1 Introduction

1.1 Product Characteristics

TOM-DOC edits texts as ready-to-print documents based on commands entered with the text. Editing functions include automatic classification of a text into chapters, sub-chapters, sections, etc., line and page breaks, automatic hyphenation in a number of national languages and automatic generation of table of contents and keyword index.

TOM-DOC is particularly well suited for producing documents required for development projects, such as studies, specifications, performance descriptions and manuals.

TOM-DOC can be used directly as a separate program or interactively together with the product TOM-TI.

When TOM-DOC is called directly, the TOM-DOC texts already entered using the BS2000 editor $EDT are read from system input and the edited result is written to the SYSLST system file.

In interactive mode, the texts are entered and modified in the TOM-TI editor and formatted as documents using TOM-DOC. The original text and the formatted document can be displayed at the same time in different windows of the editor.

TOM-DOC (SINIX), which is largely compatible with TOM-DOC (BS2000), is available on the SINIX platform. This means that text files can be created and/or edited under either BS2000 or SINIX.

TOM-DOC (SINIX) is part of SoftBench-LNK and works with the MAXed/XM (SINIX) editor.

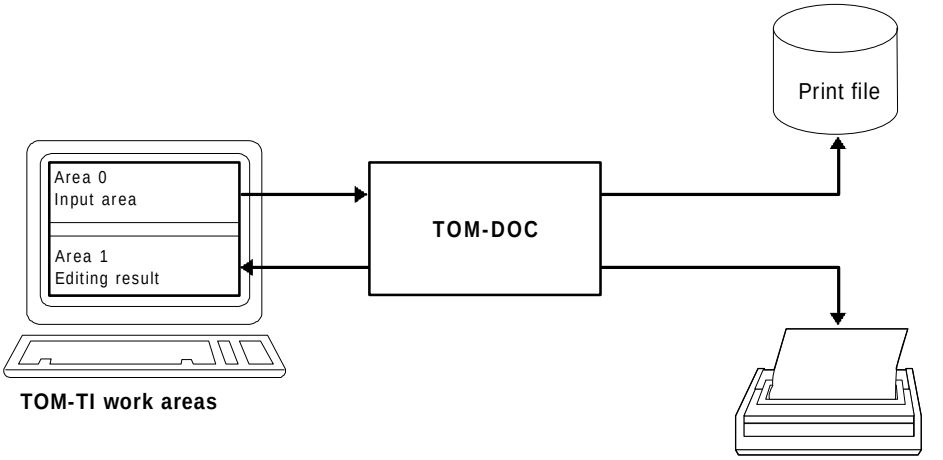


Figure 1-1: Interactive operation of TOM-DOC under TOM-TI

Description of functions

TOM-DOC offers the following functions:

- Definition of uniform page layout with header/footer
- Definition of variable page formats (DIN A3, DIN A4, ...)
- Pagination by chapter
- Automatic generation of table of contents and keyword index
- Processing of continuous text and fixed text segments
- Centering of texts
- Automatic classification of texts into chapters, sub-chapters, sections, etc.
- Automatic line/page breaks and table make-up
- Cross-referencing to other parts in the text
- Automatic hyphenation in the selected language
- Footnote handling
- Underlining, bold type
- Revision marking
- Insertion of text files
- Editing of simple graphics for laser printers
- Output of edited text direct to printer or to a BS2000 file

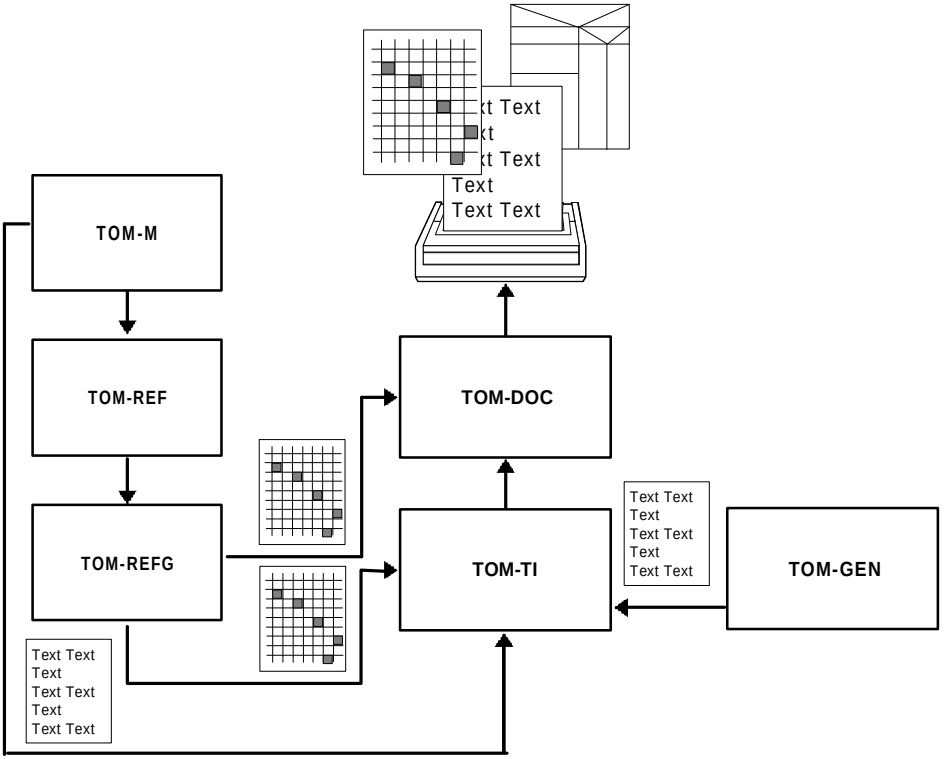


Figure 1-2: TOM-DOC composite tool

1.2 Structure of the Manual

This manual is designed to be used in two ways:

- as an **instruction manual** for users just starting to work with a text editor
- and as a **reference manual** for the various commands and dummy graphics characters.

The manual, therefore, has been divided into corresponding parts.

The first part (sections 1 and 2) gives you all the information required to work with TOM-DOC:

– how to enter commands
– how to start the text editor
– how to manage word division/hyphenation and
– how to generate dummy graphics.

The second part (section 3) presents two examples with detailed explanations of the specific commands for text and graphics input and using TOM-DOC via a central control file. If you have never worked with a text editor before, you should work through this part very carefully (ideally on the terminal).

The third part gives you detailed descriptions of the commands (section 4) and instructions on using TOM-DOC to generate printed output (section 5).

The appendix features tables which are designed for quick reference. It includes a list of error messages and troubleshooting notes.

Before starting to use TOM-DOC you should on all accounts read section 1.3 on command syntax and section 2 on TOM-DOC operation. The keyword index at the back and the table of contents are available as further aids.

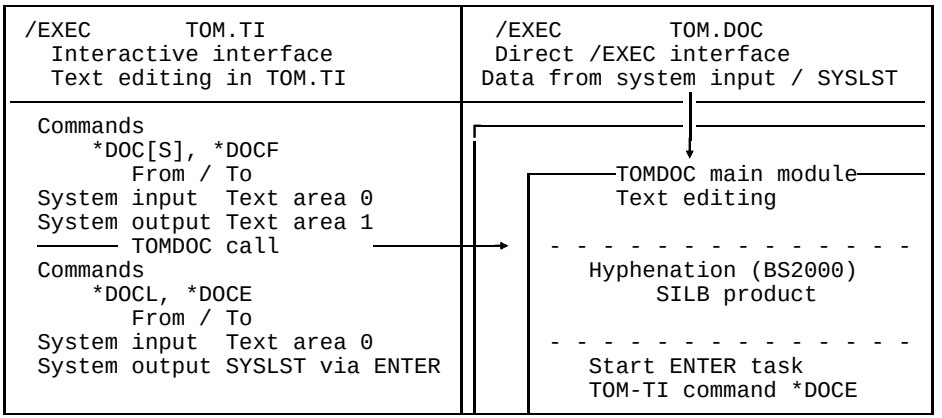
1.3 Calling TOM-DOC

TOM-DOC can be used directly as a BS2000 phase or interactively together with the product TOM-TI.

When you call the TOM-DOC BS2000 phase directly, the TOM-DOC texts already edited using the BS2000 editor $EDT are read from system input and the edited result is written to the SYSLST system file.

In interactive mode, the texts are entered and modified in the TOM-TI editor and formatted as documents using TOM-DOC. The original text and the formatted document can be displayed at the same time in different windows of the editor.

The following diagram shows how TOM-DOC is used in both direct and interactive mode:



TOM-DOC can be used in interactive mode in the TOM-TI editor by means of commands {*DOC[S], *DOCF, *DOCL, *DOCE} and in direct mode by means of an /EXEC call. In both modes, the TOM-DOC functionality is made available by calling the TOMDOC main module.

When TOM-DOC is called, the data is read from system input (SYSDTA, the interactive area of TOM-TI, or the LMS and FMS library systems) and written to SYSLST or the interactive area of TOM-TI.

1.4 Working with TOM-TI

You can call the TOM-TI editor as a BS2000 phase by means of `/EXECUTE` or `/START-PROGRAM FROM-FILE=$userid.TOM.TI` or from the menu interface of TOM-M.

1.4.1 Calling up TOM-TI

Log on and then call up the TOM-TI editor with the command:

```
/EXEC $userid.TOM.TI bzw. /START-PROGRAM FROM-FILE=$userid.TOM.TI
```

A screen is displayed with the following layout:

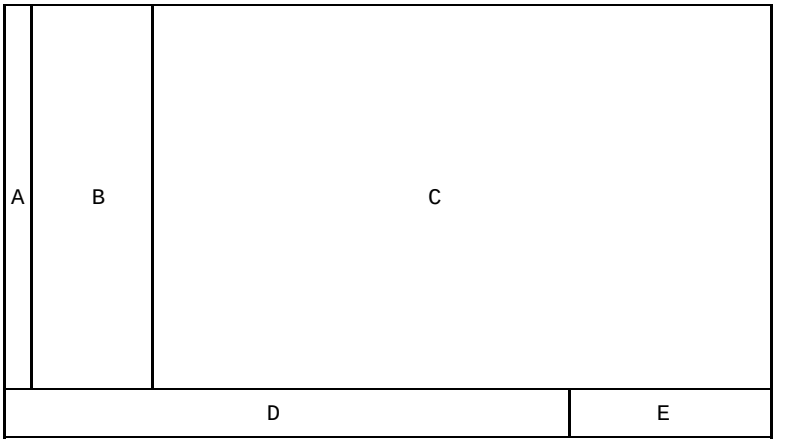


Figure 1-3: TOM-TI screen layout

The fields are designated as follows:

– Column A: **marking column**
– Column B: **line numbers**
– Field C: **text field**
– Line D: **command line**
– Field E: **status display**

When preparing text entry you must first give the command:

```
LOW [ON]
```

You enter this and the following commands in the command line D. TOM-TI can now output lowercase letters on the screen. The following commands are also very useful:

```
INDEX OFF
CODE ON
```

The INDEX OFF command switches the screen to 80 characters per line. The CODE ON command activates a facility for entering the special German characters, umlauts and "ß", not known by your keyboard.

Now you can start to enter your first text. Use the tabulator key to go to the next line, and store your text with the command

```
W[RITE]'testfile'
```

Your text is then stored under the name "testfile". With the command "R[EAD]'testfile'" you read the contents of the "testfile" back into the work area of the editor. You quit TOM-TI by entering the command "H[ALT]" in the command line.

1.4.2 Calling up TOM-TI from TOM-M

You call the TOM-TI editor by marking the field "Editor" in the TOM-M menu interface and confirming input with the DÜ1 key.

```
02     Selection                    TOM   EXAMPLE
-------------------------------------------------------------------------------
Component(s)       DATA....     ........     ........     ........     ........
Function
.      Services                                Special functions
.      Parameter generation                    Parameterset name
X..    Editor                                       ........
.      COLINDA                                      ........
.      COLLIST                                      ........
.      COLCOB                                       ........
.      Compiler                                     ........
.      COLNUMA
.      Link                                         ........
.      Test                                         ........
.      COLTEST                                      ........
.      FORMPLAG                                     ........
-------------------------------------------------------------------------------
Command: ......................................................................
-------------------------------------------------------------------------------


-------------------------------------------------------------------------------
LTG                                                  TAST
```

Figure 1-4: Calling up the TOM-TI editor from TOM-M

By setting marks in mask 2.01 (parameter generation function) you can define parameters for the editor, so that when you call the editor, the line length, screen areas and lowercase input are all preset.

1.4.3 Important TOM-TI Commands

The following list is just a limited overview of the commands which will facilitate working with TOM-TI. This is by no means complete and we recommend you to refer to the separate TOM-TI and EDT manuals.

```
LOW [ON]    : activates lowercase input mode
INDEX OFF   : suppresses line numbers
CODE ON     : selects German character set
W[RITE]     : writes text in a file
R[EAD]      : reads text from a file
H[ALT]      : quits the editor
+,-         : paging up or down in a file
>,<         : shifts screen contents to right or left
X           : in the marking column, marks a line to be edited
D           : in the marking column, deletes a line
@D[ELETE]
  or        : deletes the current work area
DE[LETE]
```

Like EDT the TOM-TI editor has several work areas. Immediately after calling TOM-TI you are in work area 0. You change work areas by entering the number of the desired work area in the command line. TOM-DOC usually works in work areas 0 to 3. It is useful, therefore, to have two work areas displayed at the same time on your screen. To do this, you use the command

```
SPLIT n(area)
```

where n indicates the number of lines and "area" the work area to be displayed on the screen. For more information on work areas, please refer to the EDT manual mentioned above.

Note

- When working under TOM-M menu control, you can use parameters to split the screen. When editing with *DOC or *DOCF, it is useful to have work area 2 (in which error messages are listed) or work area 1 (in which the edited text appears) as the second work area.
- You can start editing only from work area 0. If work areas 1 to 3 already contain data, TOM-DOC overwrites it with the results of current editing.

1.5 Changes since the Last Version of the Manual

1.5.1 New Functions

TOM-DOC V3.2A has the following new features:

- Support of NK2/4 disks for the internal work files. This does not affect the user interface in any way.
- Loading of LMS members with a user-defined PLAM type and version attribute. This affects the ..SW command.
- Printing on HP90 printers. The effects this has on the user interface are described at the relevant points in the manual and in chapter 5, "Printing with TOM-DOC", starting on page 155.

In addition to these functional changes, errors in the documentation of the last version have been corrected.

1.5.2 Compatibility

TOM-DOC(BS2000) V3.2.A is upwardly compatible with the previous version; in other words, input files created for the previous version can also be edited in TOM-DOC (BS2000) V3.2A.

1.5.3 Printer Support

3350 and 3352 ND Laser Printers can now only be used in BS2000 V10. They are supported in this version for the last time. As of BS2000/OSD V1, these printer types are replaced by the HP laser printers 2090-2, 2140-2, 2240-2, 3351 and 3353.

As of BS2000/OSD V1, when using TOM-DOC, the TOM-DOC command ..HP or DEV=HP must be inserted at the beginning of the TOM-DOC input text if TOM-DOC is to generate the output list for a printer belonging to the HP printer family.

1.6 Notational Conventions

A TOM-DOC command looks like this:

```
..ABC
```

The two dots are in the first two columns of the text area. The actual command follows directly after the dots. Although the commands can usually be written in uppercase and lowercase letters, it is best always to write in uppercase letters. This makes the commands easier to see and avoids errors, because the symbolic parameters, which are dealt with below, always have to be written in uppercase letters.

