000000000014F0F2F0060A0011F0F9
   LL7/3    FM    CR             space 240            1    space 136
F010F0F7F1F30120060A000F0F0F0F0F0F0F0F0F0F0F0F0F9BF10F0F0F0F
      f        (    z     )     d z                 delay
0F0F0F9E02862E014D02A9015D0284A901000000000000000000000000000000
                TL0        CR     PS6       LL1069  FI      CR
00000000000000014F0F0F0060A0011F0F6F010F1F0F6F90220060A000F0F   4
                                space 594
000F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F
   m                                    delay
0F0F94000000000000000000000000000000000000000000000000000000000000000
      LT                      delay                               TL6
000000C00000000000000000000000000000000000000000000000000014F0F6F006
  CR     PS9          LL7/3  FM     CR  ( 1 . 1 )       space 116
0A0011F0F9F010F0F7F1F30120060A004DF14BF15D0F0F0F0F0F0F9D0000   5
FI a     FM    =         4x—          space 40
02810F9D012E7E2E034949494901 0F0F9A03494949494949494949494949
                                37x —
49494949494949494949494949494949494949494949494949494949016B000000
               delay                                TL6     CR     PS6
0000000000000000000000000000000000000014F0F6F0060A0011F0F6F010F1
   FM      CR              space 252                     n , m
F0F6F90120060A000F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F956B940F0F0F0F0F   6
              space 473                                    n
000F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F2D9500000000000000
                        delay                                 LT
00000000000000000000000000000000000000000000000000000000C000000000000
                      delay                  TL0       CR     PS9
00000000000000000000000000000000000014F0F0F0060A0011F0F9F010F0F7
LL7/3        CR           space 237                    π  i    space 41
F1F30120060A000F0F0F0F0F0F0F0F0F0F0F0F0F2C02BA89010F0F2D0000   7
   √       (     z    2           —          1  ) (     z            +            √       (    z
0248014D02A9033202400FC0140F15D4D02A9014003 4E014002480 14D02A9
    2           —         1  ) )                                    delay
03320201406D40F15D5D0000000000000000000000000000000000000000000000000
```

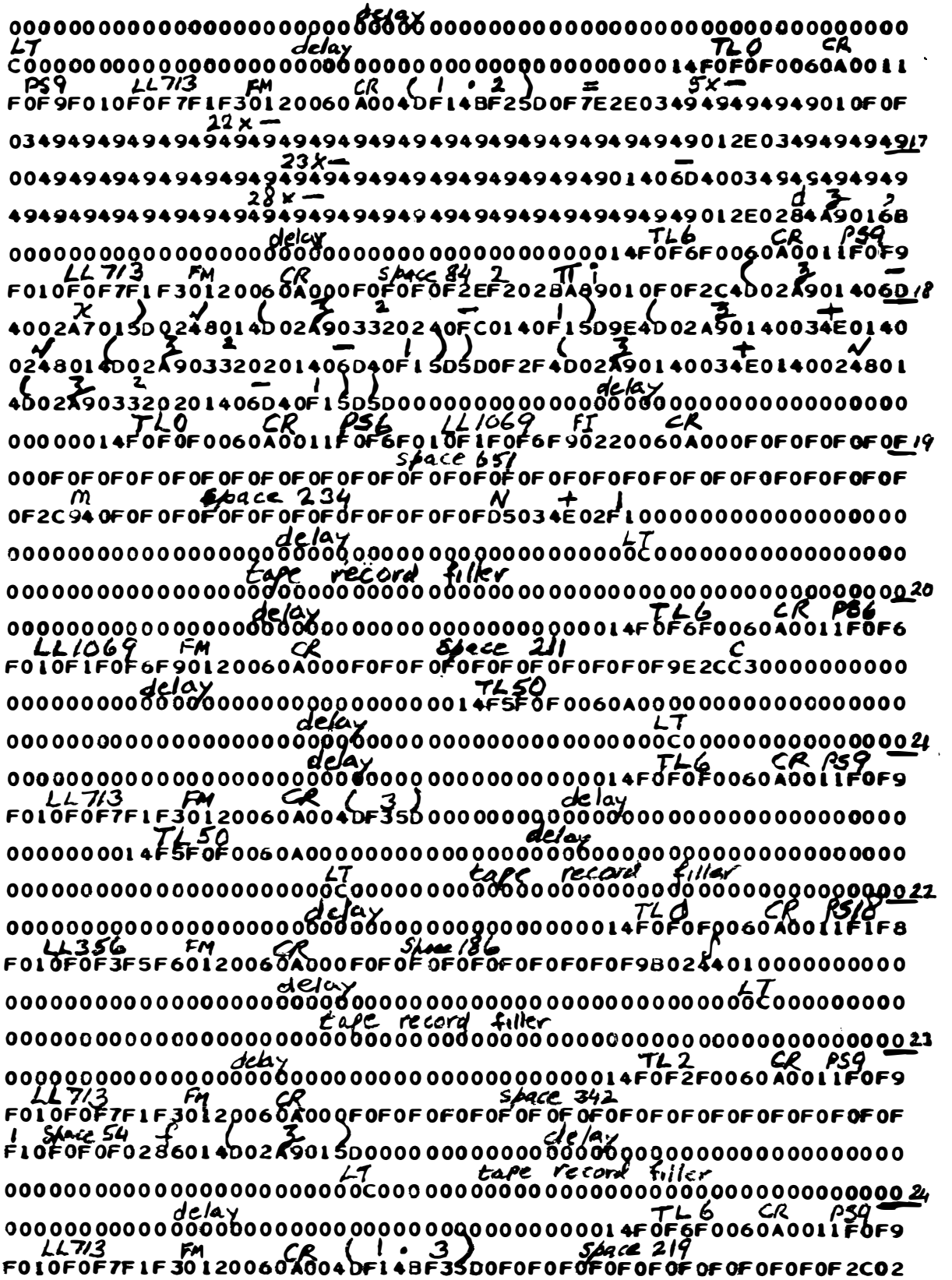
Mathematical Text : Equations

Output on Magnetic Tape - Continued



Mathematical Text : Equations

Output on Magnetic Tape - Continued



Mathematical Text : Equations

Output on Magnetic Tape - Continued

```
 (   (   x   )   =          5 x -                8 x -
860140D02A7015D2E7E2E03494949494949010F0F034949494949494949012C
  d  3   ,                    tape record filler
02B4A9016B000000000000000000000 000000000000000000000000000000000 25
               delay                              TL6      CR   PS9
000000000000000000000000000000 000000000000014F0F6F0060A0011F0F9
     LL 713      FM     CR                      space 327
F010F0F7F1F30120060A000F0F0F 0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F2C
 2  π  j              3        -       x
F202BA8901 0F0F9A02A901406D40 02A70100000000000000000000000000
               delay                      LT       tape record filler
000000000000000000000000000000 000000C0000000000000000000000000000 26
               delay                              TL6      CR   PS6
000000000000000000000000000000 000000000000014F0F6F0060A0011F0F6
   LL1069       FM     CR                  space 573
F010F1F0F6F90120060A000F0F0F 0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F
                                    C
0F0F0F0F0F0F0F0F0F0F0F0F9E9AC3000000 000000000000000000000000
            delay                   LT        CR    HA
00000000000000000000000000000000C000060A000B004040404040404040 27