Some commands require additional information, for example the commands ..CE_op, ..CSP=i and ..HY=del. In this manual these stand for the following:

```
op  :      operand   (alphanumeric string)
del :      delimiter (single marker)
i   :      integer
n   :      digit
```

You may enter more than one command in a line, as long as you separate the commands with a semicolon. A command line could look like this:

```
..HY=#;..CSP=10;..CE_This is an example
```

The command line can be 256 lines in length, but please note the following:

Note

– Do not enter a blank "_" between the semicolon and the following command, otherwise TOM-DOC does not recognize it as a command.

– With the *commands ..*, ..HD, ..HDE, ..FT and ..FTE* the semicolon is taken to be part of the operand and so cannot be used to separate the commands.

Generally, we recommend you to input TOM-DOC commands in a clear and precise manner. This is best done by limiting the number of commands you enter in one line. In particular, if you wish to re-define symbols for your own personal use, it is recommended to write them in a block at the beginning of the text. Such a block could look like this:

```
..HY=#;..* hyphenation delimiter
..ULF=‘;..* marker for underscoring
..SF=$;..* special flag for umlauts
..FM=15;..* left-hand filing margin
```

You will be instructed in the proper use of these commands in the following sections. The following command is also important:

```
..*
```

It marks the start of a comment. You can use this function to annotate your document with explanations. The comments are not edited by TOM-DOC. In the following sections we shall be using the comment character to explain the TOM-DOC commands.

2 Working with TOM-DOC

Now you know the basic command structure of TOM-DOC. The commands are entered in the text and then it is just a matter of starting the TOM-DOC editing process. The following five sections explain how this functions.

There are two methods of text editing with TOM-DOC:

- from the TOM-TI editor
- or with the TOM-DOC phase.

To learn about text editing with the TOM-DOC phase, turn to page 31. First we shall explain how to start text editing from TOM-TI.

Once you have loaded the editor with "/EXEC $userid.TOM.TI" and have written some lines, you can start text editing by writing one of the following commands in the **command line**:

– *DOC[S] Short editing
– *DOCF Full information
– *DOCL Text editing with prompts
– *DOCE ENTER task

Editing starts when the DÜ1 key is pressed.

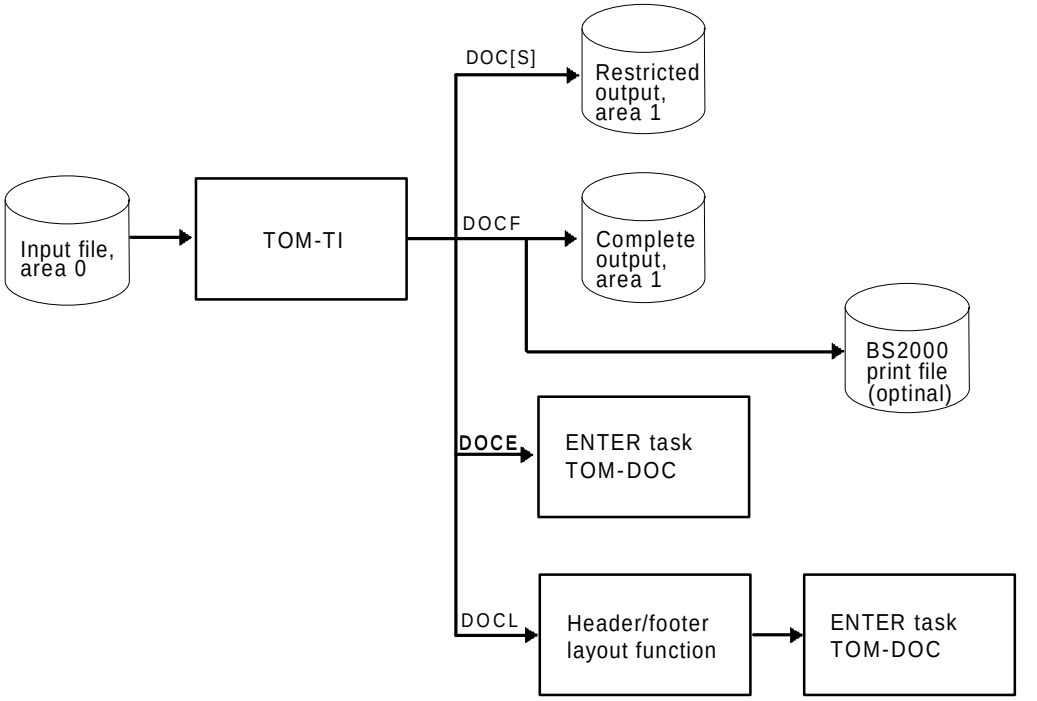


Figure 2-1: Interactive operation of TOM-DOC under TOM-TI

The examples below will show you the effects of the various commands. We shall take two TOM-DOC commands whose effects can be clearly seen:

```
..UL  (underline on)
..ULX (underline off)
```

These commands switch the underlining on and off.

Now that you are in the editor you can try out the effects of these editing commands by writing the following sentences:

```
Enter the command ..UL
..UL; to start underlining of a text.
..ULX; Enter the command ..ULX in the first
columns to stop underlining.
```

This text will be used a number of times later, so please store it.

Note:

So that the outputs of the examples can be shown better, they appear slightly differently in the manual. As in the previous manual, the output is edited for a 3350 or 3352 ND Laser Printer. However, you can now only use these printer types in BS2000 V10. They are supported in this version for the last time.

As of BS2000/OSD V1.0, these printer types are replaced by the HP laser printers 2090-2, 2140-2, 2240-2, 3351 and 3353.

Thus, if you want to go through the examples on a system running BS2000/OSD V1.0 or a later version, you have to insert the TOM-DOC command ..HP or ..DEV=HP at the beginning of your TOM-DOC input text so that the TOM-DOC output list is generated for the HP printer family.

If you use an HP printer with BS2000 V10, the same applies again of course. To generate a list for an HP printer, you have to enter the TOM-DOC command ..HP or ..DEV=HP.

2.1 *DOC[S] [ln1-ln2]: Short Editing

Start text editing by entering

```
*DOC [ln1-ln2] or
*DOCS [ln1-ln2]
```

in the command line. You can restrict editing to a specific block of lines by entering the first and last lines of the block for "ln1" and "ln2".

*DOC[S] gives a short output of the edited text. The output is "short" because not all the commands entered are interpreted. You can see this quite clearly when you look at what is in the work areas 1 and 2. *DOC[S] generates a maximum of two output files:

– edited text (in work area 1) and
– an error list (in work area 2)

If you have done everything correctly, now change over to work area 1. Enter 1 in the command line and press the DÜ1 key. Enter "--" to get to the top of the page. The following should now be displayed on your screen:

```
──────────────────────────────── HEADER ──────────────────────────────── PAGE 001 -
      Enter the command ..UL to start underlining of a text. Enter the
      command ..ULX in the first columns to stop underlining.
──────────────────────────────── FOOTER ──────────────────────────────── LINES LEFT 041 -









                                                                0001.00:001(0)
──────────────────────────────────────────────────────────────────────────────
LTG                                                                     TAST
```

Figure 2-2: The text edited by *DOC[S] in work area 1

As you can see, TOM-DOC has formatted the text. The page number is displayed in the header. The footer displays how many lines are left on this page. You cannot see, however, that part of the text is supposed to be underlined. This is the particular feature of *DOC[S]. You can quickly check the edited layout of the text, but do not see which segments of the text are to be emphasized.

Now deliberately generate an error message so that you can familiarize yourself with work area 2. In the next free line write:

```
..abc; This command does not exist.
```

Start text editing again with *DOC[S] and after a short time you will see the result. Just above the command line in work area 0 TOM-DOC displays the message:

```
% EDT0999    ***   ERRORS  DETECTED   ***
```

Now change over to work area 2 and you will see the following error message:

```
***  #DOC ERROR-FILE ***
I      6.0000 ILLEGAL COMMAND
***  END OF #DOC ERROR-FILE  ***
```

All the error messages in TOM-DOC have this structure. The type of error is indicated at the beginning of the line, followed by the line number and possibly the command the error message refers to. The actual text of the message comes next. The "I" at the beginning of the second line indicates that you have used an illegal command. Apart from "I", the uppercase letters "E","S" and "W" can also appear. The abbreviations stand for the following:

```
E   SYNTAX ERROR
I   ILLEGAL COMMAND
S   SEMANTIC ERROR
W   WARNING
```

The message **SYNTAX ERROR** appears if you make a syntax error, e.g. if you write several commands in one line without separating them with a semicolon.

A **SEMANTIC ERROR** is signalled if you assign an invalid operand to a command.

A **WARNING** is given if, for example, you are working in AS mode (without automatic line break) and your input line is longer than originally defined. The line is then truncated and an error message is displayed.

Command *DOC[S] thus provides a readable text, in work area 1, from which all the control characters have been removed. *DOC[S] is used for fast checking to see whether the text editing process is running error-free.

If you enter line numbers after the command *DOC[S], you restrict editing to the defined block. TOM-DOC will, however, continue to detect errors in the whole text. It is, therefore, possible that you give the command "*DOC[S] 10-20" and receive an error message referring to an error in the fifth line.

Note

Please note that *DOC[S] does not interpret all commands. You cannot see the following inputs in work area 1:

- Bold type
- Underlining
- Graphics conversions
- Copies of ..SW members
- HEADER/FOOTER conversions
- Revision handling
- Table of contents
- Keyword index

Reference variables are only replaced if they have been allocated before their use.

2.2 *DOCE: Enter Task

The *DOCE command initiates an enter task (the ’E’ stands for ’ENTER’). The edited text is printed out via SYSLST unless you have previously redirected SYSLST to a file.

TOM-DOC has a command with which you can prevent unwanted printer initiation. Right at the beginning of your document write:

```
..OPF=op (Output File)
```

You replace "op" with the name of the file to which the output is to be redirected. You can then check the edited text with EDT or the TOM-TI editor.

In the first line of the text file you enter the command:

```
..OPF=ABC
```

You then start text editing with

```
*DOCE
```

Now change to work area 9, for example, and display the edited file with "R[EAD]’ABC’". The following should now be displayed:

```
A


.                              ------------------------------
               Enter the command ..UL to start underlining of a text. Enter the
               command ..ULX in the first columns to stop underlining.









                                                                  0001.00:001(9)
LTG                                                                TAST
```

Figure 2-3: Output file created with *DOCE and read into TOM-TI

You may not be able to see the entire text on your screen. You could shift the text window by entering the command ">20", but then you would not see what is essential here. In contrast to the result obtained with *DOC[S] the header and footer have disappeared and a new line containing the underscore characters has been inserted. An

"A" in the first column at the top indicates that column 1 contains printer control characters. You can just make out one of these printer control characters in the first column of the newly inserted line. The dot "." (in fact it is the null character with the hexadecimal code ’00’) prevents line feed so that the text is underlined in the proper place. When generating for a different printer type, the printer control characters are different so what is displayed here is also different.

It is recommended to use the *DOCE command when formatting very long texts or when editing is terminated and the document is to be printed. In the latter case, however, the command "..OPF=op" must no longer appear in the text file. The completely edited document as generated by *DOCE is a good vehicle for making a final check of the layout. In our example the period at the end of the sentence will be underlined, which does not look very good. It would be better, therefore, to put the period after the ..ULX command. *DOCE also generates an error list appended to the end of the edited file.

Note

Direct printout with *DOCE is not recommended if dummy graphics or bold type occur in the text. Specific parameters have to be given for graphics interpretation (see section 2.8.5 Printing Graphics).

You can already do quite a lot with the commands *DOC[S] and *DOCE, but there are still two more useful commands which facilitate working with TOM-DOC. These are the commands *DOCF and *DOCL which combine the features of *DOC[S] and *DOCE.

2.3 *DOCF: Full Information

*DOCF gives you the full information ("F" for full). As soon as you initiate text editing with *DOCF, a maximum of two output files and one error list are generated:

1. In **work area 1** is the complete edited text, similar to when working with *DOCE. In other words this text contains bold type, underlining, graphics conversion, copies of ..SW members, header/footer conversion, revision characters, replaced reference variables, table of contents and keyword index.
2. In **work area 2** is an error list, just as when editing with *DOC[S]. If errors occur, work area 0 likewise displays the message

   ```
   %    EDT0999 *** ERRORS DETECTED ***
   ```

3. A **BS2000 output file** is created if you have defined one with *command* ..*OPF=op*.

Note

Please note that after text editing with *DOCF, TOM-DOC automatically resets the SYSLST system file to PRIMARY.

Although *DOCF has only been described briefly here, it will most likely become your standard TOM-DOC command, because of the wealth of information it provides you with.

2.4 *DOCL [NO[SCREEN]]: Text Editing with Prompts

It is only worthwhile using *DOCL [NO[SCREEN]] for text editing if you drop the "NOSCREEN" parameter. For *DOCL NOSCREEN is identical with *DOCE and also generates an error list at the end of the edited file. *DOCL without the NO[SCREEN] parameter displays a prompting screen immediately upon starting text editing, which provides you with auxiliary functions for defining your HEADER/FOOTER and your text layout. The prompting screen looks like this:

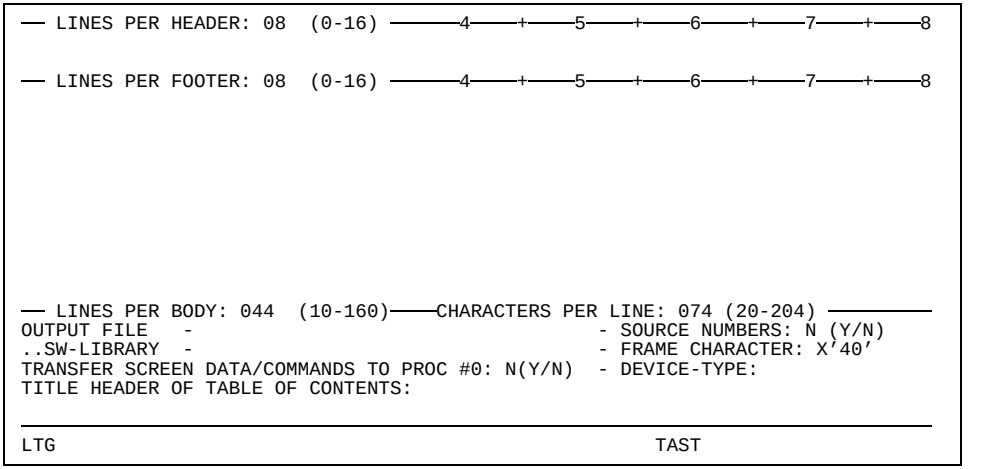


Figure 2-4: Layout of prompting screen with *DOCL [NO[SCREEN]]

2.4.1 Working with the Prompting Screen

This prompting screen helps you give your text a uniform layout and define the printer to be used. It is also a useful memory aid in case you have forgotten to make important layout specifications. Everything you enter on the prompting screen can alternatively be entered via TOM-DOC commands.

Let us now go through all the input fields of the screen, which will show you just how TOM-DOC sees a page. The following reduced sketch gives a clear picture of the page layout.

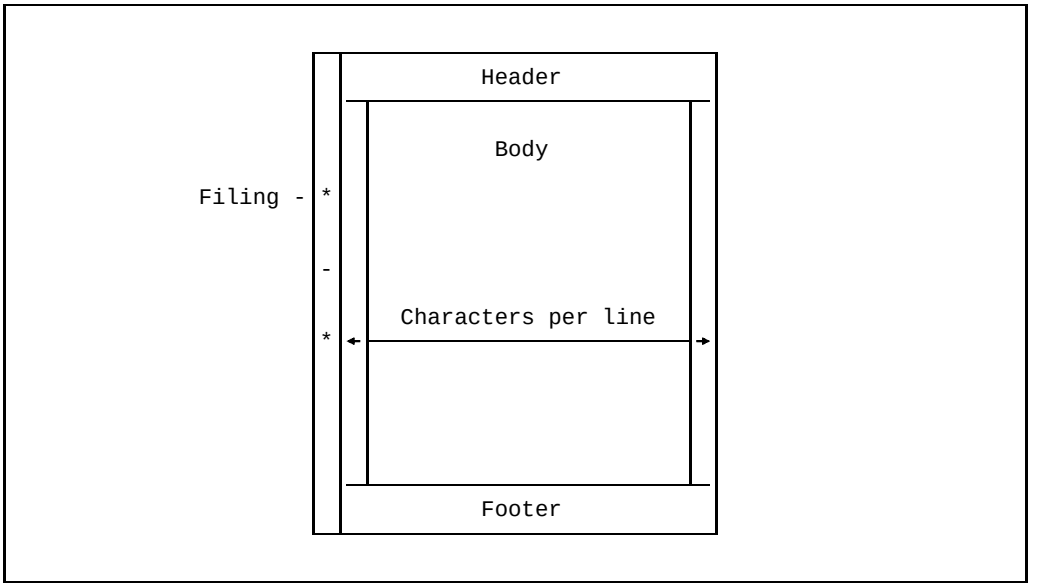


Figure 2-5: Page layout

LINES PER HEADER

LINES PER HEADER requests you to define the number of lines you wish to reserve for the header. The default value is 8 lines. If you wish to have more or fewer lines, place the cursor on the field where "08" is displayed and enter the required number of lines. The range of permissible values is given in parentheses to the right of the field. The number of lines per header must not exceed 16.

LINES PER FOOTER

Use the LINES PER FOOTER line in exactly the same way to define the number of lines you require for the footer.

Once you have done this, you can decide whether you want texts in the header and/or footer. For example, you might want to have the title of your document and the current date as a header and your name as a footer. In this case, simply enter the corresponding texts in the free fields underneath the line number input fields. The screen could then look like this:

```
── LINES PER HEADER: 02   (0-16) ──────4─────+─────5─────+─────6─────+─────7─────+─────8
This is the first line of the header with date &DATUM
This is the second line of the header.


─── LINES PER FOOTER: 02   (0-16)──────4─────+─────5─────+─────6─────+─────7─────+─────8
This is the first line of the footer.
And this is the second line of the footer.









─── LINES PER BODY: 044  (10-160)────CHARACTERS PER LINE: 074 (20-204) ──────────
OUTPUT FILE   -                                  - SOURCE NUMBERS: N (Y/N)
..SW-LIBRARY  -                                  - FRAME CHARACTER: X'40'
TRANSFER SCREEN DATA/COMMANDS TO PROC #0: N(Y/N)  - DEVICE-TYPE:
TITLE HEADER OF TABLE OF CONTENTS:

_____________________________________________________________________________
LTG                                                          TAST
```

Figure 2-6: Prompting screen with inputs

The date input "&DATUM" is a symbolic parameter. When editing, symbolic parameters are replaced by the current value. In this case, therefore, today's date appears in the edited text.

LINES PER BODY

The field to the right of LINES PER BODY is for input of the number of lines you require for the body of the text. You can select between 10 and 160 lines. The TOM-DOC default value is 44 lines. Together with the preset 16 lines for header and footer you have a total of 60 lines, which fit comfortably on an A4 page. If you do modify the header, footer and body line values, always check that your resulting text will still fit on the page.

CHARACTERS PER LINE

In the field CHARACTERS PER LINE you enter the number of characters you wish to have per line of text. The TOM-DOC default value is 74. You can, however, select between 20 and 204 characters per line.

OUTPUT FILE

The next input field is marked OUTPUT FILE. In this field you may enter the name of a file to which the edited text is to be written. If you do not enter a name in this field, the edited text is output to SYSLST, unless you have previously allocated output to a specific file with the ..*OPF command*.

Note

If you have already allocated an output file in the text and then define a new output file on the prompting screen, the instruction on the prompting screen takes priority. All other inputs in the text, however, take priority over those on the prompting screen.

If you want to redirect the text edited with *DOCL to the output file ’DATATEST’, then you write DATATEST next to OUTPUT FILE.

SOURCE NUMBERS

To the right of the output file field is a field marked SOURCE NUMBERS. The default input here is "N" for "No". If you enter "Y" for "Yes", the line numbers of your source text are written in the left margin of the edited text. These are normal serial numbers and not EDT line numbers.

..SW-LIBRARY

The next field, marked ..SW-LIBRARY, is very useful if you have long documents to edit. You use the *command* ..*SW=op* to copy externally stored text segments into your TOM-DOC input file. In this field, then, you input the name of the file from which you wish to retrieve the text segment. This may be the name of an FMS library or PLAM library.

FRAME CHARACTER

It is very often an advantage for the optical layout of a text to create a frame. You input your choice of frame character in the next input field marked FRAME CHARACTER. The default value is "X’40"'. The "X" indicates that for *DOCL the frame character must be in hexadecimal form. The "40" is the hexadecimal code for a blank ("_"). The character ’3A’ is very useful here. It looks like this: "|". This gives your document a continuous frame. Experiment with the frame and enter "3A" instead of the "40".
It is important that you use uppercase characters to do this.

TRANSFER SCREEN DATA/COMMANDS TO PROC #0

Now there are only two more input fields left. First you can decide whether the commands which have been input so far are to be written to work area 0 or 3. If you enter nothing (i.e. "N") in the field TRANSFER SCREEN DATA/COMMANDS TO PROC #0, the commands are written to work area 3. If the commands are written to work area 0 ("Y" in the input field), they can be integrated into the source text.

DEVICE-TYPE

A specific type of printer can be entered in the DEVICE-TYPE field. As default value the field is empty, which means editing is not printer-specific and no special control characters are used. If a specific printer is to be addressed, the device type required must be entered in the field. (For a list of the types which are presently supported, see the description of the ..DEV command.) For example, if you enter type 9022 in this field, control characters will be generated which the 9022 printer interprets for characters fonts, e.g. bold print. The ..DEV command generated is executed only once, i.e. if a ..DEV command exists in the TOM-DOC input file, this command is ignored. This function allows the user to generate output files for various printer types by changing the mask input, without having to change the TOM-DOC input file.

TITLE HEADER OF TABLE OF CONTENTS

The last input field is marked TITLE HEADER OF TABLE OF CONTENTS.

The usual entries here include "Table of contents", "Outline" or "Summary". This positions the table of contents at the beginning of the text. If you wish to have the table of contents elsewhere, you must define the position in the source text with the ..*TOC command*.
The completed prompting screen now looks like this:

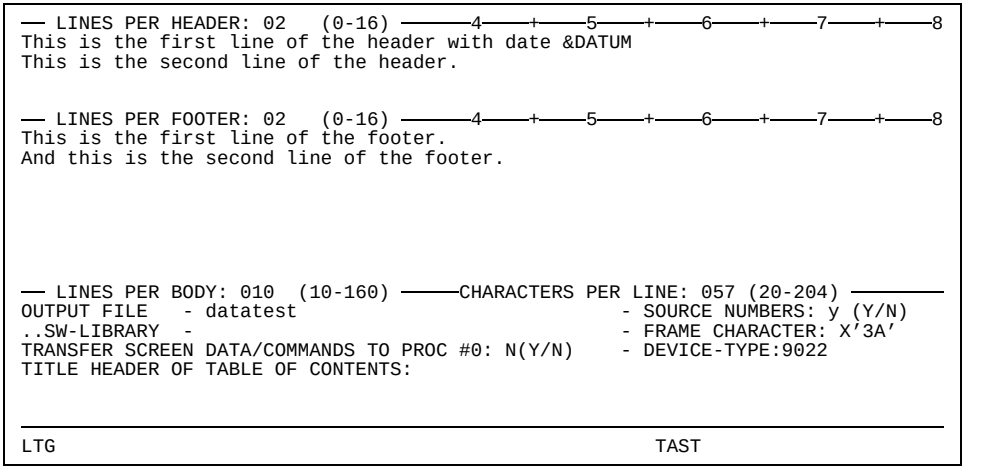


Figure 2-7: Completed prompting screen

Once you have completed the prompting screen, continue text editing by pressing the DÜ1 key.

First look at the edited text by changing over to the free work area 5 and reading in the text from the file "DATATEST". You see that the header and footer have been inserted and that the current date has been substituted. The line numbers of the source file are displayed in the left margin. If you go now to the end of your file, you see some error messages. One of these might inform you that some lines of the text have been truncated. The errors are easy to find and correct because the line numbers are given.

Note

– You cancel current action on the prompting screen by pressing the K1 key. This returns you to work area 0 and nothing is written in work areas 3 and 0.

– Files in which dummy graphic characters are used or which have been edited for an HP or LP65 printer cannot be output via SYSLST with the *DOCE or *DOCL command, only with a PRINT command (see section 2.8.5).

Although the commands are buffered in work area 3 to be used as presettings for the prompting screen, in case you start a new text editing run in the same session, it is, nevertheless, very useful to have generated commands in work area 0. This gives you access to the commands, even if the session has been interrupted. This is why it is best to enter "YES" in the prompting screen field "TRANSFER SCREEN DATA/COMMANDS TO PROC #0". Now switch to work area 0 and enter "- -" in the command line. This brings you back to the start of the text and the following is displayed:

```
..LPH=02
..LPF=02
..HD=1 'This is the first line of the header'
..HD=2 'This is the second line'
..FT=1 'This is the first line of the footer.'
..FT=2 'And this is the second line of the footer.'
..LPB=010
..CPL=040
..SRC=Y
..OPF=DATATEST
..FC=3A
..DEV=9022
Enter the command ..UL
..UL;to start underlining of a text.
..ULX;Enter the command ..ULX in the first columns to
stop underlining




                                                    0001.00:001(0)
LTG                                                 TAST
```

Figure 2-8: Commands in work area 1

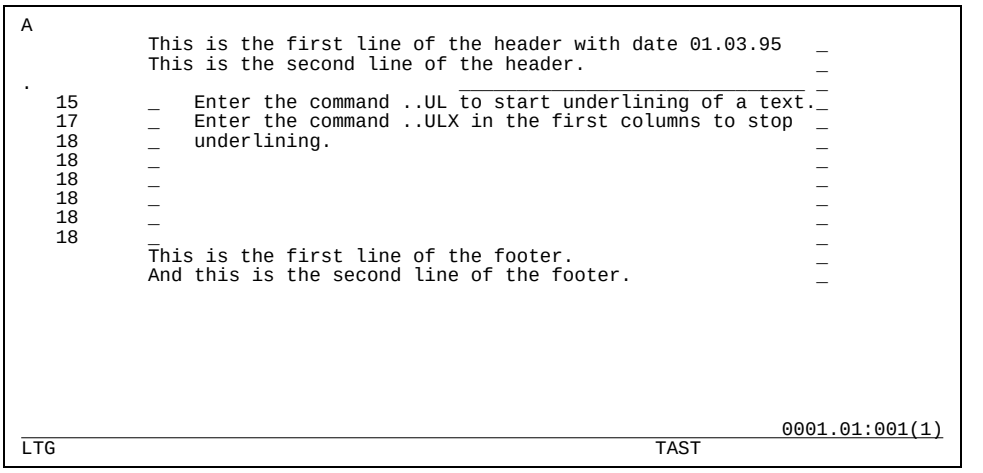
Here you see your text entries for header and footer next to the ..*HD and* ..*FT commands*. These commands are followed by a digit indicating the number of the line which contains the text enclosed in apostrophes. In the first two lines you see the ..*LPH and* ..*LPF commands* ("Lines Per Header" and "Lines Per Footer"). They indicate how many lines are reserved for the header and footer. The ..*LPB command* ("Lines Per Body") applies in the same way for the body of the text. You are already familiar with the ..OPF command which indicates the output file. The ..*SRC command* is set to "Y", which means that the line numbers of the source file are output. The .. *FC command* ("Frame Control character") is set to the hexadecimal '3A'. The ..*CPL command* ("Characters Per Line") speaks for itself: it indicates the number of characters per line. Then the ..DEV command specifies that control characters are to be generated for the 9022 printer. However, this command is only important if manipulation of character fonts is selected with the ..TD command at a later time.

If you respond to "TRANSFER SCREEN DATA/COMMANDS TO PROC #0 with "Y", you should ensure that the increment of the EDT lines is such that the generated text can

also be written before the first line in work area 0.

Thus, using the appropriate commands, you could also have made the necessary entries yourself.

If you start editing now with *DOCF, the following is displayed in work area 1:



2.4.2 Prompting Screen Inputs

Prompting screen	Meaning	Input rule	TOM-DOC command
LINES PER HEADER	Number of lines in header	0 ≤ n ≤ 16	..LPH
[free field]	Header input	Text	..HD
LINES PER FOOTER	Number of lines in footer	0 ≤ n ≤ 16	..LPF
[free field]	Footer input	Text	..FT
LINES PER BODY	Number of lines in text body	10 ≤ n ≤ 160	..LPB
CHARACTERS PER LINE	Number of characters/line	20 ≤ n ≤ 204	..CPL
OUTPUT FILE	Output file	File name	..OPF
SOURCE NUMBERS	Line numbers	Y/N	..SRC
..SW-LIBRARY	Input library	Library name	..SWL
FRAME CHARACTER	Type of frame character	Hexadecimal value	..FC
TRANSFER SCREEN DATA/ COMMANDS TO PROC #0	Copy screen data to work area 0	"Y" commands in work area 0 "N" commands in work area 3; valid for one session only	
DEVICE-TYPE	Output device	Printer type (up to 8 characters)	..DEV
TITLE HEADER OF TABLE OF CONTENTS	Title of table of contents	Text	..TOC

2.5 The TOM-DOC Phase

Up till now we have dealt with commands used to start TOM-DOC from TOM-TI. TOM-DOC, however, can be started as an independent program. First enter your text with EDT. Enter all the TOM-DOC commands that you need for layout in the text and then store it. The rest follows the BS2000 schema - first define your text file as an input file via SYSDTA and define an output file. Then start text editing as follows:

```
/EXEC [$userid.]TOM.DOC
```

The whole operation proceeds as follows:

```
/ASSIGN-SYSDTA TO-FILE=*LIBRARY-ELEMENT(LIB=lms-library,ELEM=element)
/ASSIGN-SYSLST TO-FILE=output_file
/START-PROGRAM FROM-FILE=[$userid.]TOM.DOC
/ASSIGN-SYSLST TO-FILE=*PRIMARY
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
```

Note

If you have already defined an output file with the ..OPF command, then there is no need to define another. After each TOM-DOC run always reset SYSLST to "PRIMARY" (even with ..OPF). This is not done automatically by TOM-DOC.