END OF INPUT.   27 RECORDS READ
```

Mathematical Text : Equations

Final Galley

CSIR-FOR NIPR.
DISK:490 SCHOOLBOOK.
MODE:HYPHENLESS

(1)

(1.1) $$a_{n,m} = \frac{1}{\pi i}\int_C \frac{f^m(z)dz}{\sqrt{(z^2-1)}(z+\sqrt{(z^2-1)})^n},$$

(2)

(1.2) $$p_N^m(x) = \frac{1}{2\pi i}\int_C \frac{f^m(z)}{(z-x)\sqrt{(z^2-1)}}\left[\frac{L_{N+1}(x)}{(z+\sqrt{(z^2-1)})^m} - \frac{L_m(x)}{(z+\sqrt{(z^2-1)})^{N+1}}\right]dz,$$

(3)

(1.3) $$f(x) = \frac{1}{2\pi i}\int_C \frac{f(z)}{z-x}dz,$$

ERROR RETURNS (Version 5.2)

Codes for action taken by program :-

E : error count increased by 1 : program will continue with initialising, processing command strings and heading but will stop when this stage has been completed,

S : the program stops immediately,

C : the program reports the error, replaces the invalid text character by a marker and continues.

Error Number	Reason	Program Line Number of Test	Program	User Response
(1)	Too many marker codes	280	E	Increase dimension of NCD, NACD and value for NATCD
(3)	At least one error in marker codes	323	S	Correct the errors that have previously been reported
(4)	No command string given for this marker	398	C	Provide a command string or correct the code
(5)	Check words in program and text file differ	655	S	Check that correct tables are being used
(6)	Wrong delimiter for a command	751	E	Correct the command string
(7)	Mnemonic in command string not been allocated a hex code	670	E	Correct the mnemonic or add mnemonic to HXTAB with its hex code
(8)	Mnemonic coded in OPTAB has no hex code	831	E	Check OPTAB code. If correct then do as for (7)
(9)	Mnemonic has been allocated same code as an EBCDIC character	831	E	Check code in HXTAB

Error Number	Reason	Program Line Number of Test	Program Action	User Response
(10)	There is no character on Typesetter disk corresponding to the Leader character	881	E	Study disk layout. If report is correct substitute another character or modify control tape*. Otherwise check OPTAB code
(11)	Character to be used as Leader is not an EBCDIC character	881	E	If report is correct replace with an EBCDIC character. Otherwise check OPTAB code
(12)	Storage area for command strings is full	many	E	Increase dimension of LCODE and value of MXLCOD
(13)	Character in text is not an EBCDIC character	949	C	If report is correct, reenter with an EBCDIC character. Otherwise check OPTAB code
(14)	Character in text, although a valid EBCDIC character is not available on the Typesetter disk	949	C	Study disk layout. If is correct substitute another EBCDIC character. Otherwise check OPTAB code
(15)	Mnemonic lies outside the range allowed for in MNTAB	619/624	E	1) If this occurs during initialising, check mnemonics in HXTAB, CR,LG and Font 2) If this occurs during scanning of the marker table, check the command string
(16)	Symbol chosen for marking errors in text is not available on the Typesetter disk	1030	E	Choose another symbol

Error Number	Reason	Program Line Number of Test	Program Action	User Response
(17)	Non EBCDIC character in Imbedded text	805	E	Check the string
(18)	Command string over-flows input buffer	many	E	Correct the length of command string
(19)	Nested repetition within a command string	777	E	Correct the command string
(20)	Length of command string (possibly containing a repeated command) exceeds a tape record	932	E	Break up the command string
(21)	A repetition used in text is nested	363	C	Correct the text
(22)	Non-numeric repetition number	372/781/786	E	In a command string : Correct the command string
			C	In text : Correct the text
(23)	An invalid mnemonic (i.e. not two upper case alphabetic characters) has been used in HXTAB or for special commands or symbols in SR TABLES	644	E	Check the mnemonics used.

Error Number	Reason	Program Line Number of Test	Program Action	User Response
(24)	Errors in preliminary processing	341	S	Correct the errors
(25)	Imbedded text not available on disk	805	E	Study disk layout and HXTAB and OPTAB entries
(26)	Command not provided for in computed go to for galley heading printing parameters	997	E	Correct the command or modify the computed go to
(27)	Repeated command causes overflow of array for converted strings	725	E	Enlarge the LCODE array
(28)	Size of Input Buffer (CH) is not a multiple of the input record (ITBREC)	251	E	Correct INCHAR and the dimension of CH
(29)	Size of Output Buffer (KCH) is not a multiple of the output record (IMBREC)	255	E	Correct KOUCHS and the dimension of KCH

*By suitably modifying the control tape a specified ECBCDIC character can be made to produce any character on the typesetter's disk.

The computer output shows the Marker Table in hexadecimal codes. This can be checked for errors in the HXTAB and OPTAB tables or in the assignment of codes to the FONT array or LG. (Line 500)

1) The most difficult errors to understand are those of incorrect Leading and Point Size and the following method helps considerably to throw light on what has happened.

Mark all Carriage Returns and all specifications of Point Size or Leading. In the margin opposite each Carriage Return write down the Leading and any changed Point Size. In the tape record identify the successive lines of the Galley and write down the Point Size and the Leading measured from the Galley. A comparison of specified and actual results shows where additional Carriage Returns between text and change of Point Size or additional delays between Leadings are required.

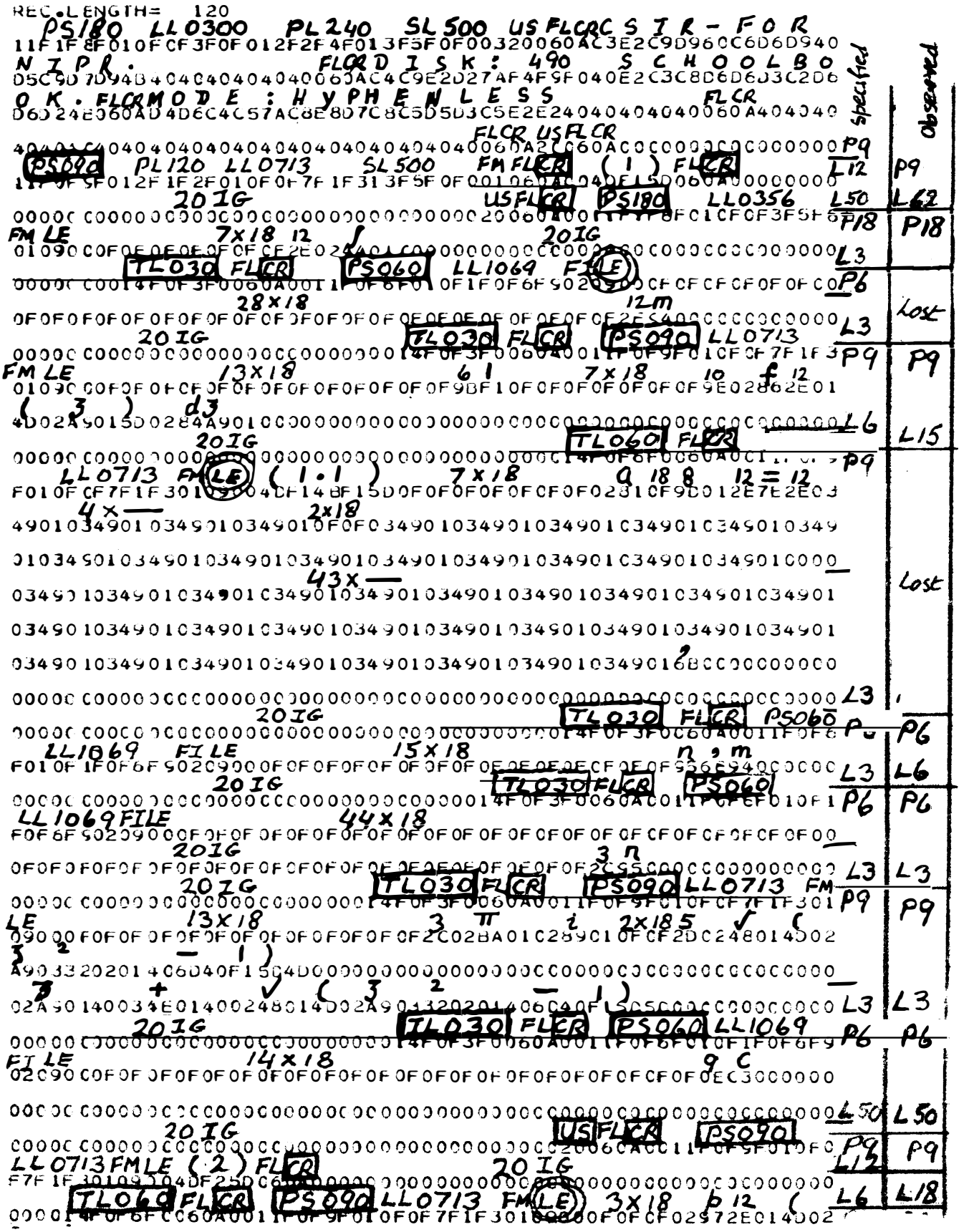
An example of this method is shown on the next page.

2) A second and very confusing error arises when the Typesetter receives a line longer than the current Line Length. This often arises when a change of Point Size is applied to a wrong part of the text, possibly through the omission of a Carriage Return.

When this occurs the machine prints, sometimes repeatedly, parts of preceding lines and it may overwrite characters on the right-hand margin.

Notes on Analysing Tape Output

Tape Output marked with Point Sizes and Leadings



Notes on Analysing Tape Output

Galley with Errors

CSIR-FOR NIPR.
DISK:490 SCHOOLBOOK.
MODE:HYPHENLESS

(1)

eintegral and

$$e_{-1} \qquad f_{n,m}(z)dz \qquad \int \qquad \frac{}{\pi i \; C \quad \sqrt{(z^2-1)}(z+\sqrt{(z^2-1)})^{n}}$$

(2)

$$p_N(x)$$

$$\frac{8}{a^8} \quad - \quad \frac{1}{i} \int \frac{f^n(z)}{(z-x)\sqrt{(z^2-1)}\;(z+\sqrt{(z^2-1)})}\,dz, \quad \left[\frac{}{(z+\sqrt{(z^2-1)})} \right. \quad L_N(x) \quad N+1 \quad \left. \right]^{L_N(x)} \quad N \quad N+1$$

(3)

$$\div \quad \frac{}{}\,dz, \qquad N \qquad \frac{1}{2\pi i}\int_C \frac{}{z-x} \qquad N+1 \qquad 1 \qquad 830 \qquad f(z)$$

Appendix 10

Character Widths

(South African Format)

	Times Roman All fonts	Mega-ron All fonts	School book	
			Med-ium & Italic	Bold
a	9	10	11	12
b	10	11	11	12
c	8	10	9	10
d	10	11	11	12
e	8	10	9	11
f	6	6	7	8
g	9	11	11	12
h	10	11	12	13
i	5	5	6	7
j	5	5	6	7
k	10	10	11	13
l	5	5	6	7
m	15	16	17	18
n	10	11	12	13
o	10	11	10	12
p	10	11	11	12
q	10	11	11	12
r	7	7	8	10
s	7	9	9	10
t	6	6	7	8
u	10	11	11	12
v	9	9	11	12
w	13	14	15	18
x	10	10	11	12
y	9	10	11	12
z	8	9	9	10

	Times Roman All fonts	Mega-ron All fonts	School book	
			Med-ium & Italic	Bold
A	14	13	15	16
B	12	13	14	15
C	13	14	14	15
D	15	14	15	17
E	12	12	14	15
F	12	11	13	14
G	14	15	15	16
H	15	14	16	17
I	7	6	8	9
J	9	10	11	13
K	14	13	15	17
L	12	11	13	14
M	18	16	18	18
N	14	14	16	17
O	14	15	15	16
P	12	12	13	15
Q	14	15	15	16
R	14	13	14	16
S	11	12	12	13
T	12	12	13	14
U	14	14	16	17
V	13	12	15	16
W	18	17	18	18
X	14	12	14	16
Y	13	12	14	16
Z	12	12	12	13

Information supplied by AM International and Graphic Arts (CSIR)

Character Widths
(South African Format)

	Times Roman All fonts	Mega-ron All fonts	School book Medium & Italic	School book Bold
Digits	9	10	10	12
Super- and subscripts	6	6	7	7
Fractions	15	14	15	15
,	5	5	5	6
.	5(4)	5	5	6
:	5	5	5	6
;	5(4)	5	5	6
‘	4(5)	4	5	6
’	4(5)	4	5	6
?	7	10	9	10
!	6	5	5	6
´	4	5	5	6
"	7	5	5	10
-	6	7	5	8
—	9	9	9	9
——	18	18	18	18
*	9	8	9	9
ß	9	10	10	12
£	9	10	10	12
/	9	7	10	12
%	15	15	14	16
#	9	10	10	12
(	6	7	6	6
)	6	7	6	6
´n	14	15	14	14
ê	8	10	8	8
ë	8	10	5	5
ï	5	5	14	16
&	14	13	6	6
[]	6	7		

	Times Roman Medium	Mega-ron Medium	School book
( )	4	4	4
Accents	0	0	0
| | '	4	4	4
ℓ	7	7	7
μ	10	11	11
@	17	17	-
*	9	18	9
•	14	10	14
·	5	10	5
□	18	18	18
△	20	18	20
▽	-	18	-
◇	-	18	-
+	14	18	14
x	14	18	14
÷	18	18	18
=	11	18	11
>	14	18	14
<	14	18	14
±	18	18	18
{}	-	6	-
°	9	10	9
†	9	10	9
¤	9	10	9
®	-	11	-
©	-	11	-
™	-	11	-

Intercepts for Times Roman, School book widths are estimates

Character Widths

(South African Format)

	Times Roman	Mega-ron	School book
α	11	11	11
β	12	11	12
γ	12	12	12
δ	10	9	10
Δ	14	13	14
ι	7	6	7_
λ	11	11	11
μ	11	11	11
υ	9	9	9
ω	13	12	13
Ω	14	14	14
π	13	12	13
σ	11	10	11
Σ	12	12	12
θ	11	9	11
$\surd$	13	16	13_
$\int$	11	10	11
$\mathring{A}$	14	13	14
Supershift	-	3	-
$\equiv$	14	14	14
$\not\leq$	-	14	-
$\rightarrow$	-	15	-
Leader .	3	7	3
Leader ..	9	9	9
Leader -	9	9	9
Leader _	9	9	9

Note : Times Roman and Schoolbook have been estimated

Interword spacing

The minimum size of a space is 3 units but this is increased to 6 units for a short line (left or right adjusted or centred).

```
L.007 C     HALL NIPR NI50(5.2) - INSERTING TYPSETTING CODES IN TEXT
L.008 C     ------------------------------------------------------
L.009 C     (5. THIS VERSION HAS THE IMBEDDED TEXT,REPEAT AND LT FACILITIES)
L.010 C     (5.2 ELIMINATES UNNECESSARY FONT CHANGES)
L.011 C     FOR USE WITH ADDRESSOGRAPH-MULTIGRAPH 747 & 748 PHOTOTYPESETTING MACHINES
L.012 C     THE COMMANDS FOR THE TYPESETTING COMPUTER ARE HEXADECIMAL NUMBERS WHICH
L.013 C     HAVE NO EBCDIC EQUIVALENTS.IN ORDER TO HANDLE THESE COMMANDS ON
L.014 C     ON A TYPEWRITER EACH COMMAND IS REPRESENTED BY TWO UPPER CASE LETTERS
L.015 C     PRECEDED BY A COMMAND DELIMITER,SAY $.THE SR. TABLES TRANSLATES LETTERS
L.016 C     INTO HEXADECIMAL PRINTER COMMANDS USING THE ARRAY MNTAB.
L.017 C     TYPESETTING CHARACTERS NOT ON TELETYPE KEYBOARDS ARE ALSO GIVEN UNUSED
L.018 C     HEX CODES AND ENTERED AS A DELIMITER AND TWO UPPER CASE LETTERS.
L.019 C     THE END OF A COMMAND STRING IS SIGNALLED BY TWO COMMAND DELIMITERS EG.$$
L.020 C     THE TABLES OF COMMANDS AND ADDITIONAL CHARACTERS AND THEIR HEX CODES
L.021 C     AS WELL AS THE CHECK WORD,ARE IN THE SR. TABLES
L.022 C
L.023 C     AS  COMMANDS REPEAT REGULARLY THROUGHOUT TEXT IN GROUPS,
L.024 C     TYPING CAN BE SIMPLIFIED BY REPLACING EACH GROUP BY A CODE
L.025 C     CHARACTER PRECEDED BY ,SAY @. THE CORRESPONDENCE BETWEEN THE @ CODES AND
L.026 C     THE PRINTER COMMAND GROUPS IS GIVEN BY A DICTIONARY AT THE START OF
L.027 C     EACH FILE OF TEXT .THUS THE @ CODES CAN  BE VARIED FOR DIFFERENT
L.028 C     BODIES OF TEXT.THE CODES USED ARE PRINTED WITH EACH RUN.
L.029 C
L.030 C     REPEAT FACILITY
L.031 C     REPEATED COMMANDS WITHIN A COMMAND STRING CAN BE CONDENSED BY USING A
L.032 C     REPEAT DELIMITER,SAY,%, AND WRITING THE COMMAND AS:- %0503$MV  WHERE
L.033 C     05 IS THE NO. OF REPETITIONS(2 DIGITS) AND 03 IS THE LENGTH OF THE STRING
L.034 C     TO BE REPEATED(2 DIGITS).
L.035 C     TO REPEAT THE WHOLE COMMAND STRING ASSOCIATED WITH AN @ CODE
L.036 C     PRECEDE @ WITH THE REPEAT DELIMITER AND THE NO. OF REPETITIONS
L.037 C     (2 DIGITS),EG.%05@W WILL INSERT W'S COMMAND STRING 5 TIMES.
L.038 C     THE TWO KINDS OF REPEATS MAY BE USED TOGETHER,BUT NEITHER MAY BE
L.039 C     NESTED.
L.040 C
L.041 C     IMBEDDED TEXT
L.042 C     TEXT CAN BE INCLUDED IN COMMAND STRINGS IF PRECEDED BY A
L.043 C     TEXT DELIMITER,SAY &,WHICH INCLUDES FOLLOWING CHARACTERS UP TO
L.044 C     THE NEXT COMMAND OR REPEAT DELIMITER
L.045 C
L.046 C
```