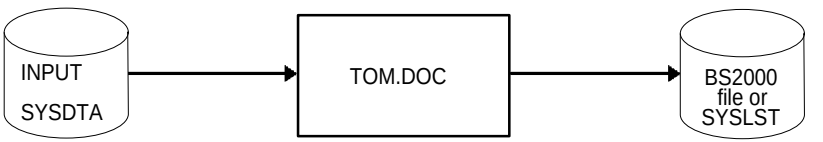


Figure 2-10: Direct calling of TOM-DOC from the BS2000 command level

You can, of course, call TOM-DOC within a procedure. This is particularly recommended when working with positional parameters. Here is an example of such a procedure:

```
/BEGIN-PROC LOGG=N,                                                    -
/           PAR=YES(PROC-PAR=(&OUT=                                    -
/                            ,&INPUT=                                  -
/                            ,&LIB=LMS.BIBLIOTHEK                      -
/                            ,&PRG=TOM.DOC)                            -
/           ,ESC-CHAR=C'&')
/ASSIGN-SYSDTA TO-FILE=*LIBRARY-ELEMENT(LIB=&LIB,ELEM=&INPUT)
/ASSIGN-SYSLST TO-FILE=&INPUT..&OUT
/START-PROGRAM FROM-FILE=&PRG
/ASSIGN-SYSLST TO-FILE=*PRIMARY
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/END-PROCEDURE
```

Whichever way you do it, the TOM-DOC phase produces a completely edited print file, just the same as commands *DOCE, *DOCF and *DOCL. The only exception is that error messages are not displayed on the screen. The error messages are appended to the end of the output file just as with *DOCE and *DOCL. When proofreading, always start by checking the end of the output file for possible errors.

Remark: Execution of Text Editing

During execution of text editing, the following temporary BS2000 files are created:

```
*DOCL.tsn.time.ENTER        Enter procedure
*DOCL.tsn.time.TFILE1       Input source
#DOC.tsn.time.TFILE2        Temporary output file
#DOC.tsn.time.TFILE3        File containing markings for
                            bold type and underlinings
```

The files ENTER and TFILE1 are created only for the enter task. The files TFILE2 and TFILE3 are not created with the *DOC[S] command. Normally, the files are erased at the end of the editing process. The files TFILE2 and TFILE3 are erased at the latest on termination of the user task (LOGOFF). File TFILE2 contains the entire edited text, except for the bold type and underlining characters and the reference variables not yet set. At the end of TFILE2 are the keyword index and the table of contents (unsorted). Errors are listed at the end of the text with source line references.

2.6 Symbolic Parameters and Their Use

You have already used one of these so-called symbolic parameters to enter the current date in the header of the test file. You wrote

```
&DATUM
```

and TOM-DOC then automatically inserted the correct date. Current date entry is a typical example of the use of symbolic parameters. Others include entry of the current page number, the current time of day, the current number of a footnote or other count variables. TOM-DOC supports these functions with various system variables. There are also reference variables for cross-referencing within a text.

TOM-DOC variables start with an ampersand (&) followed by the variable name, as we have already seen for the date. Variable names must always be written in uppercase.

2.6.1 System Variables

TOM-DOC has four types of system variables:

- page counters
- time variables
- count variables
- title headers

Page counters are:

&S1	Page counter up to 4 digits, right-justified
&S2	Page counter+1 up to 4 digits, right-justified
&S3	Page counter with end identifier + or - up to 5 digits, right-justified
&P1	Page counter up to 4 digits, left-justified
&P2	Page counter+1 up to 4 digits, left-justified

Time variables are:

&DATE	Date (US) format: mm/dd/yy
&DATUM	Date (European) format: dd.mm.yy
&TIME	Time of day format: hh:mm:ss

The count variable is:

&Tn	Count variable n=1...6 up to 3 digits, left-justifed

Title header variables are:

&LNi	Value of individual level i of the last title header, up to 3 digits, left-justified
&LCi	Full decimal classification of the last title header of level i, up to 40 digits
&LTi	Text of last title header of level i

The following example demonstrates how system variables are used. Copy the following input. The comments explain the procedure.

```
..LPH=2;..LPF=2;..* set length of header and footer
..LPB=10;..* set number of lines
..HD=1 'Page &S1'
..HD=2 'Date &DATUM'
..*header text
..FT=1 'Next page &S2'
..*footer text
..*the text starts here
This is a count using system variables:
..AS;..* 1:1 mode activated
..NL;..* new line
&T1) first
&T1) second
&T1) third
..AX;..* 1:1 mode deactivated
..NL;..* new line
That was a count. The time is now &TIME.
..NP;..* to go to the next page




                                                          0001.00:001(0)
LTG                                                  TAST
```

Figure 2-11: Input text with symbolic parameters

Note that the count variables all have the same number, in this case "&T1". If you were to write "&T2" or "&T3" instead, the count would not be continued, but a new count would be started.

Note

Please note that the count variable &Tn is always to be used for counts if you wish to make references to items in your count later on in the text. Do not use the *"..IN" command* in such cases.

The *..NP command* ensures that a new page is started. If the variable "&S2" is given in the header or footer and this following page is missing, the variable "&S2" is not interpreted. For example, TOM-DOC automatically gives you the option of indicating at the bottom of the page that there is a following page. However, this is the case only if &S2 is specified in the header/footer. If there is no following page, this note is omitted.

Now use *DOCF to start editing of this short example text. If you have made any errors, they will be listed in work area 2; the complete edited text is contained in work area 1. Your text (with blank lines removed) should then look like this:

```
A
          Page   1
          Date 01.03.95
             This is a count using system variables:

          1)   first
          2)   second
          3)   third

             That was a count. The time is now 17:05:27.


          Next page









                                                              0001.00:001(1)
LTG                                                    TAST
```

Figure 2-12: Edited text in work area 1 with interpreted system variables

Note

Use the ..SET command to re-set the initial values of the system variables.

2.6.2 Reference Variables

Now that you know all about system variables, you can go on to learn how to use the **reference variables**. As the name implies, these are used to refer to something, for example, to a specific page, chapter, or items in a count. Unlike the system variables which TOM-DOC sets automatically (though you do have the possibility of re-setting the values of the system variables), the reference variables have to be assigned specific system variables. You do this using the ..SET command.

Below is an example of the allocation of a system variable to a reference variable:

```
..SET &RFXXXX1(2)=&S1
```

This gives the reference variable &RFXXXX1(2) the value of the page it is on. Let us suppose that you have written a long text and in the last chapter you want to make a reference to a specific point in the first chapter. This is difficult, because you do not know where this point will be after printing. This is where the reference variable helps: just before or after the point in the first chapter you write

```
..SET &RFCHAP1=&S1.
```

Now, when you reach the place in the last chapter where you wish to make the reference, you write

```
"(_see_&RFCHAP1_)"
```

or

```
"(_see_&RFCHAP1(2))"
```

TOM-DOC subsequently inserts the correct page number.

Note

Please note the conventions for writing reference variables. The first places are defined by "&RF", then you can add a string of up to 5 characters which must be in UPPERCASE LETTERS. This helps you to identify the reference variable. The number of digits used for the variable may be put in parentheses at the end. The default value is 3, which should be sufficient for most applications.

Try out the procedure on the following example which is an extension of the previous one.

```
..LPH=2;..LPF=2;..* set length of header and footer
..HD=1 'Page &S1'
..HD=2 'Date &DATUM'
..FT=1 'Next page &S2'
..*footer text
..*the text starts here
This is a count using system variables:
..AS;..* 1:1 mode activated
..NL;..*new line
..SET &RFBSP=&S1;..* reference variable &RFBSP set
&T1) first
&T1) second
&T1) third
..AX;..* 1:1 mode deactivated
..NL;..* new line
That was a count. The time is now &TIME.
..NP;..*to go to the next page
On the next page I refer to the table on page
&RFBSP.



                                                          0001.00:001(0)
LTG                                                               TAST
```

Figure 2-13: Source text with allocation of system variable to reference variable

After editing with *DOCF, the following result is displayed in work area 1 (a number of blank lines have been removed):

```
A
          Date 02.03.95
             This is a count using system variables:

          1)   first
          2)   second
          3)   third

             That was a count. The time is now 11:15:29.



          Next page   2

A
          Page   2
          Date 02.03.95
             On the next page I refer to the table on page   1.



          Next page
                                                                0001.00:001(1)
LTG                                                      TAST
```

Figure 2-14: Edited text with interpreted reference variables

As you see, TOM-DOC inserted the page reference as required.

Note

Error messages referring to variables ("Unsatisfied Reference") always give the last source line as a reference, because their actual value is not necessarily defined until the end of the text.

Example

```
***  #DOC ERROR-FILE  ***
S    20.0000     UNSATISFIED REFERENCE  ->  RFBSP
***  END OF #DOC ERROR-FILE  ***
```

2.7 Automatic Word Division

When you start text editing, the automatic word division function is activated.

Note

The module library TOM.EDIT.OML contains the module DOCSILB. This module is linked to the software product SILB (BS2000). In addition to the DOCSILB module in the TOM.EDIT.OML library, which implements the automatic word division, the file COSY-SILT-AUSNAHMEWORTE must be located. The following sequence must be observed when searching for COSY-SILT-AUSNAHMEWORTE:

1. FILE assignment
2. User's own ID
3. TOM-DOC installation ID (please note: SHARE=YES)

The exception file, which is required for word division, can also be declared by entering a /FILE command with the LINK name CTDDSK. In this case, the file may have any name and may be stored under any user ID.

The results of the word division program will not always adhere to the German word division rules. The problems an automatic word division program has to surmount immediately become obvious when considering the very first rule in the DUDEN for word division in the German language:

```
Polysyllabic simple words and derivatives are to be divided
according to the spoken syllables which naturally occur when
the word is spoken slowly.
```

Rules derived from the spoken language cannot be used for automatic word division. Computer-supported word division must, therefore, take a different approach which is more formal and better adapted to the physical appearance of the written language.

Since the first simple rule does not always lead to sensible word divisions there are already rules for the written German language which deviate from the first rule in the DUDEN.

In the following we shall be considering a number of rules for word division in the German language, which will help to understand how the automatic word division function in TOM-DOC works. This is important, because all the rules have their exceptions. Take, for example, the rule that "s" and "t" should never be divided. Exceptions immediately become obvious when considering the division of words such as "Diens-tag" and "Donners-tag". Language is a living thing and it is, therefore, not possible to include all the exceptions in a program. TOM-DOC, however, gives you the possibility of maintaining your own excepted word file. Here you not only need to know the input formalities, but also the rules for word division in the German language.

Below is a summary of the word division rules which you can always refer to, followed by a function description of the word division program. Finally, it is explained how to deal with exceptions to the rules.

2.7.1 Word Division Rules

The word division rules can be split into the following categories:

- Division of consonants
- Division of vowels
- Division of lengthening letters
- Division of compound words

Division of consonants

The following rules apply for the division of consonants:

- A single consonant is placed on the following line (e.g. tre-ten).
- The last consonant of a sequence is placed on the following line (e.g. war-ten, Rit-ter).
- Suffixes starting with a vowel take the preceding consonant (e.g. Lüf-tung).

Division of vowels

The following rules apply for the division of vowels:

- A single vowel is not to be left standing alone. Words such as "Ader" or "Abend" cannot, therefore, be divided.
- Two vowels which are spoken as one are not to be divided. The same applies for diphthongs (e.g. Waa-ge, Eu-le). It is possible, however, to divide two vowels if they do not form a vocal entity (e.g. be-enden).

Division of lengthening letters

When dividing lengthening letters a difference must be made between the lengthening h and vowels which have a lengthening effect on the preceding vowel.

- The lengthening h is considered to be a consonant. The same rules apply, therefore, as for the division of consonants (e.g. nä-hen, ge-hen).
- If an "i" or an "e" has a lengthening function (above all in place names such as Troisdorf or Coesfeld), then the same rules apply as for two vowels forming a single vocal entity (e.g. Trois-dorf).

Division of compound words

Compound words are to be divided according to their constituent parts. Here, prefixes and suffixes are to be considered as constituent parts just as the words which form the compound word (e.g. Be-stand-teil).

In general it is best to avoid word divisions which could lead to misunderstandings; hence not "Spargel-der", but "Spar-gelder". The so-called joining s is not to be placed on the following line (e.g. Geschichts-bild).

2.7.2 Function Description of the Automatic Word Division Program

As you can see, the rules for word division cannot be applied in all possible cases. Ideally we would need to analyze all German words into their constituent syllables. This would not, however, be worth the effort considering the extent of the German language. It would also exclude spontaneous neologisms. Only a specific number of words, therefore, are explicitly stored and examined, and for the rest TOM-DOC restricts itself to analyzing words into characteristic and frequently occurring constituent parts. These include prefixes and suffixes.

Apart from prefixes and suffixes there are also words which include vowel sequences (diphthongs) which should not be divided. The TOM-DOC automatic word division program analyzes words to check whether they include such diphthongs and then replaces them with a special internal code. This phase of the process is known as **compression**. Compression gives the diphthongs and the letter combination "qu" priority over other letter combinations in the internal compression table. It is particularly important to recognize diphthongs where three vowels occur in sequence. In the TOM-DOC word division program the combinations "ai", "au", "ei" and "ie" have priority over other letter sequences.

After compression the words are naturally somewhat shorter. The automatic word division program then analyzes the words for suffixes. Words can always be divided in front of the suffix, though sometimes an extra letter has to be carried over with the suffix to make the division valid.

TOM-DOC goes back to the beginning of words for prefix analysis only after completion of compression and suffix analysis. Prefixes also often have characteristic letter combinations which render the prefix longer than originally expected.

In the final step after prefix and suffix division TOM-DOC analyzes the remaining word bodies for consonant combinations located between two vowels, which can be divided simply according to the rule. The result is a division within the word. Specific rules apply for special cases such as the joining s, ck, th, and sch amongst others.

It is not possible to divide a single letter from a word with the automatic word division program, as this is contrary to the rules of word division in the German language.

2.7.3 Excepted Words and Manual Word Division

If division errors occur in your text, then there are two ways available for correcting them:

– supplementing the excepted word file
– manual word division.

Let us first take a look at "manual" or "semi-automatic" word division. As the name already suggests, you have to mark your own word divisions by setting hyphenators with the TOM-DOC ..*HY=del command*.

With this command, you define a hyphenation delimiter, e.g.:

```
..HY=*
```

Now you can mark possible division points in a word with the character "*" as in the following example:

```
Aus*sichts*turm*wär*ter
```

The word "Aussichtsturmwärter" will now be divided as required at the marked points. The same character, however, can be used to prevent a word being divided. In such cases simply set the hyphenator in front of the word as in our example:

```
*Aussichtsturmwärter
```

Now the word "Aussichtsturmwärter" will not be divided, but carried over as a whole word if it does not fit on a line. This type of marking is recommended for hyphenated words which should not be divided (e.g. *TOM-DOC).

In addition to this type of manual word division you can also store words which you use very frequently in an excepted word list. The word division program always analyzes this list first, so that all user-specified word divisions are implemented during text editing. How to maintain your exception list is described in detail below.

2.7.4 Exception List

The exception list contains a collection of words with marked division points. If you wish to add new words to the exception list, carefully follow the instructions below to avoid any errors occurring.

1. The exception list should contain all foreign words which are not divided according to the general word division rules (e.g. Chir-urg).
2. The exception list should also contain words which are to be divided at certain points only for the sake of better comprehension (example: "be-inhalten" instead of "bein-halten").
3. The exception list should include words which have to have a third consonant added (e.g. voll-laufen).
4. If you are careful, you can drop endings of up to three characters from the exception words. This saves you storing all the inflections of an excepted word.

2.7.4.1 Hyphenators

To maintain your exception list you must note four hyphenators if you are writing Dutch or Flemish, but only three if you are writing texts in another European language. These hyphenators are used in the exception list where they fulfil specific tasks.

The first hyphenator is:

It marks a normal word division point. You can write it at the end of a syllable which, when necessary, will cause a word always to be divided after that syllable.

Example

Ap- divides Ap-fel, etc.
although wrongly divides Ap-orie (instead of Apo-rie).

The second hyphenator is:

It has a similar function to the "-" hyphenator with the difference that it is always written at the end of a syllable, whereas the "-" can be placed in the middle of a syllable combination. The "?" hyphenator clearly marks division points applicable for several words.

Example

mei-st? divides Mei-ster, mei-stens, etc. although wrongly divides mei-stbie-tend (instead of meist-bietend)

The third hyphenator is:

It is used relatively frequently and its very nature indicates that it adds something on. It is used to mark word division points where a consonant is doubled after the division.

Example

Schiff+ahrt divides correctly as Schiff-fahrt.

The fourth hyphenator is for use in Dutch and Flemish:

Its function is to delete a preceding character when a word is divided.

Example

taxie#tje is correctly divided as taxi-tje.

Note

It can be seen from the examples that **false application** of the hyphenators in the exception list could easily cause a lot of errors. Only use abbreviations when you are absolutely sure of their effect. It is better to write two words too many, than two letters too few.

2.7.4.2 Calling the Exception Table

Having noted the above you can now go ahead and maintain your own exception list. Enter the following command:

```
DO CTD.DO.EXD,<filename>
```

Transmit the command with the EM and DÜ1 keys. If you do not want to read in a BS2000 file with excepted words, enter a dummy name for the filename, e.g. "DO CTD.DO.EXD,X". Otherwise note that the input file must be of the ISAM type. We will see later on how to transfer existing files with excepted words.

On calling the exception list you will be asked for the prefix of the excepted word file. This helps to identify the individual excepted word files because they must all end in "COSY-SILT-AUSNAHMEWORTE". Enter the prefix required and confirm with the DÜ1 key. Now there are three new files in your directory:

- the file which you specified with the "DO CTD.DO.EXD" command, e.g. "DUMMY";
- a print file with the dummy name and suffix DR, e.g. "DUMMY.DR";
- the exception table in the form: "<prefix>.COSY-SILT-AUSNAHMEWORTE".

2.7.4.3 The Screen Mask

If you have written and transmitted your command correctly, the following screen mask is displayed:

```
Aktion: E              Wortanfang: A                        Sprache: 01
(Action)               (First letters)                      (Language)
                                                                    (1)
Erfassen (E) Trennen (T) Blättern(+/-) Ausdrucken (P/PA) Einlesen (RF) Halt(H)
(Enter)      (Divide)    (Page)         (Print)           (Transfer)    (Halt)

01 = DEUTSCH (GERMAN)
02 = ENGLISCH (ENGLISH)
03 = FRANZÖSISCH (FRENCH)
05 = ITALIENISCH (ITALIAN)
09 = SPANISCH (SPANISH)                                        (2)
11 = NIEDERLÄNDISCH (DUTCH)
12 = NEU-GRIECHISCH (MODERN GREEK)
13 = PORTUGIESISCH (PORTUGUESE)
14 = DÄNISCH (DANISH)
15 = SCHWEDISCH (SWEDISH)

                                                                    (3)
                                                 Ausnahmedatei
                                               (Exception file)

LTG                                                  TAST
```

Figure 2-15: Input mask for maintaining the excepted word file of the automatic word division program

The screen mask comprises:

(1) Mask head with title and header
(2) Data part with text entry lines
(3) Mask foot with error line.

Mask head with title and header

Aktion (Action)

The "Aktion" field is reserved for a maximum of 2 characters. In this field you can enter any of the characters proposed in the second line of the mask head, thus: E, T, H, P, PA and RF, as well as "+" and "-". The characters stand for:

E	Enter
T	Test division
H	Halt (dialog terminated)
P	Print part of the exception list
PA	Print all of the exception list
RF	Transfer of existing excepted word file
+/-	Paging in the excepted word file

As you see, when the mask is called the "Aktion" field is preset with "E".

Wortanfang (First letters)

In the "Wortanfang" field you enter the first letter(s) of the excepted word. This allows you to access any point in the excepted word list and enter or read an excepted word. Precision goes up to the first three letters of a word.

Sprache (Language)

The "Sprache" field is for selecting one of the languages available. This field is preset with your national language, in this case "01" for German. Simply enter a different number to select another language, e.g. "02" for English or "09" for Spanish.

The word division program functions for the following languages:

- German
- English
- French
- Italian
- Spanish
- Dutch
- Portuguese
- Greek (modern)
- Danish
- Swedish

The entries in the mask head determine what is to be done after the exceptions have been stored.

Data part with text entry lines

New entries in the excepted word file can be made anywhere in the center part of the mask. Just make sure that the individual words are separated by a blank "_". The words can be entered in upper or lowercase letters. They will be automatically changed into uppercase letters as this has no influence on the division points. The excepted words are sorted and entered as per the national code table selected.

Press the DÜ1 key when you have entered your list of exceptions. Up to 15 words are then displayed in the data part, all starting at the beginning of the line. You can page by entering "+" or "-".

Mask foot with error line

Invalid inputs in the mask are indicated in the mask foot. The error messages indicate, for example, invalid characters or unknown actions. You can correct the errors immediately in the mask head.

2.7.4.4 Practical Example of the Exception Program

When you have the first mask displayed on the screen you can either accept the parameters preset in the mask head or select different parameters (a different language or other first letters). Whatever the case you must then confirm the selection by pressing the DÜ1 key.

The following screen then appears:

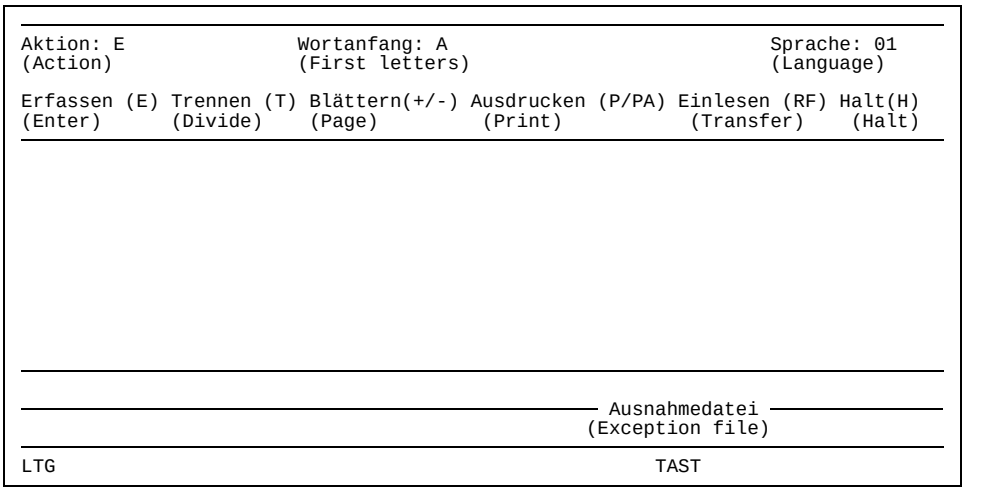


Figure 2-16: Empty input screen for the excepted word file

Now you can enter words as you please in the data part of the screen, i.e. where the language codes where listed before. Try it out yourself by entering the following words in the table by way of example: "Amerika, Afghanistan, Angestellter, Arbeitsamt, Ausgaben, altrussisch and Augen-arzt". It is important here to remember to enter a blank "-" between each word. If you set a hyphenator to mark a division point in a word, as in the word "Augen-arzt", then the word will be divided at that point only.

Your screen should now look like this:

```
 Aktion: T              Wortanfang: A                          Sprache: 01
(Action)              (First letters)                        (Language)

Erfassen (E) Trennen (T) Blättern(+/-) Ausdrucken (P/PA) Einlesen (RF) Halt(H)
(Enter)      (Divide)    (Page)         (Print)           (Transfer)    (Halt)

AMERIKA AFGHANISTAN ANGESTELLTER
ARBEITSAMT AUS-GABEN ALTRUSSISCH
AUGEN-ARZT

                                                  Ausnahmedatei
                                                (Exception file)
LTG                                                    TAST
```

Figure 2-17: Example of entries for the excepted word file

Now we want to know how the system proposes to divide these words. Enter "T" for division in the action field and then press the DÜ1 key. The following should then appear on your screen:

```
Aktion: ?              Wortanfang: A                          Sprache: 01
(Action)               (First letters)                        (Language)

Erfassen (E) Trennen (T) Blättern(+/-) Ausdrucken (P/PA) Einlesen (RF) Halt(H)
(Enter)      (Divide)    (Page)         (Print)           (Transfer)    (Halt)
AUGEN-ARZT
AUS-GABEN
AF-GHA-NI-STAN
AL-TRUS-SISCH
AMERIKA
AN-GE-STELL-TER
AR-BEITS-AMT

                                                  Ausnahmedatei
                                                (Exception file)
LTG                                                      TAST
```

Figure 2-18: Divisions proposed by the word division program

In the action field there is now a question mark. This means that TOM-DOC wants to know whether you accept the proposed word divisions. TOM-DOC has divided all the words according to its internal rules, except for those words which you entered with a hyphenator, i.e. "Augen-arzt" and "Aus-gaben". The word AMERIKA was not divided. Change AMERIKA to
AME-RI-KA.

Now you can select the words which have been correctly divided and delete them by overwriting them with blanks "_". In the case of the wrongly divided word "altrussisch" insert a hyphenator at the correct point: "ALT-RUS-SISCH". The excepted words are stored by pressing the DÜ1 key.

Note

If you wish to enter excepted words beginning with different letters, it is not necessary to call a new mask for each first letter. Simply delete the "Wortanfang" (First letters) field in the mask head. TOM-DOC will then display a blank mask in which you can enter all the desired excepted words. TOM-DOC will take care of the sorting process.

2.7.4.5 Printout

If you want to print out the excepted word list, you have to first prepare it for printing. This is done by entering "P" or "PA" in the action field.

If you enter "P", only those excepted words are printed whose first letter is given in the mask head. If you enter "PA", the system prints out a list of all the excepted words in one language.

Printout of the excepted word list is initiated when the program is terminated by entering "H" in the action field. Termination of the program is indicated by display of the message:

```
***ENDE DIALOG AUSNAHMEDATEI***
```

2.7.4.6 Transfer of Existing Files

If you already have an exception file for word divisions, then you can transfer it by entering the RF parameter in the action field. At the start of the program you must enter the name of the BS2000 file as filename. This file must be of the ISAM type. You must enter the language code in this external file before the first excepted word. Your list should then look like this:

```
01
AUS-NAH-ME
...
ZU-GEH-FRAU
```

Generally the structure of the table is as follows:

```
nn
Excepted word
    .
    .
    .
Excepted word
```

In each line there is just one excepted word, the first letter of which must be in the first column. Enter "RF" in the action field in order to integrate this external excepted word file, once processed, into your TOM-DOC word division program. Confirm integration by pressing the DÜ1 key.

2.8 The TOM-DOC Graphic Module

2.8.1 Introduction

The TOM-DOC graphic module enables you to insert graphics into your text, ranging from simple frames and brackets to complex diagrams. You just write dummy graphic characters in your text and TOM-DOC converts them during the editing process. Certain input conventions must be observed. For example, you must indicate to TOM-DOC where graphics conversion is to start. This you do with the command

```
..COD (graphics mode on)
```

You indicate where graphics conversion is to stop with the command

```
..COX (graphics mode off)
```

In the following you will first learn all the dummy graphic characters recognized by TOM-DOC. Then you will be made familiar with the rules TOM-DOC observes for graphics conversion. In the next section you will find a step-by-step example of how to integrate graphics into a document.

Note

- It is recommended to create graphics in AS mode only.
- Dummy graphic characters can be changed at will with the ..*CH command*.
- For graphics in the header and footer use the ..*CHF=Y command*.
- Files in which dummy graphic characters are used cannot be output via SYSLST with a *DOCE or *DOCL command, only with a PRINT command (see section 2.8.5, chapter 5 and the appendix).

2.8.2 Dummy Graphic Characters

TOM-DOC recognizes the following characters as dummy graphic characters:

-	=	%	I	!	’	‘	+	*	#	(	)	v	?	^	<	>

When graphics conversion is activated, these characters are converted to their assigned graphic characters, but this only happens when the characters are in a specific context. For example, a single "!", is only converted to a vertical line when there is another dummy graphic character in its immediate vicinity. Normally, a dummy graphic character stands for itself. You will find examples below that illustrate the conversion rules.

If you need a dummy graphic character in the text, you can use the command `..CH=char1>char2` to specify a different character that is to be converted to the appropriate graphic character. This applies until it is explicitly reset by means of `..CH` without operands.

Example

```
..CH=/>!
```

The character "/" instead of "!" is converted to a vertical line. The character "!" stands for itself, regardless of the context.

If you use a dummy graphic character as a control character - for bold type, as an index flag or as a metablank, for example - and you use this control character in a graphic, this may have unwanted results. Therefore, whenever possible, do not use dummy graphic characters as control characters.

2.8.3 Simple Conversion Rules

If simple dummy graphic characters are input at least twice horizontally or vertically adjacent, they are converted uniquely into horizontal or vertical lines.

Input

```
--  ----    ! %  ! %  I    I    ====  ==   Ix  Ix  I1  I1  xI_ II_
            ! %  ! %  I    I              Iy  Iy  I2  I2  yI_ II_
%%  %%%%        ! %  I                        Iz      I3
```

Output

As you can see, the "I" is not converted if it occurs twice vertically adjacent in a string. The "I" is, however, converted if followed by another "I" or if it is the last character in a string. Three vertically adjacent "I"s are always converted. To avoid this you should insert a word or at least a metablank at the appropriate point in your text.

Note

Horizontal and vertical broken lines are represented by means of the dummy graphic character "%". During conversion the horizontal "%" characters take priority over the vertical "%" characters. If two broken lines cross, mark the point of intersection with a "+" character.

The assignment of other characters is no longer unique but still straightforward. Simple slanting lines are represented as follows by means of the " ‘ " and " ’ " characters.

Input

```
  ‘‘    ‘   ‘   ’   ’   ’’
    ‘‘   ‘   ‘   ’   ’   ’’
```

Output

2.8.4 Complex Conversion Rules

This section explains how TOM-DOC supports the creation of complex graphics such as curves, pictograms, arrows, syntax diagrams and eccentric lines.

The main obstacle in creating complex graphics is that the laser printer character set of no more than 255 defined characters does not suffice when lines of different types and different gradients are to converge at a point. TOM-DOC avoids this problem by taking advantage, in graphics mode, of the ND laser printer technique of printing two lines one on top of the other. The HP laser printer draws on its larger reserve of character sets to print graphics. In either case, you have simply to input the specific characters in accordance with the TOM-DOC graphics rules.

To make things clearer in the following description we shall now define the concepts of Z and ES lines. In fact, they almost speak for themselves, once the abbreviation is written out in full.

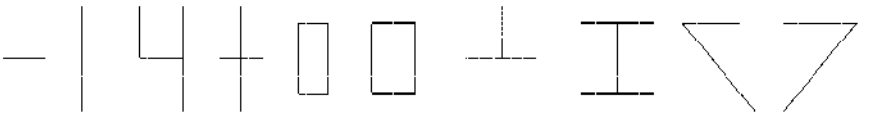
1. Z lines (centric lines) are lines which pass through the center of a rectangle and connect diagonally opposed corners or middle points of opposing sides. Z lines can be printed as normal, broken or bold-type lines.
2. ES lines (eccentric slanted lines) are lines which do not pass through the center of a rectangle.

Two important characters for the graphic module are the "+" character and the "*" character. The "*" is used as a joiner to mark the point of intersection of two centric lines. The "+" character, on the other hand, is a universal joiner which, depending on the environment, can join centric and/or eccentric lines. This is clearly demonstrated in the examples below:

Input

```
     !   !  !   !   +-+  +==+     %     ==+==  +---   -----+
-+-  +   +--+  -+-  ! !  I  I   %%+%%     !     ''       ''
     !       !   !   +-+  +==+           ==+==     ''  ''
```

Output



Each dummy graphic character links up with the others in specific directions in accordance with set conventions. In this example you see the function of the "+" character which can set up links in all possible directions.