```
L.047 C      ENTERING TEXT OR MAKING CORRECTIONS
L.048 C     AN IGNORE SYMBOL IN TEXT OR COMMANDS CAN BE USED FOR FILLING UNWANTED
L.049 C     SPACES AND THE LT COMMAND FOR SKIPPING BLANKS FILLING OUT AN INPUT LINE.
L.050 C     BLANKS FOLLOWING THE C/R,LINE END AND HALT COMMANDS ARE ALSO IGNORED
L.051 C
L.052 C     STORE RULE AND LEADER COMMANDS
L.053 C     THE RULE OR LEADER CHARACTERS MAY BE FROM THE KEYBOARD OR RETRIEVED FROM
L.054 C     THE TYPESETTER DISK BY MEANS OF A COMMAND DELIMITER AND MNEMONIC
L.055 C
L.056 C     DEFAULT DELIMITERS AND MARKERS
L.057 C     COMMAND DELIMITER                $ (Z5B)
L.058 C     @ CODE MARKER                    @ (Z7C)
L.059 C     IGNORE                           ¢ (Z4A)
L.060 C     IMBEDDED TEXT                    & (Z50)
L.061 C     REPEAT DELIMITER                 # (Z7B)
L.062 C     INVALID @ CODE IN TEXT           (Z51)
L.063 C     INVALID TEXT CHARACTER           (Z52)
L.064 C     FOR THE LAST TWO,WHICH ARE NOT EBCDIC, SEE PERS 305,APPENDIX 8,P(ii)
L.065 C
L.066 C     THE USER MAY OVERRIDE THE SYMBOLS USED FOR COMMAND DELIMITER,@ MARKER,
L.067 C     IGNORE,TEXT AND REPEAT DELIMITERS AND THE SYMBOLS MARKING
L.068 C     INVALID @ CODES OR CHARACTERS IN THE TEXT BY INSERTING THE
L.069 C     REQUIRED CHARACTER IN THE APPROPRIATE POSITION IN THE SECOND LINE
L.070 C     OF THE FILE.
L.071 C     (SEE FILE FORMAT BELOW)
L.072 C     ANY FACILITY CAN BE MADE INOPERATIVE BY INSERTING A ZERO
L.073 C     IN THE APPROPRIATE POSITION IN THE SECOND LINE
L.074 C     ONLY THE,IGNORE CHARACTER AND THE REPEAT DELIMITER
L.075 C     ARE FORBIDDEN CHARACTERS FOR THE TEXT.THE @ CODE MARKER MAY BE USED IF
L.076 C     FOLLOWED BY A BLANK
L.077 C
L.078 C     THE FIRST LINE(80 CHARS) OF THE INPUT FILE IS WRITTEN AS A HEADING ON
L.079 C     THE GALLEY IN LINES OF 20 CHARACTERS AND CAN BE USED FOR IDENTIFICATION OR
L.080 C     INSTRUCTIONS FOR THE TYPESETTER OPERATOR
L.081 C     THE FIRST FOUR CHARACTERS OF THIS LINE ARE COMPARED
L.082 C     WITH THE CHECK WORD IN THE PROGRAM AS A CHECK THAT THE CORRECT VERSION
L.083 C     OF THE PROGRAM IS BEING USED.
L.084 C
L.085 C     FOR FURTHER DETAILS SEE NIPR SPECIAL REPORTS PERS 305,PERS 317
L.086 C
```

```
L.087 C     FILE FORMAT
L.088 C     -----------
L.089 C     LINE 1 CHECK WORD AND GALLEY IDENTIFICATION (20A4)
L.090 C          2 COMMAND DELIMITER,@ CODE,IGNORE,TEXT AND REPEAT DELIMITERS
L.091 C            AND ERROR MARKERS FOR INVALID @ CODES AND TEXT,SEPARATED BY SPACES
L.092 C     OR ANY OTHER SEPARATOR.(7(A1,1X))
L.093 C            IF LEFT BLANK,DEFAULTS WILL BE USED,IF=0,THEN INOPERATIVE
L.094 C          3 NO.OF @ CODES,N  (I2)
L.095 C          4 @ CODE(A1);2 SPACES;COMMANDS(76A1),END WITH 2 COMMAND DELIMITERS
L.096 C     (UP TO LINE N+4)
L.097 C     N+5 SESSION IDENTIFIER
L.098 C     N+6 ETC., TEXT
L.099 C
L.100 C     IF THERE ARE ERRORS IN THE PRELIMINARY STAGES THE PROGRAM
L.101 C     WILL COMPLETE THE PROCESSING OF THE PRELIMINARY STAGES REPORTING
L.102 C     ALL ERRORS BUT PROCEEDING NO FURTHER.
L.103 C     THE NO.OF LINES OF TEXT IS UNLIMITED,BUT A HALT COMMAND SHOULD
L.104 C     BE PLACED AFTER BLOCKS OF 400 LINES TO PERMIT CHANGING THE PAPER
L.105 C     IN THE TYPESETTER
L.106 C
L.107 C     SYSTEM REQUIREMENTS: RJE         80 BYTE RECORDS(TELETYPE LINES)
L.108 C                          TYPESETTER 120  "     "   (MAGTAPE RECORDS)
L.109 C     (A COMMAND STRING IS LIMITED TO ONE TAPE RECORD)
L.110 C
L.111 C     IF CHANGES ARE REQUIRED
L.112 C     -----------------------
L.113 C     TO MODIFY SIZE,CHANGE DIMENSIONS AND STRUCTURAL PARAMETERS
L.114 C     BUFFERS:ARRAYS:CH KCH
L.115 C               PARAMETERS:INCHAR KOUCHS
L.116 C     RECORD LENGTH(BYTES):TAPE;IMBREC,TTY;ITBREC. WORD LENGTH: IWD
L.117 C     ARRAYS FOR @ CODES AND COMMAND STRINGS: LINCD LCODE NCD NACD
L.118 C                    PARAMETERS:INBF MXLCOD NATCD
L.119 C
L.120 C     IF A DIFFERENT TYPESETTER DISK IS USED,MODIFY HXTAB AND OPTAB IN TABLES
L.121 C     AND POSSIBLY OPTAB GOTO'S
L.122 C
L.123 C     IF A DIFFERENT MACHINE IS USED,E.G. 748,NOT ONLY MUST THE CHANGES
L.124 C     FOR A DIFFERENT DISK BE MADE,BUT THE NO. OF FONTS MAY DIFFER
L.125 C     FUNCTIONS MAY ALSO DIFFER,REQUIRING CHANGES IN HXTAB
L.126 C     OPTAB AND FLAGS,AND CHANGES IN THE ACTIONS FOLLOWING THE COMPUTED
L.127 C     GO TO'S. THE SR.S CHFON,MARK AND LEADER MAY ALSO NEED MODIFYING
L.128 C
```

```
L.129 C     SUBROUTINES-INTERNAL: TABLES-HXTAB,OPTAB,CHECK WORD
L.130 C                              ENTRIES-CHECK,OPCH,OPMN
L.131 C                           CONV-PROCESSES MARKER TABLE,OPTAB GOTO
L.132 C                           KTEST-PROCESSES TEXT,OPTAB GOTO
L.133 C                           HEAD-PROCESSES HEADING PARAMETERS,OPTAB GOTO
L.134 C                           INSFON-ERROR MARKERS,OPTAB GOTO
L.135 C                           BLOCK DATA-FLAGS,DEFAULTS,HEADING PARAMETERS
L.136 C                           WROUT-OUTPUT A BLOCK OF TEXT
L.137 C                           WRIN-INPUT A BLOCK OF TEXT
L.138 C                           IKTEST-TEST FOR END OF INPUT BLOCK
L.139 C                           CHFON-INSERT FLAGS FOR FONT CHANGE AND RESTORE
L.140 C                           MARK-INSERT FLAGS FOR FONT STORE ,LT AND C/R
L.141 C                           FOLLOW-PICK UP DIGITS WITH COMMAND
L.142 C                           LEADER-INSERT FLAGS FOR LEADER OR RULE CHARACTERS
L.143 C                           LINE-CALCULATE LINE NUMBER
L.144 C                           QNTG-CONVERT LOGICAL*1 TO INTEGER*2
L.145 C                           LIMS-CALCULATING LIMITS
L.146 C                           QCALC-CALCULATING ADDRESSES FROM MNEMONICS
L.147 C                           NUM-CONVERT TWO NUMERIC CHARACTERS TO AN INTEGER
L.148 C                           DELIM-TESTING FOR A DELIMITER OR MARKER
L.149 C                           DEFAL-SUPPLYING DEFAULT DELIMITERS OR MARKERS
L.150 C                           QCOMOP-COMPARING OPTAB CODES OF TWO CHARACTERS
L.151 C                           ERREP-REPORTING ERRORS IN TEXT
L.152 C
L.153 C     OPERATING SPECIFICATIONS
L.154 C     ----------------------
L.155       implicit logical*1 (l),integer*2 (q)
L.156       generic
L.157 C     INPUT BUFFER: 500 LINES OF 80 BYTES
L.158 C     OUTPUT BUFFER 400 120-BYTE RECORDS
L.159 C     BUFFER SIZES MUST BE MULTIPLES OF CORRESPONDING INPUT OR OUTPUT RECORDS
L.160       logical*1 ch(40000),kch(48000)
L.161 C     ARRAYS FOR  60 0 CODES,76 BYTE INPUT BUFFER AND 800 BYTE BUFFER FOR
L.162 C     CONVERTED COMMAND STRINGS
L.163 C      NACD DIMENSION ONE MORE THAN NCD
L.164       integer ncd(60),nacd(61)
L.165       logical*1 lincd(80),lcode(800)
L.166 C     END OF OPERATING SPECIFICATIONS
L.167 C     ------------------------------
L.168 C
```