Note

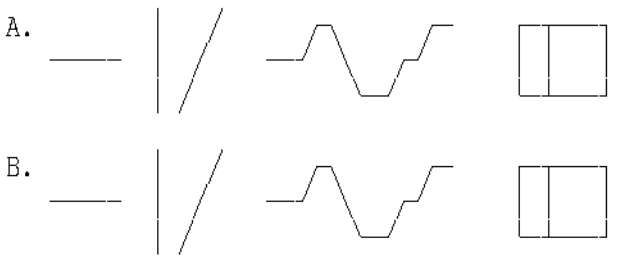
In order to avoid misinterpretations, the "+" character should only be used where necessary.

Input

```
A.        !   ’      ++      +-    +-+-+-+
   -+++-  *  +    -++  +   ++      + +   +
          ! ’           +++        +-+-+-+

B.        !   ’      ++      +-    +-+---+
   -----  !  ’    --+  +   ++      ! !   !
          ! ’          +-+        +-+---+
```

Output



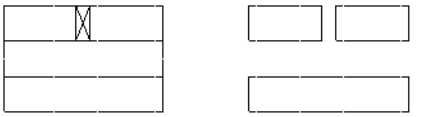
The results of the graphic conversion are identical.

There is one possible case where you have two adjacent "+" characters but you do not want them joined in one line. This problem is solved with the "*" character. This character has the quality of joining all characters except "+" characters. It may not, however, be used with ES lines.

Input

```
+----++----+     +----+*----+
+----++----+     +----+*----+
+----------+     *----------*
+----------+     *----------*
```

Output

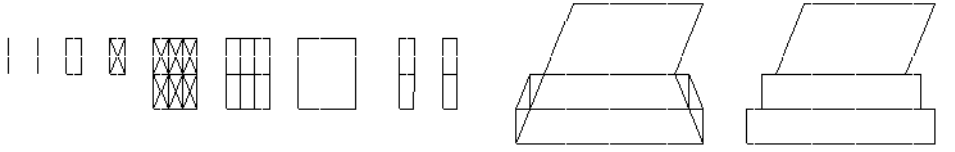


Another very useful character is the "#". Like the "+" and "*" characters the "#" character is also a joiner, but only for vertical or horizontal links, as can be seen below:

Input

```
                                    *--------*      +--------+
+ # ## ++ ++++ #### #---#  ## ##    ,        ,      ,        ,
+ # ## ++ ++++ #### !   !  ++ **   **--------**    #+--------+#
          ++++ #### #---#  ## ##  **----------**  ##----------##
                                  *------------*  +------------+
```

Output

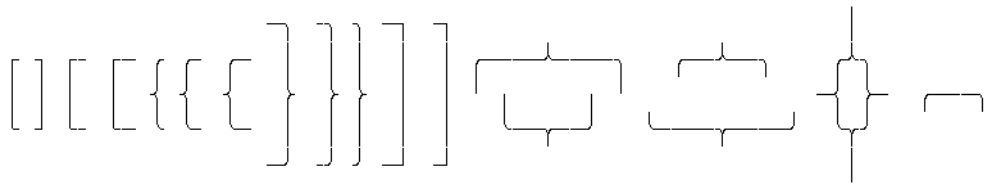


Brackets and braces can be included in your text using specific dummy graphic characters as illustrated in the following figure:

Input

```
                  -) +) ) -+ ++                              !
< > ++ +- ( (+ (-  !  ! !  !  ! (----^----)   (--^--)    (^)
! ! !  !  < <  <   >  > >  !  ! + +     + +             -< >-  (---)
< > ++ +- ( (+ (-  !  ! !  !  !   (--v--)   (----v----)  (v)
                  -) +) ) -+ ++                              !
```

Output



Note

– In the case of vertical brackets and braces opening to the left or right, the length of the ends can be specified with the "+" or "-" characters.

– The representations of brackets

```
<     >
! and !
<     >
```

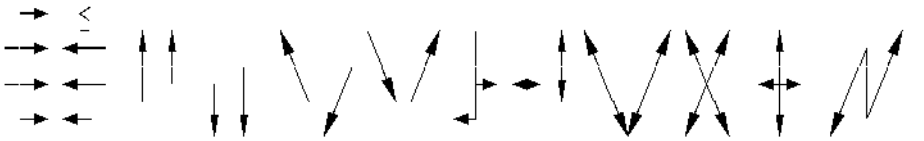
are special cases; the length of the ends opening to the left or right is invariable.

Arrow points can be represented for all Z lines using the characters "<", ">", "^" and "v" :

Input

```
 => <=
==> <==  ^ ^       ^     ‘   ^  !     ^ ^    ^ ^ ^   ^     + ^
--> <--  ! I  I !   ‘  ’  v ’   +> <> v  ‘  ’   +   <+>   ’!’
 -> <-       v v     v        <+         vv   v v   v   v +
```

Output



Note

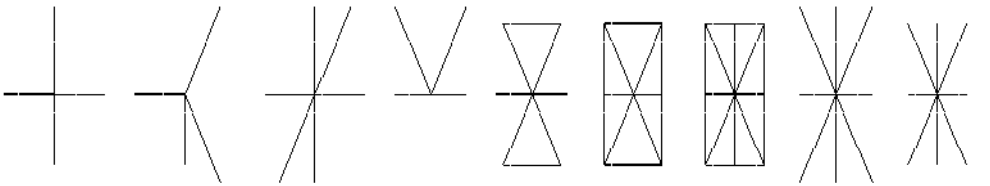
- A "v" within a string of alphabetical or numerical characters is not interpreted as a dummy graphic character to be converted unless it is the last character of the string or followed by another "v".
- Short arrows in bold type pointing to the left cannot be represented (see example) because "<=" is the input convention for "less than or equal to".

Complex graphics with centric lines can contain line elements of no more than two of the three classes of Z lines, i.e. normal font, broken, bold type. All combinations of horizontal, vertical and diagonal lines are, therefore, possible.

Input

```
   !          '      ! '   '   '  +---+  +===+  +-+-+  ' ! '  + + +
   !         '       !'     ' '    ' '   I' 'I  !'I'!   '!'    '!'
===+---  ===+     ---+---  --+--  ==+==  *-+-*  +=+=+  --+--  +-+-+
   I        I'      '!             ' '   I' 'I  !'I'!   '!'    '!'
   I        I '    ' !            +---+  *===+  +-+-+  ' ! '  + + +
```

Output

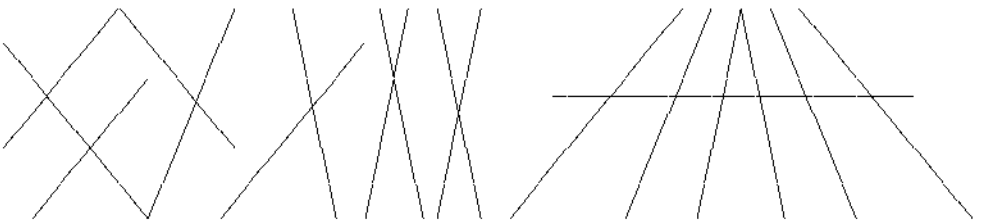


Eccentric node points are created with "+" characters by overlaying two ES lines or an ES line and a Z line. No more than two lines are allowed to intersect. A "+" character must be entered for each point of intersection.

Input

```
     ''''     '    '     ''  ' '          '' ' '' ' ''
''  ''    ''  '    '  '' '' ' '         ''  '  ''  '  ''
  ++    ''  '+       +'   ''   +      ---++---+--+--+--+---++--
''  ''''    ' ''   '''    ''   +       ''    '   '  '   '    ''
    ''''   '     ''   '  '  ' ' '    ''     '   '    '   '     ''
  ''    '''    ''     '  '  ' ' '  ''      '    '    '    '      ''
```

Output



Note

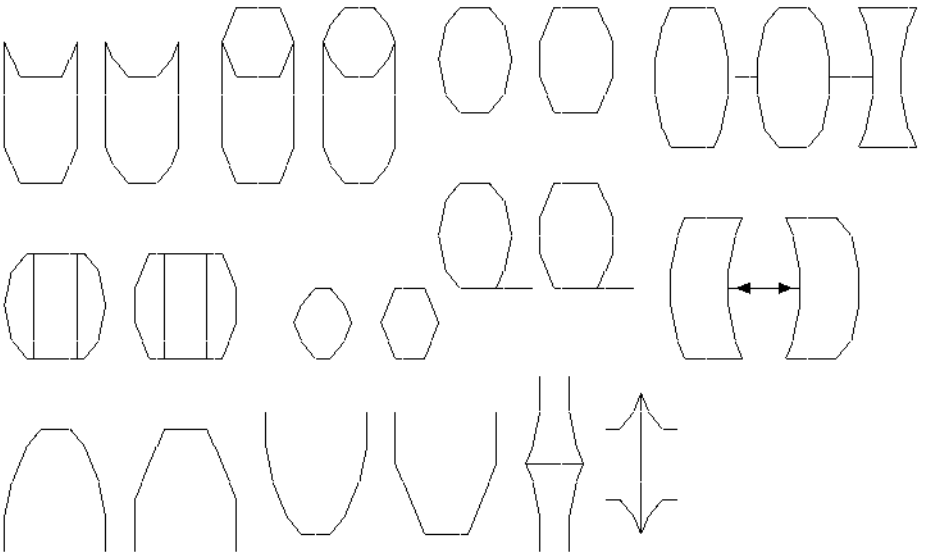
If the points of intersection with other lines marked by a "+" character on a line are too close together, the TOM-DOC graphic module is not always able to interpret the connections unequivocally.

As we have already seen, the characters " ’ " and " ‘ " are converted to slanting lines. However, in certain cases they are used to replace the "+" character, which results in a **flattened curve**.

Input

```
                +--+   ’--‘    ’--‘   +--+    +--+   ’--‘  +---+
+    + +    +  +    + +    +  ’    ‘ +    +  ’    ‘ ’    ‘  ‘ ’
!+--+! !‘--’!  !+--+! !‘--’!  ’    ‘ +    +  !    !-+    +--+ !
!    ! !    !  !    ! !    !   ‘--’   +--+   ’    ‘ ’    ’  ‘ ‘
+    + +    +  +    + +    +                  +--+   ‘--’  +---+
 +--+   ‘--’    +--+   ‘--’    ’--‘   +--+
                          ’    ‘ +    +    +---+  +---‘
 ’+--+‘   ++--++             ’    ‘ +    +   ’   ’    ‘   ‘
’ !  ! ‘ + !  ! +    ’-‘   +-+ ‘--+-- +--+--  !   +<-->+   !
‘ !  ! ’ + !  ! +   +   + +   +               ’   ’    ’   ’
 ‘+--+’   ++--++     ‘-’   +-+                 +---+  +---’
                                  ! !    +
  ’--‘     +--+   !      ! !      !  ’ ’  -’!‘-
 ’    ‘   +    +  ’      ’ +      + +---+   !
’      ‘ +      +  ’    ’   +    +   ‘ ’  -‘!’-
!      ! !      !   ‘--’     +--+    ! !    +
```

Output



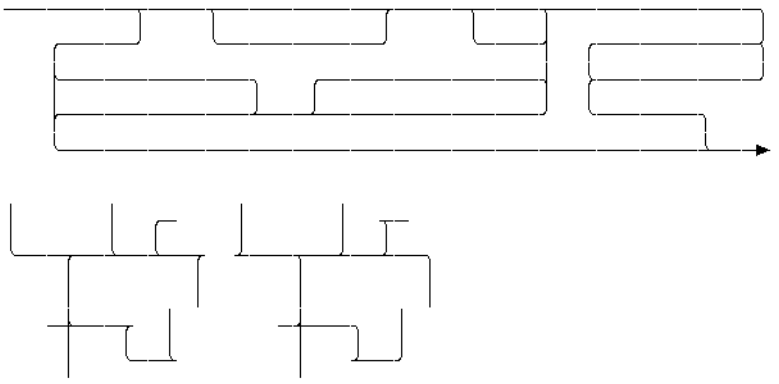
Rounded branches used in syntax diagrams, for example, are generated via the characters "(", ")" and "?". The "?" character creates downward branches in a vertical sequence, whereas the parentheses create upward branches.

Input

```
---------)----)-----------(-----)----)--------------)
   (-----)    (-----------)     (----)  (-----------<
   (-------------)   (---------------?  >-----------)
   ?-------------)---)---------------)  (-------)
   (-------------------------------------------(--->

!      !  (-    !      !  )-
(---(--(--(--(  )---)--)--)--)
    !        !      !        !
   -(---(  !       -)---)  !
    !   (--(        !   )--)
```

Output

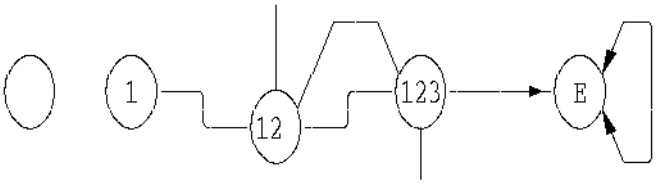


The parentheses and the "-" character can also be used to create **circles** whose size is unchangeable and which can accommodate up to three alphanumeric characters.

Input

```
                    !   +--+                 +-)
 (-)    (-)         !  ’    ‘(-)         (-)v  !
(   )  ( 1 )--)    (-)’  (--(123)----->( E )  !
 (-)    (-)   (--(12 )--)   (-)         (-)^  !
                   (-)         !               +-)
```

Output



Arithmetic special characters can be inserted in your text using dummy graphic characters. The arithmetic special characters are generated by overlaying standard characters according to the following rules:

Input

```
<=        >=       =/       =^       +-
```

Output

```
≤         ≥        ≠        ≙        ±
```

Note

You can use the ..*CH=del>del command* to re-define all the dummy graphic characters to suit your requirements.

2.8.5 Printing Graphics

The basic commands for printing graphics are as follows:

```
/PRINT file,CHAR=(DE),SPACE=E,IMAGE=POST
/PRINT file,CHAR=(DL),SPACE=E,IMAGE=POST
/PRINT file,CHAR=(DI),SPACE=E,IMAGE=POST
```

If you have bold type in your text, you must use one of the following commands:

```
/PRINT file,FORM=...,CONTROL=PHYS,CHARS=(DE,DF),IMAGE=POST
/PRINT file,FORM=...,CONTROL=PHYS,CHARS=(DL,DM),IMAGE=POS
/PRINT file,FORM=...,CONTROL=PHYS,CHARS=(DI,DJ),IMAGE=POST
```

If you are using an HP laser printer, your text file must include the ..*HP command*. The print commands are then as follows:

```
/PRINT file,CONTROL=PHYS,CHARS=(DE1,DE2,DE3,DE4),IMAGE=POST,FORM=...,DEV=...
/PRINT file,CONTROL=PHYS,CHARS=(DL1,DL2,DL3,DL4),IMAGE=POST,FORM=...,DEV=...
/PRINT file,CONTROL=PHYS,CHARS=(DI1,DI2,DI3,DI4),IMAGE=POST,FORM=...,DEV=...
```

If you require bold type, you use the ..CSP command in your text file to define the character spacing to coincide with the character set defined by the CHARS parameter. The character sets are:

DE : large character set (10 characters per inch)
DI : medium-sized character set (12 characters per inch)
DL : small character set (15 characters per inch)

The ..*CSP command* is always needed if you are using an HP laser printer.

Under FORM you must define the paper format used by your printer. Check with your computer center, if you are not sure.