```
L.169         integer ident(20)
L.170 C       ARRAY FOR RECORDING @ CODES USED
L.171         INTEGER*2 atab(256)/256*0/
L.172          integer  ich/'    '/
L.173 C       DELIMITER AND FLAG ARRAYS
L.174         equivalence (icm,lpcm),(iat,lpat),(imb,lpmb),(irp,lprp),(ign,lpgn)
L.175 C       FIRST WORD OF INPUT FILE CHECKS PROGRAM VERSION
L.176         equivalence (ch(1),iflck)
L.177 C       FOR HANDLING SINGLE CHARACTERS
L.178         equivalence (ich,lch),(iatcd,lat),(icem,lcem),(item,ltem)
L.179         common ioup,iout,in,iwd
L.180         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L.181         common /logs/lb(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.182 C       NOTE:A CONDENSED FORM OF /TBLE/ IS USED THE SUBROUTINES
L.183 C       ICM TO IAT=IDEL,IEE TO IDE=IELG
L.184         common /tble/icm,imb,irp,ign,ibl,iat,ndk,iff,irf,ifs,idf(5),nflg,
L.185        1nhd,nldl,glg,gonld,grf
L.186         common /dims/mxlcod,inbf,inchar,kouchs
L.187         common /points/k,kf,ik,il,ic,ih
L.188 C
L.189 C       IN AND OUT UNITS
L.190 C       ---------------
L.191 C       PRINT
L.192         ioup=6
L.193 C       EXTERNAL STORAGE
L.194         in=11
L.195         iout=21
L.196 C        RECORD LENGTHS
L.197 C       TAPE RECORD(BYTES)(MB)
L.198         imbrec=120
L.199 C       CORRESPONDING FORMAT IN WROUT IS (120A1)
L.200 C       TERMINAL RECORD(BYTES)(TB)
L.201         itbrec=80
L.202 C       CORRESPONDING FORMAT IN WRIN IS (80A1)
L.203 C
L.204 C        OPERATING PARAMETERS
L.205 C       ------------------
L.206 C       TEXT BUFFER SIZES (BYTES)
L.207         inchar=40000
L.208         kouchs=48000
```

```
L.209 C       0 CODES TABLE DIMENSIONS(BYTES)
L.210         natcd=60
L.211 C       COMMAND STRING INPUT BUFFER(BYTES)
L.212         inbf=76
L.213 C       CORRESPONDING FORMAT IN ST.2 IS (76A1)
L.214 C        ARRAY FOR COMMAND STRINGS(BYTES)
L.215         mxlcod=800
L.216 C       WORD LENGTH(BYTES)
L.217         iwd=4
L.218 C       LENGTH OF ARRAYS FOR DELIMITERS AND MARKERS(IDEL) AND DEFAULTS(LDEL)
L.219         nldl=6
L.220 C       LENGTH OF IDEL ARRAY FOR DELIMITERS-EXCLUDING 0 IN IDEL(6)
L.221 C        AS THIS IS NOT USED IN SR.DELIM
L.222         ndk=5
L.223 C       LENGTH OF IFLG ARRAY FOR FLAGS
L.224         nflg=8
L.225 C       LENGTH OF LH ARRAY FOR PRINT PARAMETERS FOR GALLEY HEADING
L.226         nhd=31
L.227 C       DEPENDENT  PARAMETERS
L.228         inlins=klim(inchar,itbrec)
L.229 C
L.230 C        END OF OPERATING PARAMETERS
L.231 C       ------------------
L.232 C
L.233 C        INITIALISATION
L.234 C        --------------
L.235         do 5 i=1,mxlcod
L.236       5 lcode(i)=lbl
L.237         kblk=0
L.238         mrecou=0
L.239         lnflag=0
L.240         ierfl=0
L.241         k=0
L.242         ik=0
L.243         ic=0
L.244         ih=0
L.245         nacd(1)=0
L.246         irt=1
L.247         jrt=1
```

```
L.248         write(ioup,6)
L.249       6 format('1 N.I.P.R. TYPESETTING PROGRAM (5.2)'//)
L.250 C       CHECK BUFFERS AND INPUT/OUTPUT RECORDS ARE CONSISTENT
L.251         if(mod(inchar,itbrec) .eq. 0) go to 130
L.252         write(ioup,120) inchar,itbrec
L.253     120 format(/' (28) INCHAR(=',I7,'). NO MULTIPLE OF ITBREC(=',I5,')'/)
L.254         ierfl=ierfl+1
L.255     130 if(mod(kouchs,imbrec) .eq. 0) go to 150
L.256         write(ioup,140) kouchs,imbrec
L.257     140 format(/' (29) KOUCHS(=',I7,'). NO MULTIPLE OF IMBREC(=',I5,')'/)
L.258         ierfl=ierfl+1
L.259     150 call tables
L.260 C
L.261 C       GALLEY HEADING AND TABLE OF @ CODES,READ FROM FILE AHEAD OF TEXT
L.262 C       ------------------------------------------------------------
L.263 C       GALLEY HEADING
L.264         read(in,7) (ch(i),i=1,itbrec)
L.265       7 format(80A1)
L.266 C       CHECK VERSION OF PROGRAM
L.267         call check(iflck)
L.268 C       DELIMITERS COMMANDS,CODES,IGNORE,TEXT,REPEAT AND ERROR MARKERS(CODES,TEXT)
L.269         read(in,8) icm,iat,ign,imb,irp,icem,item
L.270       8 format(7(a1,1x))
L.271 C       REPLACE DELIMITERS OR MARKERS BY DEFAULTS
L.272         call defal
L.273 C       MUST DEFAULT ERROR MARKERS BE OVERRIDDEN?
L.274         if(icem .ne. ibl) call insfon(latr,lcem)
L.275         if(item .ne. ibl) call insfon(ltxr,ltem)
L.276 C       NO.OF @ CODES
L.277      14 read(in,1)n
L.278       1 format(i2)
L.279 C       CHECK NO. OF @ CODES
L.280         if(n .le. natcd) go to 10
L.281         write(ioup,9) iat, n,natcd
L.282       9 format(/' (1) TOO MANY ',A1,' CODES:ACTUAL ',I4,';ALLOWED ',I4/)
L.283         n=natcd
L.284         ierfl=ierfl+1
L.285 C       @ CODES,LENGTHS AND TYPESETTING CODES READ FROM BEGINNING OF FILE
L.286      10 write(ioup,3) n
L.287       3 format(/' CODES USED'/' N=',I3/' CD:LEN :FLAGS :TYPESETTING CODES
L.288        1IN HEX')
```

```
L.289         do 11 i=1,n
L.290         read(in,2) iatcd,(lincd(j),j=1,inbf)
L.291       2 format(A1,2X,77A1)
L.292 C
L.293 C        CONVERT MNEMONICS TO HEX CODES
L.294 C        --------------------------
L.295      13  call conv(iatcd,lincd,lcode)
L.296 C       RECORD CODE LENGTH AND CUMULATIVE LENGTH
L.297 C       --------------------------------
L.298         ncd(i)=nc
L.299         nacd(i+1)=nacd(i)+nc
L.300 C       NUMERICAL VALUE OF CHARACTER GIVES ARRAY ADDRESS
L.301         atab(qntg(lat))=i
L.302 C       PRINT TYPESETTING CODES
L.303         is=nacd(i)+1
L.304         ie=nacd(i+1)
L.305         write(ioup,4) iatcd,ncd(i),(lcode(j),j=is,ie)
L.306       4 format(1X,A2,2X,I2,1X,3Z2,2X,53Z2/(8X,56Z2))
L.307      11 continue
L.308         write(ioup,12)
L.309      12 format(/' DELIMITERS AND MARKERS FOR THIS RUN')
L.310         write(ioup,17) icm,lpcm,iat,lpat,ign,lpgn,imb,lpmb,irp,lprp,
L.311        1latr(2),latr(2),ltxr(2),ltxr(2)
L.312      17 format(' COMMAND DELIMITER',7X,A1,3X,Z2/' TYPESETTER CODE MARKER',
L.313        12X,A1,3X,Z2/' IGNORE',18X,A1,3X,Z2/' TEXT DELIMITER',10X,A1,3X,Z2/
L.314        2' REPEAT DELIMITER',8X,A1,3X,Z2/' ERROR MARKERS:CODES',5X,A1,3X,Z2
L.315        3/15X,'TEXT',6X,A1,3X,Z2//)
L.316         write(ioup,16)ic,mxlcod
L.317      16 format(1X,I5,' USED OUT OF ',I5,' AVAILABLE IN LCODE ARRAY'/)
L.318 C        TEXT IDENTIFIER(NOT WRITTEN TO TAPE)
L.319          read(in,20) ident
L.320      20  format(20a4)
L.321          write(ioup,18)ident
L.322      18  format(/' BATCH IDENTIFIER:',1X,20A4//)
L.323         if(ierfl .eq. 0) go to 35
L.324         write(ioup,32) ierfl,iat
L.325      32 format(/' (3) ',i5,' ERRORS IN ',A1,' CODES'/)
L.326 C
L.327 C       WRITE TEXT FROM DISK TO TAPE REPLACING @ CODES BY COMMAND STRINGS
L.328 C       ------------------------------------------------------------
```

```
L.329 C      WRITE HEADING FOR GALLEY
L.330 C      -------------------
L.331    35  call head(kch,0)
L.332        do 37 i=1,itbrec
L.333        ik=ik+1
L.334        call ktest(kch,ch)
L.335        if(mod(ik,20) .eq. 0) call head(kch,27)
L.336    37  continue
L.337        call head(kch,25)
L.338 C
L.339 C      TEST FOR ERRORS IN PRELIMINARY STAGES
L.340 C      -------------------------------------
L.341        if(ierfl .eq. 0) go to 39
L.342        write(ioup,38) ierfl
L.343    38  format(' (24) ',I4,' ERRORS IN PRELIMINARY STAGES:STOP'/)
L.344        stop
L.345 C
L.346 C      INPUT TEXT
L.347 C      ----------
L.348    39  call wrin(ch)
L.349 C
L.350 C      TESTING TEXT FOR CODES
L.351 C      ----------------------
L.352 C      CENTRAL LOOP
L.353    57  call iktest(ch,lch,kch)
L.354    58  if(ich .eq. iat) go to 70
L.355        if(ich .eq. ign) go to 57
L.356        if(ich .eq. irp) go to 62
L.357 C      CHARACTER IS TEXT
L.358    60  call ktest(kch,ch)
L.359        go to 57
L.360 C
L.361 C      ACTION ON ENCOUNTERING REPEAT DELIMITER
L.362 C      ---------------------------------------
L.363    62  GO TO(68,63),JRT
L.364    63  write(ioup,66)
L.365    66  format(/' (21) NESTED REPEAT')
L.366        call errep(ch,kch,latr)
L.367        ik=ik+2
L.368        go to 57
```

```
L.369 C      COMMAND MAY OVERFLOW INPUT BLOCK
L.370    68  call iktest(ch,11,kch)
L.371        call iktest(ch,12,kch)
L.372        call num(irt,11,12,*57)
L.373 C      PREVENT NESTING
L.374        jrt=2
L.375        go to 57
L.376 C
L.377 C      ACTION ON ENCOUNTERING 0
L.378 C      ----------------------
L.379 C      IDENTIFY THE CODE
L.380    70  call iktest(ch,lch,kch)
L.381        if(ich .ne. ibl) go to 85
L.382 C      0 FOLLOWED BY BLANK,TRANSFER BOTH
L.383        ik=ik-1
L.384        if(k+1 .gt. kouchs) go to 60
L.385        kch(k)=ch(ik)
L.386        kch(k+1)=ch(ik+1)
L.387        k=k+1
L.388        ik=ik+1
L.389        go to 57
L.390 C      IF 0 FOLLOWED BY IGNORE,SKIP IGNORE SYMBOLS
L.391    85  if(ich .ne. ign) go to 86
L.392        call iktest(ch,lch,kch)
L.393        go to 85
L.394 C
L.395 C      GET 0 CODE,RIGHT-ADJUSTING TCH
L.396 C      ------------------
L.397    86  ia=atab(qntg(lch))
L.398        if(ia .ne. 0) go to 95
L.399 C      INVALID CODE
L.400        write(ioun,90) ich
L.401    90  format(/' (4) ',A1,' IS NOT IN MARKER TABLE')
L.402        call errep(ch,kch,latr)
L.403        go to 57
L.404 C
L.405 C      INSERT TYPESETTING CODE
L.406 C      ----------------------
```

```
L.407 C      TEST FOR C/R,LE,HA,LT,FONT STORE AND FONT RESTORE FLAGS
L.408    95  jcr=1
L.409        jrf=1
L.410        jfs=1
L.411        jrt2=1
L.412        jrt3=1
L.413        jrf2=1
L.414        is=nacd(ia)+1
L.415 C      TEST IF CODE HAS A MARKER
L.416        lch=lcode(is)
L.417        if(ich .eq. iff) jrf=2
L.418        lch=lcode(is+1)
L.419        if(ich .eq. iff) jfs=2
L.420        lch=lcode(is+2)
L.421        if(ich .eq. iff) jcr=2
L.422 C      LOCATE THE COMMAND STRING
L.423        is=is+3
L.424        ie=nacd(ia+1)
L.425        kln=ncd(ia)-3
L.426 C      WILL ALL REPETITIONS FIT BUFFER OR RECORD?
L.427        ksmt=kln*irt+k
L.428        ktend=(k/imbrec+1)*imbrec
L.429        if(ksmt .le. kouchs) jrt2=2
L.430        if(ksmt .le. ktend) jrt2=3
L.431        do 107 j=1,irt
L.432        ksm=kln+k
L.433        go to (96,98,100),jrt2
L.434 C      TEST IF THIS REPETITION WILL OVERRUN OUTPUT BUFFER
L.435    96  if(ksm .le. kouchs) go to 98
L.436        kdif=kouchs-k
L.437        do 97 i=1,kdif
L.438        k=k+1
L.439        kch(k)=lg
L.440    97  continue
L.441        call wrout(kch)
L.442        k=0
L.443        go to 100
L.444 C      TEST IF THIS REPETITION WILL OVERRUN TAPE RECORD
L.445 C      END OF NEXT RECORD
L.446    98  if(ktend .ge. ksm)go to 100
L.447        kdif=ktend-k
```

```
L.448         do 99 i=1,kdif
L.449         k=k+1
L.450         kch(k)=lg
L.451    99   continue
L.452         ktend=ktend+imbrec
L.453   100   go to(108,113),jrt3
L.454 C       SECOND AND SUBSEQUENT REPETITIONS (TRACE REMAINS WHEN UROUT IS ACTIVATED)
L.455   113   do 114 i=k1,k2
L.456         k=k+1
L.457         kch(k)=kch(i)
L.458   114   continue
L.459         go to 107
L.460 C       FIRST TIME ROUND
L.461   108   k1=k+1
L.462         do 105 i=is,ie
L.463         k=k+1
L.464         kch(k)=lcode(i)
L.465         lch=lcode(i)
L.466         go to(103,102),jrf
L.467 C       TEST FOR RESTORE FONT
L.468   102   if(ich .ne. irf) go to 103
L.469 C       IS REQUIRED FONT SAME AS CURRENT FONT?
L.470         if(qntg(lcode(i-2)) .ne. qntg(lfon)) go to 101
L.471 C       ELIMINATE CHANGE OF FONT
L.472         k=k-2
L.473         kch(k)=kch(k+1)
L.474         kch(k+1)=lg
L.475         kch(k+2)=lg
L.476         go to 105
L.477 C       REPLACE RF BY FONT TO BE RESTORED
L.478   101   kch(k)=lfon
L.479   103   go to(105,104),jfs
L.480 C       TEST FOR STORE FONT FLAG,DISCARD FLAG
L.481   104   if(ich .ne. ifs) go to 105
L.482         lfon=kch(k-1)
L.483         k=k-1
L.484    105  continue
L.485         if(irt .eq. 1) go to 116
L.486 C       ELIMINATE UNNECESSARY FONT CHANGES BETWEEN REPETITIONS
L.487         if(qntg(lcode(ie)) .ne. qrf) go to 106
L.488         jrf2=qcomop(lcode(is+1),lcode(ie-1))
L.489         go to(106,110),jrf2
```

```
L.490 C     ELIMINATE RESTORE FONT,FONT CHANGE BETWEEN REPETITIONS
L.491  110  k=k-1
L.492       k1=k1+1
L.493  106  k2=k
L.494       kln=k2-k1+1
L.495       jrt3=2
L.496  107  continue
L.497       go to(116,112),jrf2
L.498 C     ADD FINAL RESTORE FONT IF IT HAS BEEN REMOVED
L.499  112  k=k+1
L.500       kch(k)=lfon
L.501  116  irt=1
L.502       jrt=1
L.503 C     SKIP BLANKS AFTER C/R,LE,HA OR LT
L.504       go to (57,115),jcr
L.505  115  lch=ch(ik+1)
L.506       if((ich .ne. ibl) .and. (ich .ne. ign)) go to 57
L.507 C     SKIP ONLY TO END OF INPUT LINE
L.508       if(mod(ik,itbrec) .eq. 0) go to 57
L.509       ik=ik+1
L.510       go to 115
L.511       end
L.512 C     ---------------------
L.513 C
L.514       BLOCK DATA
L.515 C     ----------
L.516       implicit logical*1 (l),integer*2 (q)
L.517       common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.518       common /tble/idel(6),ndk,iflg(8),nflg,nhd,nldl,qlg,qopld,qrf
L.519 C     FLAGS:MARKER(EE),RESTORE FONT(RF),FONT STORE(FS),LEADER,FONTS AND IG
L.520 C     FONT(4)=IG=ANY FONT
L.521       data iflg/'EE  ','RF  ','FS  ','LD  ','FM  ','FI  ','FB  ','IG  '/
L.522 C     DELIMITER ARRAY
L.523       data idel/6*'    '/
L.524 C     DEFAULT ERROR MARKERS FOR CODES AND TEXT
L.525       data latr/Z03,Z51,Z90/,ltxr/Z02,Z52,Z90/
L.526 C     DEFAULTS FOR COMMAND,IMBEDDED TEXT,REPEAT DELIMITERS;IGNORE,0 CODE MARKERS
L.527 C     LCM=$,LMB=&,LRP=#,LGN= ,LBL=' ',LATC=0
L.528       data ldel/Z5B,Z50,Z7B,Z4A,Z40,Z7C/
L.529       data lbl/' '/,lfon/Z01/
```

```
L.530 C      PRINTING PARAMETERS FOR GALLEY HEADING
L.531        data lh/'P','S','1','8','0','L','L','0','3','0','0','P','L','2',
L.532       1'4','0','S','L','5','0','0','E','B','C','I','U','S','E','L',
L.533       2'C','P',' '/
L.534        end
L.535 C      ----------------
L.536 C
L.537        SUBROUTINE TABLES
L.538 C      CODES AND TABLES FOR A SPECIFIC DISK
L.539 C        ENTRIES:CHECK-CHECK THAT CORRECT VERSION OF PROGRAM IS BEING USED
L.540 C                OPMN-GET OPTAB CODE FOR A MNEMONIC
L.541 C                OPCH-GET OPTAB CODE FOR A CHARACTER
L.542 C
L.543        implicit logical*1 (l),integer*2 (q)
L.544        equivalence (iflg(4),lpld)
L.545 C      DELIMITERS
L.546 C      FOR HANDLING CHARACTERS
L.547        logical*1 lqcode(2),lhex(2),licode(2),lmn(2)
L.548        equivalence (qhex,lhex(1)),(qcode,lqcode(1)),(icode,licode(1))
L.549        common ioup,iout,in,iwd
L.550        common /counts/inflag,imbrec,ithrec,kblk,inlins,ierfl,mrecou,nc
L.551        common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.552        common /tble/id(4),ibl,iat,ndk,iflg(8),nflg,nhd,nldl,qlg,qopld,qrf
L.553 C
L.554 C      ------------------------------
L.555 C      THE FOLLOWING ARE SPECIFIC TO THE DISKS 489,490 AT THE CSIR
L.556 C      AND THE NLPR CONTROL TAPE (VERSION 5)
L.557 C      ------------------------------------------------------------
L.558 C       CHECK WORD FOR CURRENT TABLES
L.559        integer ipack/'CSIR'/
L.560 C       TO BE CHECKED AGAINST FIRST WORD OF INPUT FILE
L.561 C
L.562 C      SIZE OF FONT ARRAY=NO.OF FONTS ON DISK + 1
L.563        data nfont/4/
L.564 C
L.565 C      TYPESETTER CODE TABLES
L.566 C      ----------------
```

```
L.567 C      OPERATIONS TABLE
L.568        integer*2 optab(256)
L.569        data optab/1,3*6,19,1,3*1,3*7,1,3*1,11,7*10,2*9,17,18,11*1,
L.570       120,8*1,8*4,3,4,2,3,4,2*3,2,15,2*5,4,2*3,2*2,3,4,16,15,13,15,
L.571       214,12,15,4,3,3*2,4,3,2*10,15,16,13,2*15,16,2*15,2,3,2,2*3,2,3,2*2,
L.572       33*15,12,15,10*4,15,2*16,15,12,15,19,9*15,2,2*3,4*19,9*15,6*1,2*19,
L.573       48*15,4*5,3,4,2*3,2*2,3,2,5*3,5*2,7,9*15,7*19,9*15,8*19,8*15,
L.574       56*19,10*15,2*2,3,4,2*19/
L.575 C
L.576 C      MNEMONICS TABLE
L.577        logical*1 mntab(1688)/1688*' '/
L.578 C      USED IN QCALC
L.579        common /mnt/qc0,qrmn
L.580 C      NO. OF MNEMONICS
L.581        data kmn/274/
L.582 C      HEX CODES FOR MNEMONICS TABLE
L.583 C      MNEMONIC 2BYTES,HEXCODE 1 BYTE
L.584        integer*2 hxtab(274)/'IG',Z00,'EM',Z01,'EI',Z02,'EB',Z03,'NT',Z05,
L.585       1'EL',Z06,'EC',Z07,'ER',Z08,'LF',Z09,'CR',Z0A,'HA',Z0B,'ZW',Z0C,
L.586       2'DA',Z0D,'DN',Z0E,'DM',Z0F,'LL',Z10,'PS',Z11,'PL',Z12,'SL',Z13,
L.587       3'TL',Z14,'IA',Z15,'IB',Z16,'IC',Z17,'SE',Z18,'UE',Z19,'LD',Z1A,
L.588       4'SR',Z1B,'SS',Z1C,'SH',Z1D,'UN',Z1E,'US',Z20,'AL',Z21,'UA',Z22,
L.589       5'UB',Z23,'AI',Z24,'UC',Z25,'CI',Z26,'UL',Z27,'MT',Z28,'ER',Z29,
L.590       6'NE',Z2A,'EQ',Z2B,'TA',Z30,'TB',Z31,'TC',Z32,'TD',Z33,'TE',Z34,
L.591       7'TF',Z35,'TG',Z36,'TH',Z37,'TI',Z38,'TJ',Z39,'TO',Z3A,'TS',Z3B,
L.592       8'TR',Z3C,'TP',Z3D,'TM',Z3E,'TO',Z3F,'MO',Z41,'MC',Z42,'MP',Z43,
L.593       9'MI',Z44,'MA',Z45,'MD',Z46,'MM',Z47,'MR',Z48,'MV',Z49,'OS',Z51,
L.594       1'OT',Z52,'OL',Z53,'OR',Z54,'OC',Z55,'OD',Z56,'OM',Z57,'AA',Z62,
L.595       1'AG',Z63,'AD',Z64,'AC',Z65,'AV',Z66,'AW',Z67,'AX',Z68,'AY',Z69,
L.596       2'AZ',Z6A,'BA',Z70,'BB',Z71,'BC',Z72,'BD',Z73,'BE',Z74,'BF',Z75,
L.597       3'BG',Z76,'BH',Z77,'BI',Z78,'BJ',Z79,'BO',Z8A,'BS',Z8B,'BP',Z8C,
L.598       4'SN',ZAA,'SD',ZAB,'SC',ZAC,'SI',ZAD,'GA',ZB0,'GB',ZB1,'GC',ZB2,
L.599       5'GD',ZB3,'GG',ZB4,'GI',ZB5,'GL',ZB6,'GM',ZB7,'GN',ZB8,'GO',ZB9,
L.600       6'GP',ZBA,'GQ',ZBB,'GS',ZBC,'GT',ZBD,'GW',ZBE,'GZ',ZBF,'LW',ZFA,
L.601       7'LX',ZFB,'LY',ZFC,'LZ',ZFD,'LT',ZC0,'RB',ZAE,'RT',ZAF,'FF',ZA0,
L.602       8'RF',Z90,'FS',Z80,'DC',Z2C,'DE',Z2D,'DJ',Z9F,'DL',Z2F,'DB',Z1F,
L.603       9'DD',Z9A,'DF',Z9B,'DG',Z9C,'DH',Z9D,'DI',Z9E,'DK',Z2E/
L.604 C      SIZE OF MNTAB
L.605        qrmn=41
L.606 C      START OF MNTAB
L.607        qc0=192
L.608 C
```