Note

All output files edited by TOM-DOC can be printed out on BS2000 systems using the /PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (using the product RSO) and printed out from there.

As of SPOOL V3.0, it is still possible with the BS2000 product DPRINT to print out the edited TOM-DOC output files on remote printers (e.g. including printers connected to SINIX systems) without RSO and the printer emulation on SINIX. You will find relevant information on the products and their versions required on the BS2000 systems supported by TOM-DOC, the BS2000 commands, the correct selection of devices and paper sizes, the use of suitable character sets, the control of character and line spacing, and the monitoring of SPOOLOUT jobs in chapter 5, "Printing with TOM-DOC". This contains detailed information that helps you to use the extensive printing options for TOM-DOC files efficiently.

2.9 Summary

1. You call TOM-DOC either as a phase by means of /EXEC or /START-PROGRAM or in TOM-TI by means of the command *DOC[S], *DOCE, *DOCF or *DOCL. Before you enter text, you should enter the TOM-TI commands "LOW ON" and "CODE ON" in order to be able to enter lowercase characters and German special characters. For information on using the EDT command CODE, please refer to the EDT manual.
2. TOM-DOC commands are always preceded by two dots in the first two columns. Several commands can be written in one line if they are separated by a semicolon (";"). There are restrictions, however, when entering text.
3. You start text editing with TOM-DOC by means of the command *DOC[S], *DOCF, *DOCL or *DOCE. The effects of these commands are summarized in the following table:

Command	Function	Key to use
*DOC[S] ln1-ln2	Restricted interactive text editing	DÜ1
*DOCE	Editing of a complete output text in batch mode without auxiliary functions	DÜ1
*DOCF	Editing of a complete output text in interactive mode	DÜ1
*DOCL	Editing of text (in batch mode) with layout auxiliary functions via input in the screen mask in interactive mode	DÜ1
*DOCL NO[SCREEN]	Same function as the *DOCE command	DÜ1

4. "Symbolic parameters" can be used to input references to pages, chapters, lists and times. A summary of the parameters is given in the tables below.

Para-meter	Format	Length	Meaning (entry in page header or footer)
&S1	nnnn	1-4	Page counter, right-justified
&S2	nnnn	1-4	Page counter+1, right-justified
&S3	nnnnx	2-5	Page counter, right-just. with end identifier +/-
&P1	nnnn	1-4	Page counter, left-justified
&P2	nnnn	1-4	Page counter+1, left-justified
&DATUM	dd.mm.yy	8	Current date (European format)
&DATE	mm/dd/yy	8	Current date (US format)
&TIME	hh:mm:ss	8	Time of day

Variable	Meaning
&Tn	Count variable n=1..6, up to 3 digits, left-justified
&LNi	Value of individual level i of last title header, up to 3 digits, left-justified
&LCi	Full decimal classification of last title header of level i (i=0..9), up to 40 digits
&LTi	Text of last title header of level i (i=0...9)
&PA=i	Increment page counter
&RFx(i)	Reference variable (set to a specific value of a system variable with the ..SET command). x: is an alphanumeric identification (1 to 5 uppercase characters) i: gives the number of digits for the variable (up to 99, default value 3)

5. TOM-DOC can be used to produce simple graphics by entering the ..*COD and* ..*AS commands*. Graphics are printed out by entering the required character set, the parameter "IMAGE=POST" and the required paper format.

Character	Function
-	normal-font horizontal line
=	bold-type horizontal line
%	broken horizontal or vertical line
!	normal-font vertical line
I	bold-type vertical line
’	diagonal line slanted to right
‘	diagonal line slanted to left
+	connections in any direction
*	connections for centric lines
#	connections in the horizontal and vertical directions
(	rounded branch or normal branch upwards to left
)	rounded branch or normal branch upwards to right
?	rounded branch downwards
v	arrow point downwards
^	arrow point upwards
<	arrow point to left
>	arrow point to right

3 Example

3.1 The Task

In this example session we are going to familiarize ourselves more with the TOM-DOC commands by creating a document with several pages. The document is to contain a number of general comments on structured programming and is to be enhanced with tables and graphics. The pages are to be numbered in sequence and provided with header and footer. A table of contents is to be at the beginning of the document and a keyword index at the end. The text is to be enclosed in a frame created by the "|" character.

On the next page there is a diagram that illustrates the most important control commands for page layout.

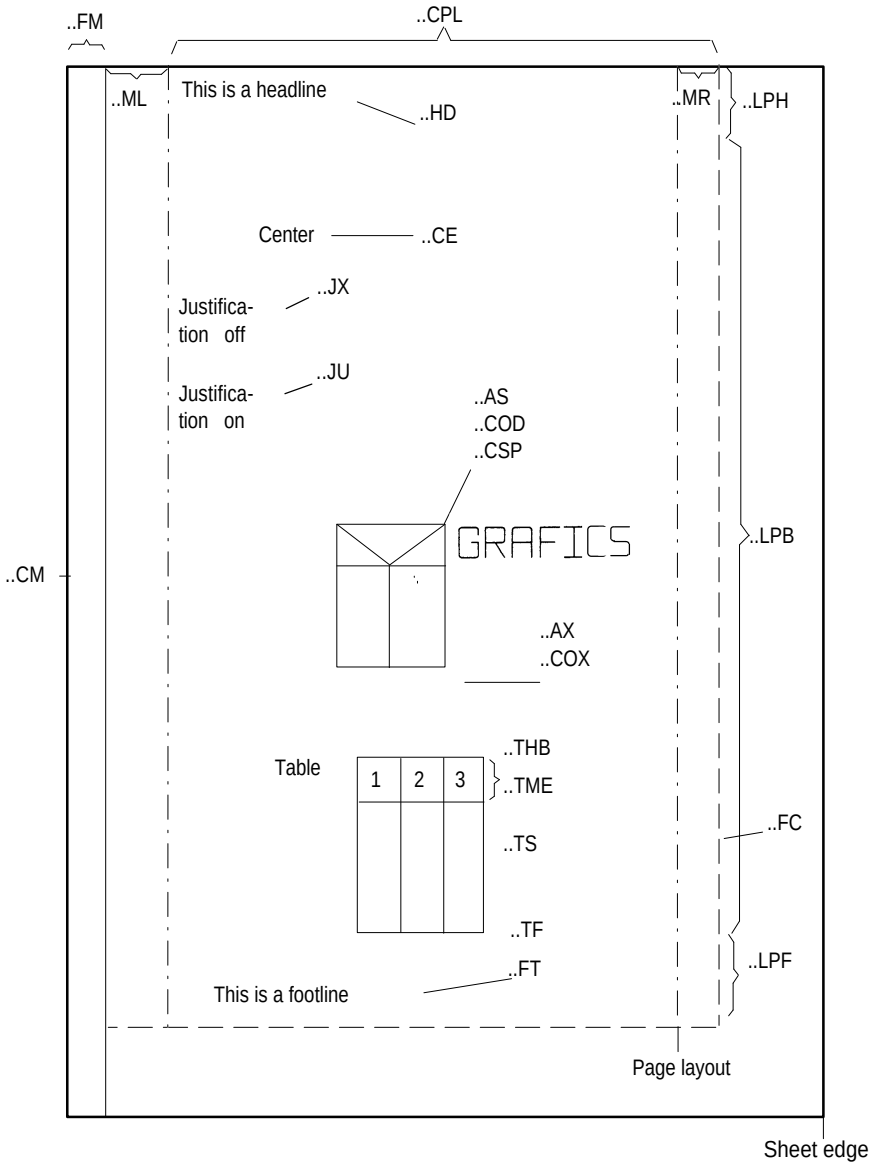


Figure 3-1: Page layout with TOM-DOC commands

3.2 Page Layouting

3.2.1 Page Dimensions

The layout of a page is determined by the elements used, such as the header, footer, tables, continuous text, margins, etc.

You set the size of these different elements by means of TOM-DOC commands, e.g. ..LPH for the number of lines in the header or ..CPL for the number of characters per line. Note that continuous text always begins with an indent of three blanks. You cannot change this.

If you wish, you can work with TOM-DOC's default values, but we will change these default values below in order to illustrate how the individual commands work.

3.2.2 Page Layout

We need the following commands to define the layout of the page:

Command	Default
..FM (File Margin)	10
..CPL (Characters Per Line)	74
..LPB (Lines Per Body)	44
..LPH (Lines Per Header)	8
..LPF (Lines Per Footer)	8
..ML (Margin Left)	1
..MR (Margin Right)	= CPL - 6

These commands alone suffice to give your text an attractive layout. Now write the following comment in the first line of your editor:

```
..* Page layout
```

This comment line serves to structure the individual sections within the definition part. It is best to get used to using such comments from the very start, because they will help you later when revising the file and they give a better overview when searching for errors.

Then we define the length of the page with the following commands:

```
..LPH=4
..LPB=55
..LPF=3
```

With these three commands we reserve 4 lines for the header of the page, 3 lines for the footer and 55 lines for the body of the text. This gives us a total of 62 lines, which corresponds approximately to the number of lines on an A4 page. These commands

also ensure that there is a page break after 62 lines.

To set the left and right text margins we continue by entering the following commands in our file:

```
..ML=5
..MR=58
..CPL=70
```

With these three commands we achieve the following:

1. With ..ML we set the left text margin at column 5.
2. With ..MR we set the right text margin at column 58. This means that we can fit 53 characters into one line. You may well now ask why we need the ..CPL command ?
3. With ..CPL we define the effective print width, the number of characters per line.

3.2.3 Frame

You will remember that we wanted to use the "|" character to place a frame around our text. This character is printed in column 0 (after the printer control character) on the left and in the column you specified with ..CPL (default: 74) on the right. To calculate the frame character on the right, you begin at the above-mentioned column 0 (so that it is always relative to the column on the left). Type the following command, which defines the frame character, in your list of commands:

```
..FC=3A (frame char.)
```

Now use your editor to write a text about two or three lines long. Start text editing with *DOCF. If you have made a mistake, you will have an error list in work area 2; the complete edited text is contained in work area 1. Here you can review the text and study the effect of your first commands. Note that the frame character (’3A’ corresponds to the character "|") appears on the screen as a non-printing character.

Now delete the text but leave the TOM-DOC commands in your file.

3.3 Defining Control Characters

Now that we have defined the page layout, we shall have to think about what other characters we need for editing the text. Here we are talking about how to underline, set words in bold type, mark words for the keyword index and, in our case, how to represent the special German characters such as Ä,ä,Ü,ü,ö,Ö and ß. It is useful sometimes to leave blanks "_" and also to prevent words from being divided at wrong points.

We shall tackle these problems in this part of our exercise. Now write in the next line:

```
..* Define control characters
```

We need the following commands to define the individual control characters:

```
..MB=del  (metablank)
..SF=del  (special flag)
..EMF=del (bold type)
..ULF=del (underlining)
..IF=del  (index flag)
..HY=del  (hyphenator)
```

The definition commands just assign specific functions to the characters you specify. We would like to draw your attention here to the following note which will help you to avoid errors:

Note

For the control characters only use characters which you know will definitely not be needed later on in the text and, as far as possible, no dummy graphic characters.

If you are thinking that one can never know exactly, then you are quite right. This is why we would like to draw your attention to another note:

Note

You can cancel the definitions for the control characters by simply repeating the definition command but without entering an operand. You cancel the control character for underlining, for example, by writing the following in your text:

..ULF

What do all the characters mean now? Let us look at them one by one.

3.3.1 The Metablank

With ..MB=del you define a "metablank" character. The metablank stands for a blank space which is to be retained. Generally TOM-DOC deletes all superfluous blanks or makes a line break at those points. If you have long numbers with intermediate spaces, however, a line break in the middle would not look good and could lead to confusion. In such cases write:

```
..MB=@
```

```
1@234@567@890
```

This ensures that the long number will always be undivided on one line.

The printout of the number will then look like this:

```
1 234 567 890
```

You can also use the metablank when using TOM-DOC to create forms. Your text input could then look like this:

```
Name:@@@@@@@@@@@@@@@@@@@@@@Forename:@@@@@@@@@@@@@
```

The printout will look like this:

```
Name:                          Forename:
```

You can usually decide at the beginning of a session whether you will be using the metablank frequently in a text. Then it is best to define it at the beginning of the TOM-DOC text, otherwise as and when needed.

3.3.2 Marking Umlauts

With the character which you have defined as the "special flag" you can generate umlauts and the "ß" character. Input of the special German characters in the CODE mode of the editor does not function with every keyboard. This is why these special characters often have to be specially marked with the ..SF command. It looks like this:

```
..SF=$
```

```
Ich wu$nsche, das$ immer Sommer wa$re.
```

This will then be printed as follows in the final document:

```
Ich wünsche, daß immer Sommer wäre.
```

3.3.3 Underlining and Bold Type

The next two control characters we are going to define are for underlining and emphasizing parts of the text in bold type. There are commands in TOM-DOC format for underlining and bold type. A marker (delimiter), however, has the advantage that it can be entered directly in the text.

You are already familiar with the TOM-DOC command for underlining:

```
..UL  (underline on)
..ULX (underline off)
```

There is a similar command for bold type:

```
..EM  (emphasize, or bold type, on)
..EMX (emphasize, or bold type, off)
```

It is recommended to use these specific commands especially if you have large sections of text to mark. If you just want to emphasize one word or sentence, then it is much easier to define a marker.

Enter the following assignments in your text file:

```
..EMF=[
..ULF='
```

Text sections enclosed in the characters "[" or "'" will now be emphasized/underlined in the edited text.

Notes:

- It's not possible to underline only parts of words.
- It you want to use bold type, you must also set the character spacing explicitly. You do this by means of the following command:

  ```
          { 10 }
  ..CSP=  { 12 }  (characters per inch)
          { 15 }
  ```

  With this command you define the character spacing per inch to be used for printing. On printing, the CHARS print parameter must coincide with this assignment.

We want to work with a large character set, so we write the following in our text file:

```
..CSP=10
```

Note

– If you want to underline a word or emphasize it with bold type and at the same time wish to avoid the word being divided by setting a defined hyphenator, then you must write the control character for underlining/emphasis first and then the hyphenator.

– If you do not yet know the character set you want to use when printing your document, simply type the command ..CSP (Character Spacing) in your document, without operands. When you print the document subsequently, you can set the required character set by means of the CHARS operand of the BS2000 /PRINT command.

3.3.4 Marking Keywords

We are now going to define a character which we are going to use to mark words for the keyword index. Here we use the command:

```
..IF=]
```

Words which are to be included in the keyword index must now be enclosed in "]" characters.

It depends on your text whether you can make good use of this character or not. Normally, entries in the keyword index are in the nominative form. This can however lead to problems. Consider the following sentence:

```
The girl’s dress is pretty.
```

If you mark "girl’s", then "girl’s" will be entered in the keyword index and not "girl" as desired. You have to mark "girl" in a different way. You do this with the command

```
..SI=girl
```

You can use the command

```
..SI=op
```

to mark any keyword in the right declension for inclusion in your keyword index.

3.3.5 Hyphenators

There is just one more important marker to be defined: the hyphenator. You define it with the following command:

```
..HY={
```

The hyphenator, in this case "{", fulfills a number of functions:

1. It marks a possible word division point. If a word cannot fit on one line, then it is divided at the point marked by the hyphenator. For example, the marked word "Datei{aufbereitung" appears as "Datei-aufbereitung" in the edited text if it has to be divided.
2. If you set the marker in front of a word (e.g. "{Dateiaufbereitung"), then the word will not be divided at all.
3. For words which are hyphenated anyway the marker can be used to mark the hyphen as a division point. In this case the marker has to be placed in front of the hyphen. The word will then be divided, if necessary, at that given point, but the hyphen will only appear once.