```
L.509 C      END OF SPECIFIC DATA
L.610 C      ----------------------------------
L.611 C
L.612 C      INITIALISING
L.613 C      ------------
L.614        qbl=qntg(lbl)
L.615 C      BUILD UP MNTAB FROM HXTAB
L.616        do 5 i=1,kmn,2
L.617        qcode=hxtab(i)
L.618        qhex=hxtab(i+1)
L.619        mntab(qcalc(lqcode,ierr))=lhex(2)
L.620     5  CONTINUE
L.621 C      SETTING HEXTAB CODES IN FLAGS AND FONTS,ENSURING CONSISTENCY WITH MNTAB
L.622        do 14 i=1,nflg
L.623        icode=iflg(i)
L.624        licode(1)=mntab(qcalc(licode,ierr))
L.625        licode(2)=lbl
L.626        iflg(i)=icode
L.627    14  continue
L.628 C      SET LG
L.629        icode=iflg(nflg)
L.630        lg=licode(1)
L.631        qlg=qntg(lg)
L.632 C      SET QRF
L.633        icode=iflg(2)
L.634        qrf=qntg(licode(1))
L.635 C      SET FONTS IN FONT ARRAY
L.636        nfn=nflg-nfont
L.637        do 15 i=1,nfont
L.638        icode=iflg(i+nfn)
L.639        lfnt(i)=licode(1)
L.640    15  continue
L.641 C      SETTING OPTAB VALUE FOR LD IN IFLG(4)
L.642        qopld=optab(qntg(lpld)+1)
L.643        ld=lg
L.644        if(ierfl .eq. 0) go to 10
L.645     3  write(ioup,4) ierfl
L.646     4  format(/' (23)',1X,I3,' ERRORS FOUND DURING INITIALISING'/)
L.647    10  write(ioup,11)
L.648    11  format(' INITIALISATION COMPLETED'/)
L.649        return
L.650 C
```

```
L.651 C      CHECK VERSION OF PROGRAM
L.652 C      -----------------
L.653        entry check(iflck)
L.654 C      --------
L.655        if(ipgck .eq. iflck) go to 8
L.656        write(ioup,6) iflck,ipgck
L.657     6  format(/' (5) CHECK CODE IS ',A4,' INSTEAD OF ',a4,':STOP'/)
L.658        stop
L.659     8  return
L.660 C
L.661 C      CONVERT MNEMONICS TO HEX CODES AND GET OPTAB VALUE
L.662 C      ------------------------------------
L.663        entry opmn(qhx,ldx,lmn,/j/,mx,*,*)
L.664 C      --------------------------
L.665        j=j+2
L.666        if(j .gt. mx) return 2
L.667        ldx=mntab(qcalc(lmn,ierr))
L.668        if(ierr .ge. 2) return 1
L.669        qhex=qntg(ldx)
L.670        if(qhex .eq. qbl) go to 54
L.671   100  qhx=optab(qhex+1)
L.672        return
L.673    54  write(ioup,53) lmn(1),lmn(2)
L.674    53  format(/' (7) MNEMONIC ',2A1,' NOT IN MNTABLE')
L.675        return 1
L.676        entry opch(qhx,ldx)
L.677        qhex=qntg(ldx)
L.678        go to 100
L.679        end
L.680 C
L.681 C       --------------------
L.682        SUBROUTINE CONV(IATCD,LINCD,LCODE)
L.683 C      PROCESS COMMAND STRINGS OF MARKER TABLE
L.684        implicit logical*1 (l),integer*2 (q)
L.685 C      ARRAYS FOR ORIGINAL AND CONVERTED COMMAND STRINGS
L.686        logical*1 lcode(mxlcod),lincd(inbf)
L.687 C      DELIMITERS
L.688 C      FOR HANDLING CHARACTERS
L.689        logical*1 lqcode(2)
L.690        equivalence (ich,lch),(qcode,lqcode(1))
```

```
L.691 C      FLAG ARRAY
L.692 C      HEX CODES FOR FLAGS MARKER(FF),FONT RESTORE(FR),AND FONT STORE(FS)
L.693        equivalence (lff,iflg(1)),(lrf,iflg(2)),(lfs,iflg(3))
L.694        common ioup,iout,in,iwd
L.695        common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L.696        common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.697 C      /TBLE/ ELSEWHERE CONDENSED:ICM TO IAT=IDEL
L.698        common /tble/icm,id(3),ibl,iat,ndk,iflg(8),nflg,nbd,nldl,qlg,goold
L.699       1,qrf
L.700        common /dims/mxlcod,inbf,inchar,kouchs
L.701        common /points/k,kf,ik,il,ic,ih
L.702        ich=ibl
L.703        jrp=1
L.704        jrf=1
L.705        il=0
L.706 C      LEAVE SPACE FOR FLAGS
L.707        if(ic+3 .gt. mxlcod) go to 155
L.708        lcode(ic+1)=lg
L.709        lcode(ic+2)=lg
L.710        lcode(ic+3)=lg
L.711        ic=ic+3
L.712 C      COMMAND STRINGS INVOLVING C/R,FONT STORE OR FONT RESTORE ARE PRECEDED
L.713 C      BY FLAG LFF=ZA0:POS 1=FONT RESTORE,POS 2=FONT STORE,POS 3=END OF LINE
L.714 C      POINTER TO FIRST FLAG
L.715        kf=ic-2
L.716    30  go to(76,72),jrp
L.717 C
L.718 C      TEST FOR END OF REPEATED BLOCK
L.719 C      ------------------------------
L.720    72  if(il .lt. irl) go to 76
L.721 C      REPEAT THE BLOCK IRT TIMES
L.722        kre=ic
L.723        krbl=krb+1
L.724 C      TEST FOR COMMAND ARRAY OVERFLOW
L.725        if(irt*(kre-krb)+kre .le. mxlcod) go to 73
L.726        write(ioup,77) iat,iatcd
L.727    77  format(/' (27) REPEATED COMMAND IN ',2A1,' OVERFLOWS LCODE ARRAY')
L.728        go to 42
L.729 C      WRITE OUT REPEATED COMMAND IRT TIMES
```

```
L.730 C      IF RESTORE FONT FLAG IS SET ,SET BRANCH
L.731    73  if(qntg(lcode(kre)) .ne. qrf) go to 33
L.732        if(qcomop(lcode(krb1+1),lcode(kre-1)) .lt. 2) go to 33
L.733        ic=ic-1
L.734        krb1=krb1+1
L.735        kre=kre-1
L.736        jrf=2
L.737    33  do 74 i=1,irt
L.738        do 74 j=krb1,kre
L.739        ic=ic+1
L.740    74  lcode(ic)=lcode(j)
L.741        go to(34,32),jrf
L.742 C      ADD FINAL RESTORE FONT IF IT HAS BEEN REMOVED
L.743    32  ic=ic+1
L.744        lcode(ic)=lcode(kre)
L.745    34  jrp=1
L.746        jrf=1
L.747 C
L.748 C      NEXT DELIMITER
L.749 C      --------------
L.750    75  continue
L.751    76  call delim(lincd,&170,%100,%63,&750,%75,%148)
L.752 C      CHARACTER NOT A DELIMITER OR BLANK
L.753    39  write(ioup,40)
L.754    40  format(/' (6) WRONG DELIMITER')
L.755        lch=lincd(il)
L.756        il=il+2
L.757        if(il .gt. inbf) go to 148
L.758        lqcode(1)=lincd(il-1)
L.759        lqcode(2)=lincd(il)
L.760    42  write(ioup,45) lch,qcode,iat,iatcd,lg
L.761    45  format(' DELIMITER ',A1,' AND MNEMONIC OR TEXT ',A2,' IN ',2A1,' R
L.762       1EPLACED BY ',Z2/)
L.763        ierfl=ierfl+1
L.764        ic=ic+1
L.765        if(ic .gt. mxlcod) go  to 155
L.766        lcode(ic)=lg
L.767 C      LOOK FOR NEXT COMMAND OR TEXT DELIMITER
L.768        iil=il
```

```
L.769         do 47 i=iil,inbf
L.770         call delim(lincd,&170,&100,&63,&750,&47,&148)
L.771      47  continue
L.772         go to 150
L.773 C
L.774 C       REPEATED STRING
L.775 C        -------------
L.776 C       PREVENT NESTING
L.777   750   go to (78,145),jrp
L.778    78   il=il+4
L.779         if(il .gt. inbf) go to 148
L.780 C       NO. OF REPETITIONS
L.781         call num(irt,lincd(il-3),lincd(il-2),&42)
L.782 C       LESS 1 FOR FIRST ROUND
L.783         irt=irt-1
L.784         if(irt .gt. 0)jrp=2
L.785 C       LENGTH OF STRING
L.786         call num(irl,lincd(il-1),lincd(il),&42)
L.787 C       END OF STRING
L.788         irl=irl+il
L.789         krb=ic
L.790         go to 76
L.791 C
L.792 C       TEXT IMBEDDED IN COMMAND STRING
L.793 C        ----------------------------
L.794    49   call delim(lincd,&170,&100,&65,&750,&71,&148)
L.795         go to 65
L.796 C       INSERT BLANKS IN IMBEDDED REXT
L.797    71   ic=ic+1
L.798         if(ic .gt. mxlcod) go to 155
L.799         lcode(ic)=lincd(il)
L.800         go to 49
L.801 C       THE CHARACTER FOLLOWING &
L.802    63   il=il+1
L.803         if(il .gt. inbf) go to 148
L.804 C       USING OPTAB TO CHECK TEXT EMBEDDED IN COMMAND STRINGS
L.805    65   call opch(qhx,lincd(il))
L.806         go to(60,60,60,60,60,60,60,60,60,60,60,64,64,64,66,68,60,60,60,
L.807        160),qhx
```


Appendix 11 cont.

```
L.808 C      1 TO 11,17 TO 20=ERROR.TYPESETTER COMMAND NOT TEXT
L.809    60  write(ioup,62) lincd(il)
L.810    62  format(/' (17) EMBEDDED TEXT ',Z2,' IS NOT EBCDIC')
L.811    70  lqcode(1)=lincd(il)
L.812        lqcode(2)=lbl
L.813        go to 42
L.814 C      12 TO 14=INSERT FONT,HEX CODE,RESTORE FONT FLAG
L.815    64  qhx=qhx-11
L.816        call chfon(lcode,lff,lfnt(qhx),lincd(il),lrf,&155)
L.817        go to 49
L.818 C      15=NO FONT CHANGE
L.819    66  ic=ic+1
L.820        if(ic .gt. mxlcod) go to 155
L.821        lcode(ic)=lincd(il)
L.822        go to 49
L.823 C      16=CHARACTER NOT ON DISK
L.824    68  write(ioup,69) lincd(il),lincd(il)
L.825    69  format(/' (25) ',A1,' (',Z2,')',' IS NOT  ON TYPESETTER DISK')
L.826        go to 42
L.827 C
L.828 C      COMMAND
L.829 C      -------
L.830 C      GET HEX CODE FOR MNEMONIC
L.831   100  call opmn(qhx,lhx,lincd(il+1),il,inbf,&42,&148)
L.832        go to(210,220,220,220,210,260,270,280,280,280,280,320,320,320,320
L.833       1,320,326,326,250,340),qhx
L.834 C      1,5=INSERT HEX CODE AND CONTINUE
L.835   210  ic=ic+1
L.836        if(ic .gt. mxlcod) go to 155
L.837        lcode(ic)=lhx
L.838        go to 30
L.839 C      2,3,4=INSERT FONT,ADDITIONAL CHAR HEX,AND LFF AND LRF FLAGS
L.840   220  j=qhx-1
L.841        call chfon(lcode,lff,lfnt(j),lhx,lrf,&155)
L.842        go to 30
L.843 C      19=ERROR:UNASSIGNED HEX CODE
L.844   250  write(ioup,255) qcode
L.845   255  format(/' (8) ',A2,' HAS NO HEX CODE')
L.846        go to 42
```

```
L.847 C      6=INSERT FONT STORE CODE AND FF FLAG IN POS 2
L.848  260   call mark(1,lcode,lff,lhx,lfs,&155)
L.849        go to 30
L.850 C      7=INSERT FF FLAG IN POS 3 FOR CR LE HA OR LT
L.851  270   call mark(2,lcode,lff,lhx,lg,&155)
L.852        go to 30
L.853 C      8,9,10,11=INSERT HEX CODE AND FOLLOWING DIGITS
L.854  280   j=qhx-7
L.855        call follow(j,lcode,lhx,lincd,&155,&148)
L.856        go to 30
L.857 C      12 TO 16=ERROR:KEYBOARD CHAR NOT TELEPRINTER COMMAND
L.858  320   write(ioup,325) qcode
L.859  325   format(/' (9) ',A2,' IS A KEYBOARD CHAR.')
L.860        go to 42
L.861 C      20=UL,CHAR MAY ROUIRE FONT CHANGE
L.862  340   if(qntg(ld) .ne. qlg) go to 345
L.863 C      LEAD CHAR AVAILABLE IN ALL FONTS
L.864        ic=ic+1
L.865        if(ic .gt. mxlcod) go to 155
L.866        lcode(ic)=lhx
L.867        go to 30
L.868 C      LEAD CHAR REQUIRES CHANGE OF FONT
L.869  345   call chfon(lcode,lff,ld,lhx,lrf,&155)
L.870        go to 30
L.871 C      17,18=INSERT FLAGS FOR LEADER OR RULE CHARACTER
L.872 C      TEST IF RULE OR LEADER CHAR. IS EBCDIC OR FROM DISK
L.873  326   il=il+1
L.874        if(il .gt. inbf) go to 148
L.875        lch=lincd(il)
L.876        if(ich .ne. icm) go to 328
L.877 C      CHAR. FROM DISK
L.878        call opmn(qcf,lch,lincd(il+1),il,inbf,&42,&148)
L.879        go to 330
L.880 C      CHAR. IS EBCDIC
L.881  328   call opch(qcf,lch)
L.882  330   go to(338,334,334,334,334,338,338,338,338,338,338,333,333,333,332
L.883       1,336,338,338,338,338),qcf
L.884 C      5,15=CURRENT FONT REQUIRED FOR LEADER OR RULE CHARACTER
L.885  332   ic=ic+3
L.886        if(ic .gt. mxlcod) go to 155
L.887        if(ich .ne. ibl) go to 329
```

```
L.898 C      IF RULE IS BLANK,PRECEDING FONT MUST BE BLANK
L.899         lcode(ic-2)=lbl
L.900        go to 331
L.901  329   lcode(ic-2)=lfon
L.902  331   lcode(ic-1)=lhx
L.903        lcode(ic)=lch
L.904        if(qhx .eq. qopld) ld=lg
L.905        go to 30
L.906 C      2 TO 4,12 TO 14=FONT CHANGE REQUIRED
L.907  333   qcf=qcf-11
L.908        go to 335
L.909  334   qcf=qcf-1
L.910  335   call leader(lcode,lff,lrf,lfnt(qcf),lhx,lch,&155)
L.911        if(qhx .eq. qopld) ld=lfnt(qcf)
L.912        go to 30
L.913 C      16=CHARACTER NOT AVAILABLE ON TYPESETTER DISK
L.914  336   write(ioup,337) lincd(il+1),lincd(il+1)
L.915  337   format(/' (10) LEADER CHAR ',A1,'(',Z2,')', ' IS NOT ON THE DISK')
L.916        go to 140
L.917 C      1,6 TO 11 OR 17 TO 20=INVALID -IS A COMMAND
L.918  338   write(ioup,339) lincd(il+1),lincd(il+1)
L.919  339   format(/' (11) ',A1,'(',Z2,') :  INVALID LEADER CHARACTER')
L.920  140   il=il+1
L.921         if(il .gt. inbf) go to 148
L.922        go to 42
L.923 C
L.924 C      ERROR REPORTS
L.925 C      -------------
L.926  145   write(ioup,146) iat,iatcd
L.927  146   format(/' (19) NESTED REPETITIONS IN ',2A1)
L.928        ierfl=ierfl+1
L.929        go to 42
L.930  148   write(ioup,149)iat,iatcd
L.931  149   format(/' (18) COMMAND STRING IN ',2A1,' TRUNCATED:LENGTH > LINCD
L.932       1ARRAY')
L.933        ierfl=ierfl+1
L.934  150   if(ic .le. mxlcod) go to 170
L.935  155   write(ioup,160)
L.936  160   format(/' (12) LCODE ARRAY FILLED:STOP'/)
L.937        stop
L.938 C
```

```
L.929 C      END OF STRING
L.930 C      -------------
L.931  170   nc=ic-kf+1
L.932        if(nc .le. imbrec) go to 180
L.933        write(ioup,175) iat,iatcd,nc,imbrec
L.934  175   format(/' (20) LENGTH OF ',2A1,'=',I4,' EXCEEDS TAPE RECORD=',I4)
L.935        ierfl=ierfl+1
L.936  180   return
L.937        end
L.938 C
L.939 C      ------------------------------
L.940        SUBROUTINE KTEST(KCH,CH)
L.941 C      PROCESSING TEXT
L.942        implicit logical*1 (l),integer*2 (q)
L.943        logical*1 ch(inchar),kch(kouchs)
L.944        common ioup,iout,in,iwd
L.945        common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.946        common /dims/mxlcod,inbf,inchar,kouchs
L.947        common /points/k,kf,ik,il,ic,ih
L.948 C      TRANSFER NEXT CHAR SUBJECT TO OPTAB REQUIREMENTS
L.949        call opch(qhx,ch(ik))
L.950        go to(510,510,510,510,510,510,510,510,510,510,510,520,520,520,530
L.951       1,540,510,510,510,510),qhx
L.952 C      1 TO 11,17 TO 20=ERROR:TYPESETTER COMMAND INSTEAD OF TEXT CHARACTER
L.953  510   write(ioup,515) ch(ik)
L.954  515   format(/' (13) HEX CODE ',Z2,' IS NOT AN EBCDIC CHARACTER')
L.955        call errep(ch,kch,ltxr)
L.956        go to 550
L.957 C      12,13,14=INSERT SPECIFIED FONT,HEX CODE,AND RESTORE FONT
L.958  520   qhx=qhx-11
L.959        if(qhx .eq. qntg(lfon)) go to 530
L.960        k=k+1
L.961        if(k+2 .le. kouchs) go to 525
L.962        krm=kouchs-k
L.963        do 523 i=1,krm
L.964  523   kch(k+i-1)=lg
L.965        call wrout(kch)
L.966  525   kch(k)=lfnt(qhx)
L.967        kch(k+1)=ch(ik)
L.968        kch(k+2)=lfon
L.969        k=k+2
L.970        go to 550
```

```
L.971 C      15=NO FONT CHANGE
L.972  530   k=k+1
L.973        if(k .le. kouchs) go to 534
L.974        call wrout(kch)
L.975  534   kch(k)=ch(ik)
L.976  550   return
L.977 C      16=CHARACTER NOT ON DISK
L.978  540   write(ioup,545) ch(ik),ch(ik)
L.979  545   format(/' (14) ',A1,' (',Z2,')',' IS NOT  ON TYPESETTER DISK')
L.980        call errep(ch,kch,ltxr)
L.981        go to 550
L.982        end
L.983 C
L.984 C      ------------------------------------------------------------
L.985        SUBROUTINE HEAD(KCH,KON)
L.986 C      PROCESSING HEADING PARAMETERS
L.987        implicit logical*1 (l),integer*2 (q)
L.988        logical*1 kch(kouchs)
L.989        common ioup,iout,in,iwd
L.990        common /counts/inflag,imbrec,ithrec,kblk,inlins,ierfl,mrecou,nc
L.991        common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L.992        common /thle/idel(6),ndk,iflg(8),nflg,nhd,nldl,qlg,qopld,qrf
L.993        common /dims/mxlcod,inbf,inchar,kouchs
L.994        common /points/k,kf,ik,il,ic,ih
L.995        ih=kon
L.996  690   continue
L.997        call opmn(qhx,lhx,lh(ih+1),ih,nhd,&690,&800)
L.998        go to(730,760,760,760,730,730,730,740,740,740,740,760,760,760,760,
L.999       1760,760,760,760,760),qhx
L1000 C      INSERT HEX CODE AND CONTINUE
L1001  730   k=k+1
L1002        kch(k)=lhx
L1003        go to 690
L1004 C      COMMAND FOLLOWED BY DIGITS
L1005  740   qhx=qhx-7
L1006        k=k+1
L1007        kch(k)=lhx
L1008        do 755 i=1,qhx
L1009        kch(k+i)=lh(ih+i)
L1010  755   continue
```

```
L1011         k=k+qhx
L1012         ih=ih+qhx
L1013         go to 690
L1014 C       NOT A HEADING PRINT PARAMETER
L1015   760   write(ioup,770) lh(ih-1),lh(ih)
L1016   770   format(' (26) ',2A1,' IS NOT PROVIDED FOR IN HEADING COMMANDS'/)
L1017         ierfl=ierfl+1
L1018         go to 690
L1019   800   return
L1020         end
L1021 C
L1022 C       ----------------------
L1023         SUBROUTINE INSFON(EM,LEM)
L1024         implicit logical*1 (l),integer*2 (q)
L1025         logical*1 em(3)
L1026         common ioup,iout,in,iwd
L1027         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1028         common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L1029         qj=0
L1030         call opch (qm,lem)
L1031         go to(640,600,600,600,620,640,640,640,640,640,640,610,610,610,620
L1032        1,640,640,640,640,640),qm
L1033 C       2-4=ADDITIONAL CHARACTER WITH FONT CHANGE
L1034   600   qj=qm-1
L1035         go to 625
L1036 C       12-14=EBCDIC CHARACTER WITH FONT CHANGE
L1037   610   qj=qm-11
L1038         go to 625
L1039 C       5,15=CHARACTER AVAILABLE IN ALL FONTS
L1040   620   qj=1
L1041   625   em(1)=lfnt(qj)
L1042         em(2)=lem
L1043   630   return
L1044 C       1,6-11,16-20=MARKER NOT ON TYPESETTER DISK
L1045   640   write(ioup,650) lem,lem
L1046   650   format(' (16) ERROR MARKER',A3,'(',Z2,')NOT ON TYPESETTER DISK'/)
L1047         ierfl=ierfl+1
L1048         go to 630
L1049 C        return
L1050         end
L1051 C       ---------------------
```

```
L1052          SUBROUTINE WROUT(KCH)
L1053 C       EMPTY OUTPUT BUFFER
L1054         logical*1 kch(kouchs)
L1055         common ioup,iout,in,iwd
L1056         common /dims/mxlcod,inbf,inchar,kouchs
L1057         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1058         common /points/k,kf,ik,il,ic,ih
L1059         write(iout,10) (kch(i),i=1,k)
L1060    10   format(120a1)
L1061         mrecou=mrecou+klim(k,imbrec)
L1062         k=1
L1063         return
L1064         end
L1065 C       ---------------------
L1066         SUBROUTINE WRIN(CH)
L1067 C       FILL INPUT BUFFER TESTING FOR END OF DATA
L1068         logical*1 ch(inchar)
L1069         common ioup,iout,in,iwd
L1070         common /dims/mxlcod,inbf,inchar,kouchs
L1071         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1072         common /points/k,kf,ik,il,ic,ih
L1073 C       INPUT TEXT
L1074         ik=0
L1075         read(in,40,end=50)(ch(i),i=1,inchar)
L1076 40      format(80a1)
L1077         kblk=kblk+1
L1078         go to 56
L1079    50   inflag=1
L1080         inchar=i-1
L1081    56   return
L1082         end
L1083 C       ---------------------
L1084         SUBROUTINE IKTEST(CH,LCH,KCH)
L1085 C       PICK UP NEXT CHAR. TESTING FOR END OF INPUT BUFFER
L1086         implicit logical*1 (l),integer*2 (q)
L1087         logical*1 ch(inchar),kch(kouchs)
L1088         common ioup,iout,in,iwd
L1089         common /dims/mxlcod,inbf,inchar,kouchs
L1090         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1091         common /points/k,kf,ik,il,lc,lh
```

```
L1092         ik=ik+1
L1093         if(ik .le. inchar) go to 20
L1094         if(inflag .eq. 0) go to 10
L1095         klines=kblk*inlins+klim(inchar,itbrec)
L1096         write(ioup,55) klines
L1097      55 format(//' END OF CONVERSION'/' -----------'//I6,' LINES ENTERED')
L1098         call wrout(kch)
L1099         write(ioup,65) ierfl,mrecou
L1100      65 format(//I6,' ERRORS'//I6,' RECORDS WRITTEN'//)
L1101         end file iout
L1102         stop
L1103      10 call wrin(ch)
L1104         ik=1
L1105 C       INPUT CHARACTERS LEFT ADJUSTED
L1106      20 lch=ch(ik)
L1107         return
L1108         end
L1109 C        --------------
L1110         SUBROUTINE CHFON(LCODE,LFF,LF,LKODE,LR,*)
L1111 C       SET FRONT MARKER,FONT,CHARACTER AND RESTORE FONT FLAG IN COMMAND STRING
L1112         implicit logical*1 (l),integer*2 (q)
L1113         common /points/k,kf,ik,il,ic,ih
L1114         common /dims/mxlcod,inbf,inchar,kouchs
L1115         logical*1 lcode(mxlcod)
L1116 C       TEST IF PRECEDING CHAR REQUIRED SIMILAR FONT CHANGE
L1117 C       INITIAL FLAGS ISOLATE COMMAND STRINGS
L1118         if(qntg(lcode(ic)) .ne. qntg(lr)) go to 10
L1119         qjmp=qcomop(lcode(ic-1),lkode)
L1120         go to(10,5),qjmp
L1121       5 ic=ic+1
L1122         if(ic .gt. mxlcod) return 1
L1123         go to 20
L1124      10 lcode(kf)=lff
L1125         ic=ic+3
L1126         if(ic .gt. mxlcod) return 1
L1127         lcode(ic-2)=lf
L1128      20 lcode(ic-1)=lkode
L1129         lcode(ic)=lr
L1130         return
L1131         end
L1132 C       ---------------
```

```
L1133         SUBROUTINE MARK(KON,LCODE,LFF,LKODE,LM,*)
L1134  C      SET FRONT MARKER,CHARACTER AND FLAG IN COMMAND STRING
L1135         implicit logical*1 (l),integer*2 (q)
L1136         common /points/k,kf,ik,il,ic,ih
L1137         common /dims/mxlcod,inbf,inchar,kouchs
L1138         logical*1 lcode(mxlcod)
L1139         lcode(kf+kon)=lff
L1140         ic=ic+2
L1141         if(ic .gt. mxlcod) return 1
L1142         lcode(ic-1)=lkode
L1143         lcode(ic)=lm
L1144         return
L1145         end
L1146  C      ----------------
L1147         SUBROUTINE FOLLOW(M,LCODE,LKODE,LINCD,*,*)
L1148  C      PICK UP M FOLLOWING CHARACTERS FOR COMMAND STRING
L1149         implicit logical*1 (l),integer*2 (q)
L1150         common /points/k,kf,ik,il,ic,ih
L1151         common /dims/mxlcod,inbf,inchar,kouchs
L1152         logical*1 lcode(mxlcod),lincd(inbf)
L1153         if(ic+m+1 .gt. mxlcod) return 1
L1154         ic=ic+1
L1155         lcode(ic)=lkode
L1156         do 400 i=1,m
L1157   400   lcode(ic+i)=lincd(il+i)
L1158         ic=ic+m
L1159         il=il+m
L1160         if(il .gt. inbf) return 2
L1161         return
L1162         end
L1163  C      -------------
L1164         SUBROUTINE LEADER(LCODE,LFF,LRF,LF,LKODE,LIN,*)
L1165  C      SET FRONT MARKER,FONT,COMMAND,CHARACTER,RESTORE FONT FLAG
L1166  C      STORE FONT OF LEAD CHAR.
L1167         implicit logical*1 (l),integer*2 (q)
L1168         common /points/k,kf,ik,il,ic,ih
L1169         common /dims/mxlcod,inbf,inchar,kouchs
L1170         logical*1 lcode(mxlcod)
L1171         lcode(kf)=lff
L1172         ic=ic+4
L1173         if(ic .gt. mxlcod) return 1
```

```
L1174         lcode(ic-3)=lf
L1175         lcode(ic-2)=lkode
L1176         lcode(ic-1)=lin
L1177         lcode(ic)=lrf
L1178         return
L1179         end
L1180 C       ------------------
L1181         FUNCTION ILINE(IK)
L1182 C       CALCULATE LINE NUMBER IN TOTAL INPUT
L1183         common /counts/inflag,imbrec,ltbrec,kblk,inlins,ierfl,mrecou,nc
L1184         iline=(max0(kblk,1)-1)*inlins+klim(ik,itbrec)
L1185         return
L1186         end
L1187 C       ------------------
L1188         FUNCTION QNTG(LOG)
L1189 C       CONVERT LOGICAL*1 TO INTEGER*2,RIGHT ADJUSTED
L1190         implicit logical*1 (l),integer*2 (q)
L1191         logical*1 lqntg(2)
L1192         equivalence (qjntg,lqntg(1))
L1193         qjntg=0
L1194         lqntg(2)=log
L1195         qntg=qjntg
L1196         return
L1197         end
L1198 C       ----------------
L1199         FUNCTION KLIM(IN,ID)
L1200         klim=(in-1)/id+1
L1201         return
L1202         end
L1203 C       -----------------
L1204         FUNCTION QCALC(LMN,IERR)
L1205 C       CONVERT MNEMONIC TO TABLE ADDRESS
L1206         implicit logical*1 (l),integer*2 (q)
L1207         logical*1 el1(2),el2(2),lmn(2)
L1208         equivalence (q1,el1(1)),(q2,el2(1))
L1209         common ioup,iout,in,iwd
L1210         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1211         common /mnt/qc0,qrmn
L1212         ierr=1
L1213         q1=0
L1214         q2=0
```

```
L1215         el1(2)=lmn(1)
L1216         el2(2)=lmn(2)
L1217         qcalc=(q1-qc0-1)*qrmn(q2-qc0)
L1218         if((qcalc .ge. 1) .and. (qcalc .le. qrmn*qrmn)) return
L1219       5 write(ioup,10) lmn(1),lmn(2),lmn(1),lmn(2)
L1220      10 format(' (15) MNEMONIC ',2A1,' (HEX ',2Z2,') IS OUTSIDE MNTAB'/)
L1221         ierfl=ierfl+1
L1222         ierr=2
L1223         return
L1224         end
L1225 C       ----------------
L1226         SUBROUTINE NUM(NUMB,LA,LB,*)
L1227 C       CONVERT TWO NUMERIC CHARACTERS TO AN INTEGER
L1228         implicit logical*1 (l),integer*2 (q)
L1229         common ioup,iout,in,iwd
L1230         qa=qntg(la)-240
L1231         if((qa .gt. 9) .or. (qa .lt. 0)) go to 10
L1232         qb=qntg(lb)-240
L1233         if((qb .gt. 9) .or. (qb .lt.  0)) go to 10
L1234         numb=qa*10+qb
L1235         if(numb .eq. 0) numb=1
L1236         return
L1237      10 write(ioup,20)
L1238      20 format(/' (22) INVALID REPETITION NUMBER')
L1239         return 1
L1240         end
L1241 C        ----------------
L1242         SUBROUTINE DELIM(LINCD,*,*,*,*,*,*)
L1243 C       TEST FOR DELIMITER
L1244         implicit logical*1 (l),integer*2 (q)
L1245         integer ich/'    '/
L1246         logical*1 lincd(inbf)
L1247         equivalence (ich,lch)
L1248         common /tble/idel(6),ndk,iflg(8),nflg,nhd,nldl,qlg,qopld,qrf
L1249         common /dims/mxlcod,inbf,inchar,kouchs
L1250         common /points/k,kf,ik,il,ic,ih
L1251    840  il=il+1
L1252         if(il .gt. inbf) return 6
L1253         lch=lincd(il)
L1254         do 900 i=1,ndk
```

```
L1255 C      IDEL=ICM,IMB,IRP,IGN,IBL
L1256        if(ich .eq. idel(i))go to 920
L1257  900   continue
L1258        return
L1259  920   go to(930,940,950,840,970),i
L1260 C      COMMAND DELIMITER
L1261  930   lch=lincd(il+1)
L1262 C      END OF COMMAND STRING?
L1263        if(ich .eq. idel(1)) return 1
L1264 C      NORMAL COMMAND
L1265        return 2
L1266 C      IMBEDDED TEXT DELIMITER
L1267  940   return 3
L1268 C      REPEAT DELIMITER
L1269  950   return 4
L1270 C      BLANK
L1271  970   return 5
L1272        end
L1273 C      --------------------
L1274        SUBROUTINE DEFAL
L1275 C      REPLACING BLANK DELIMITERS OR MARKERS  BY  DEFAULTS
L1276        implicit logical*1 (l),integer*2 (q)
L1277        Integer lbl/'    '/,izd/ZD0404040/,iz/'0   '/
L1278        equivalence (ich,lch)
L1279        common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L1280        common /tble/idel(6),ndk,iflg(8),nflg,nhd,nldl,qlg,qopld,qrf
L1281        ich=ibl
L1282        do 10 i=1,nldl
L1283        if(idel(i) .eq. iz) idel(i)=izd
L1284        if(idel(i) .ne. ibl) go to 10
L1285        lch=ldel(i)
L1286        idel(i)=ich
L1287   10   continue
L1288        return
L1289        end
L1290 C      --------------------
```

```
L1291         FUNCTION QCOMOP(L1,L2)
L1292  C      COMPARING OPTAB VALUES
L1293         implicit logical*1 (l),integer*2 (q)
L1294         qcomop=1
L1295         call opch(qhx1,l1)
L1296         call opch(qhx2,l2)
L1297         if(qhx1 .eq. qhx2) qcomop=2
L1298         return
L1299         end
L1300  C      --------------------
L1301         SUBROUTINE ERREP(CH,KCH,UL)
L1302  C      ERROR ROUTINE FOR TEXT SCANNING
L1303         implicit logical*1 (l),integer*2 (q)
L1304         logical*1 ch(inchar),kch(kouchs),ul(3)
L1305         common ioup,iout,in,iwd
L1306         common /dims/mxlcod,inbf,inchar,kouchs
L1307         common /counts/inflag,imbrec,itbrec,kblk,inlins,ierfl,mrecou,nc
L1308         common /logs/lh(32),lg,lbl,ld,lfon,ldel(6),latr(3),ltxr(3),lfnt(4)
L1309         common /points/k,kf,ik,il,ic,ih
L1310         iln=iline(ik)
L1311         ikc=ik-(iln-1)*itbrec
L1312         is=ik-4
L1313         ie=ik+4
L1314         write(ioup,517)  (ch(j),j=is,ie),ikc,iln
L1315   517   format(' IN STRING ',9A1,' CHAR.NO.',I4,' IN LINE ',I5/15X,'-')
L1316         write(ioup,519) ul(2),ul(2)
L1317   519   format(' CHARACTER IS WRITTEN TO TAPE AS',A2,'(',Z2,')'/)
L1318         ul(3)=lfon
L1319         do 530 i=1,3
L1320         k=k+1
L1321         if(k .le. kouchs) go to 520
L1322         call wrout(kch)
L1323   520   kch(k)=ul(i)
L1324   530   continue
L1325         ierfl=ierfl+1
L1326         return
L1327         end
```

Doc 135142

22/5