Now you know the most important markers. These characters can help a great deal when working with TOM-DOC but can also easily be a source of errors when editing texts.

Note

Since dummy graphic characters are very often used as markers, you must always remember to cancel the command just before any part of the text in which graphics are to appear. You do this by repeating the command without giving an operand.

The use of the BS2000 hyphenator is explained in section 2.7 on page 39.

3.4 Header and Footer

We shall now consider the layout of the page header and footer. Both header and footer are to be framed in bold type in contrast to the frame of the body of the text, which we have created with the "|" character. We need graphic conversion, therefore, for the header and footer. This we achieve with the TOM-DOC command

```
        ⎡ Y ⎤ (graphic mode   )
..CHF=  ⎨   ⎬ (for header and )
        ⎣ N ⎦ (footer         )
```

The ..CHF=Y command ensures that dummy graphic characters in the header and footer are converted. Now we have to define what is to be in the header and footer lines. This, too, is relatively simple. For the header write:

```
..HD=1 '+=================================================+'
..HD=2 'I                                                 I'
..HD=3 'I                                                 I'
..HD=4 '+=================================================+'
..HD=5 '!                                                 !'
```

The process is identical for the footer. Define the line and then write the text:

```
..FT=1 '!                                                 !'
..FT=2 '+=================================================+'
..FT=3 'I                                                 I'
..FT=4 '+=================================================+'
```

The "!" character is the dummy graphic character for a normal-type vertical line. We need it to make the link to the text frame character "|". The "+" character joins the vertical and horizontal lines, and the "I" and "=" characters are needed to convert the vertical and horizontal lines respectively into bold type.

It is not enough, of course, just to have frames for the header and footer. We want to put something in them, too. The header is to contain the abbreviated title of the text and the page number. The footer is to contain the current date and the number of the following page. We fill out the header and footer frames as follows:

```
..HD=1 '+=================================================+'
..HD=2 'I  Structured Programming                &S1      I'
..HD=3 'I  - an Essay -                                   I'
..HD=4 '+=================================================+'
..HD=5 '!                                                 !'
```

and:

```
..FT=1 '!                                                 !'
..FT=2 '+=================================================+'
..FT=3 'I   written on &DATUM                     &S2     I'
..FT=4 '+=================================================+'
```

Here you can see how "symbolic parameters" are used. With "&S1" the pages are numbered consecutively, "&S2" gives the number of the following page - if any - and

"&DATUM" shows the current date in the European format.

Once you have entered all this into your text file, you should enter a line of text and then initiate text editing with "*DOCF". Now change to work area 1 and examine the result. If the dimensions of your header and footer are not quite right, correct them in work area 0. When the dimensions are correct delete the text line. We only needed it to initiate text editing.

Note

It is certainly not everyone's cup of tea to write out the whole definition and frame layout parts over and over again, so you can store these standard text parts in a separate file and copy them as required at the beginning of your text with the command

..SW file (switch input file).

You can, of course, read such a definition file into your file directly via the editor with the 'READ' command. In this case you would then only have to enter the correct texts in the header and footer.

The following example demonstrates how to create headers and footers using variables.

```
..LPH=4
..LPF=3
..LPB=20
..HD=1  &LT6
..HD=2  &LT7
..HD=3  &LT8
..HD=4  &LT9
..FT=2  &LT5
..FT=3  &LT4
..SET &LT6='THIS IS THE FIRST HEADER LINE'
..SET &LT7='THIS IS THE SECOND HEADER LINE'
..SET &LT8='THIS IS THE THIRD HEADER LINE'
..SET &LT9='THIS IS THE FOURTH HEADER LINE'
..SET &LT5='THIS IS THE SECOND FOOTER LINE'
..SET &LT4='THIS IS THE THIRD FOOTER LINE'
text text text text text text text
```

You define the page layout and dimensions for the header and footer as normal. Then you do not enter the text in the header and footer but simply set system variables for the title texts (standard format "&Ti"). Select an "i" of a relatively large order, so that there is no overlapping with other cross-references in your text. In the example we start with one of the 4th order. You then assign a text to the system variables with the ..SET command, just as shown above in the example. These texts then appear in the final edited text instead of the commands. You can employ the same method to have the header and footer change with each chapter or to tailor the layout of a title page to your needs.

3.5 Indexes

Now that we have the text layout fully in our grasp we can concentrate on the actual text processing. TOM-DOC is a great help here, too. TOM-DOC handles such tedious tasks as creating a keyword index, generating the table of contents or numbering the chapters. All you have to do is write the appropriate commands.

The table of contents should be at the beginning of our text, so immediately behind the layout definition we write the command

```
..TOC=Outline
```

This gives us the table of contents on the first page entitled "Outline".

If your text has a lot of subdivisions, there will also be a lot of entries in the table of contents. Leading lines between the text and page numbers help the reader enormously to find the right page number quickly in the table of contents. We can generate such leading lines in our table of contents by writing the following command in the text file:

```
..LL=Y
```

The chapter and section titles in the table of contents will now be joined to their corresponding page numbers by a dotted line.

We want to have the keyword index at the end of the text. At the end of the text, therefore, we have to instruct TOM-DOC to start with the keyword index via the command

```
..HIX=Keyword Index
```

TOM-DOC will now generate a keyword index entitled "Keyword Index".

Note:

To obtain correct page numbering for the whole document, without any numbers being repeated, you should enter the ..HIX=Keyword Index command after the ..TOC=Outline command.

The reason for this is that TOM-DOC creates an overall document from the different subdocuments: text, table of contents and keyword index. During this process, TOM-DOC does not have total control of the page counter for the whole document.

However, if you use the ..TCL=i command, the page after a table of contents that is not at the beginning of the output file is numbered correctly without you having to set a page counter explicitly in a new section. If the ..TOC=Outline command

comes after the ..HIX=Keyword Index command, the index is always inserted at the position at which it occurs in the sequence of subdocuments (after a page break).

Please note, therefore, that the current page numbering of the subdocument called "Keyword Index" always applies. After ..HIX=Keyword Index is copied in, its page numbering ends with a certain page number (x). This page counter is not passed on from ..HIX=Keyword Index to ..TOC=Outline.

Thus, only when ..TOC=Outline (with the possible addition of the ..TCL=i command) comes before ..HIX=Keyword Index is the correct numbering of the whole document guaranteed, because only in this way can TOM-DOC determine the value of the page counter correctly for the whole document.

TOM-DOC is designed so that the ..TOC command sensibly comes before the ..HIX command. In this case, the page numbering of the keyword index begins with the current value of the page counter.

3.6 Internal Text Commands

By "internal text commands" we understand all those commands used to format and layout texts within the defined frame. Also included are all the commands required to create footnotes, tables and dummy graphics.

At the beginning of our text we want to have a title line centered in the middle of the page, followed by a short abstract which is to be slightly indented.

3.6.1 General Control Commands

We use the ..CE command to center the title line. We also want the title to be in bold type, so we mark it accordingly with the character for bold type. If you have kept to our proposed text, then you should have the following in your text file:

```
..CE [Structured Programming[
```

For the following abstract we will indent the left margin by 5 columns and center the title "ABSTRACT" over the paragraph of text. The command sequence looks like this:

```
..HM=5
..CE ABSTRACT
```

The ..HM=5 command indents the left margin by five columns.

We shall insert two blank lines before the text of the abstract with the command

```
..NL=2
```

Now you can write a short abstract of your essay as we have done in our example (see figure 3-4).

The line indent which we set with the ..HM=5 command is automatically reset by the next command. You can also reset the indent by using the ..HM command without an operand.

3.6.2 Title Headers

We now have come to our first title header. It is to be a title header of the zero order, i.e. only one place is shown in the decimal classification of title headers. The command for the title header is as follows:

```
..L0 Basic Principles
```

In this case "Basic Principles" is the text of the title header. Whenever you want to write a new title header, you do not have to worry about the chapter number. TOM-DOC counts on automatically. In our example we have two more title headers of the same order classified with the ..L0 command only:

```
..L0 Historical Background
..L0 Prospects
```

If you want to create subchapters, you just have to change the order of the decimal classification. For the title header of the subchapter "Discussion" it looked like this:

```
..L1 Discussion
```

This then gives you a title header for a subchapter in decimal classification in the edited text, in our case:

```
1.1 Discussion
```

A layout of a text is optically more attractive if the title headers stand out typographically from the rest of the text. We achieve this by entering yet another command in the text file:

```
..EMT (emphasize title headers)
```

After this command all the title headers will be printed in bold type. A similar command is used to underline all the title headers:

```
..ULT (underline title headers)
```

3.6.3 Enumeration

The next TOM-DOC commands which we need in our example text also help you with this task. We want to draw up a short enumeration of the features of structured programming. Instead of going through the text and enumerating the features explicitly, we use the following commands:

```
..IN  (index on)
..)   (increment index)
..IX  (index off)
```

The ..IN command instructs TOM-DOC to start indexing. TOM-DOC starts numbering at "1" and makes an automatic indent. The ..) command increments the index. Make sure that you separate the command from the subsequent text with a semicolon. In the edited text there will then be a "2", etc. You will find more details on this command in the command description. We would just like to point out here that you should never forget the ..*IX command* when you have finished with an enumeration. Otherwise the left margin will not be reset.

In our example text we use this indexing command sequence for graphics, too.

3.6.4 Footnotes

The commands for footnotes are often needed. This is not a problem with TOM-DOC. First write a count variable e.g. (&T1) at the point where you would like to make a footnote. Then write the following command in a new line:

```
..FN (footnote on)
```

This command informs TOM-DOC that a footnote is to follow. Write the count variable for the footnote again and then the actual text for the footnote. Terminate the footnote by writing the following command in the next line:

```
..FNX (footnote off)
```

In our example it will look like this:

```
As opposed to program flowcharts which support unstructured development
through the ]GOTO command] (&T1),
..FN
(&T1) Bo#se Zungen or wicked tongues say in English-speaking parts of
the world that GOTO is just another "four-letter word".
..FNX
structured programming is represented in special figures, {Nassi-Shneiderman
diagrams(&T1).
..FN
(&T1) see Nassi, J.; Shneiderman, B.: Flowchart Techniques for Structured
Programming. SIGPLAN Notices of the ACM 8 (1973), No. 8, pp. 12 -26.
..FNX
```

In the edited text the footnotes appear at the bottom of the page, correctly numbered (see below in section 3.7).

Even with just these few commands you can improve the appearance of your text. If you follow our suggestion and store the basic layout in a file (possibly with system variables), then TOM-DOC will help enormously in layouting your texts.
Now, taking our example, we shall see how to integrate dummy graphic representations and tables in your text.

3.6.5 Graphics

We will begin with the representation of dummy graphics. The following command is in our introductory example:

```
..RL=14  (reserve lines)
```

Admittedly the command was inserted later because after text editing had been initialized with "*DOCF" and the result displayed in work area 1, we discovered that the page break was right in the middle of the graphics. With the ..RL=14 command we instructed TOM-DOC to start a new page if there were not 14 lines - the length of the first graphic representation - left on the page.

Another important command to give before the actual dummy graphic representation is:

```
..ULF
```

You will remember: with the ..ULF=’ command we defined the """ character as a delimiter for underlining. The """ is also a dummy graphic character which we need to represent slanting lines. We have to cancel the original assignment, therefore, by entering the ..ULF command without an operand.

Dummy graphic conversion is then started with the command

```
..COD
```

We also want TOM-DOC to execute in AS mode for the time being in order to avoid page breaks at inconvenient points. We do this by entering the command

```
..AS
```

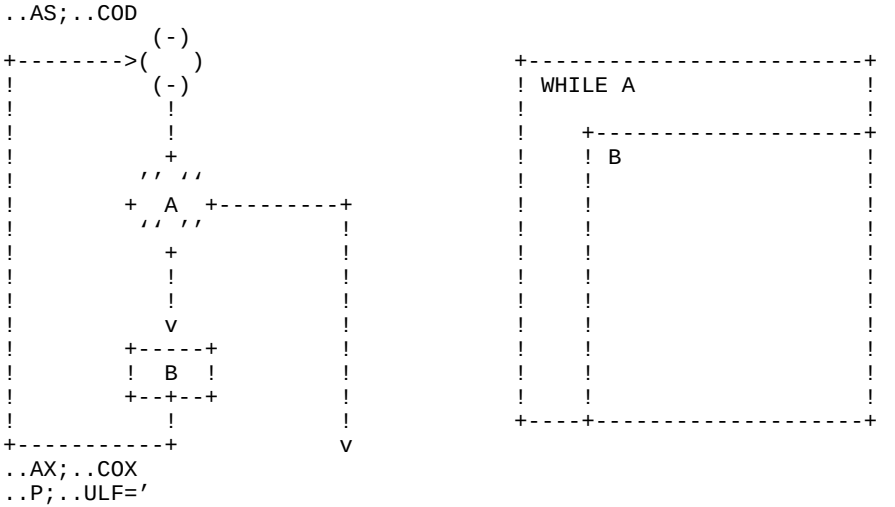
Now we can be sure that the graphics will be printed out as intended. As soon as you have entered these two commands you can draw to your heart’s content with the dummy graphic characters. You terminate dummy graphic conversion with the commands

```
..COX
```

and

```
..AX
```

In our example the circle, lozenge, rectangle and arrow are represented as follows:



The ..ULF=’ command ensures that our delimiter for underlining is enabled again. The ..*P command* instructs TOM-DOC to start a new paragraph.

3.6.6 Tables

We also need dummy graphic characters in our example to represent a table, because we want a bold-type frame and the columns in the table divided by a broken line. This we do by using the dummy graphic characters "%", "=" and "I".

It often happens that a table does not fit entirely on one page. With TOM-DOC you first create a table header which is then automatically reproduced on the following page when there is a page break. You can also define a line to be printed at the end of the page if there is not enough space for the entire table.

We need the following commands for the table header:

```
..THB (table header begin)
..THE (table header end)
```

You write the table header in between these two commands. You then have to instruct TOM-DOC that a table section has started with the command:

```
..TS (table)
```

If this ..TS command is followed by text or another character string, then whatever you write will appear in the bottom line of the table if it has to be divided by a page break.

Mark the end of the table with the command:

```
..TE (table end)
```

In our example we have included a small table of binary and decimal figures:

```
..THB
             +=============+================+
             I Binary sys. %  Decimal sys.  I
             +=============+================+
..THE
..TS=                 ------ to be continued ------
             I           0  %       0        I
             I           1  %       1        I
             I          10  %       2        I
             I          11  %       3        I
             I         100  %       4        I
             I         101  %       5        I
             I         110  %       6        I
             I         111  %       7        I
             I        1000  %       8        I
             I        1001  %       9        I
             I        1010  %      10        I
             +=============+================+
..TE
```

If the entire table does not fit on the page, we get the following result:

```
┌──────────┬──────────────┐
│ Binary sys.:  Decimal sys.
├──────────┼──────────────┘
          0 :       0
          1 :       1
         10 :       2
         11 :       3
        100 :       4
  ------ to be continued ------
```

```
┌──────────┬──────────────┐
│ Binary sys.:  Decimal sys.
├──────────┼──────────────┘
        101 :       5
        110 :       6
        111 :       7
       1000 :       8
       1001 :       9
       1010 :      10
└──────────┘
```

Figure 3-2: Table break

3.7 Example Source Code

The following pages show the final edited text of our example and the original text entered with TOM-DOC control characters. Unfortunately it is not always possible to print the pages in parallel, but we hope that the comments will give an adequate explanation of the control characters and their effect.

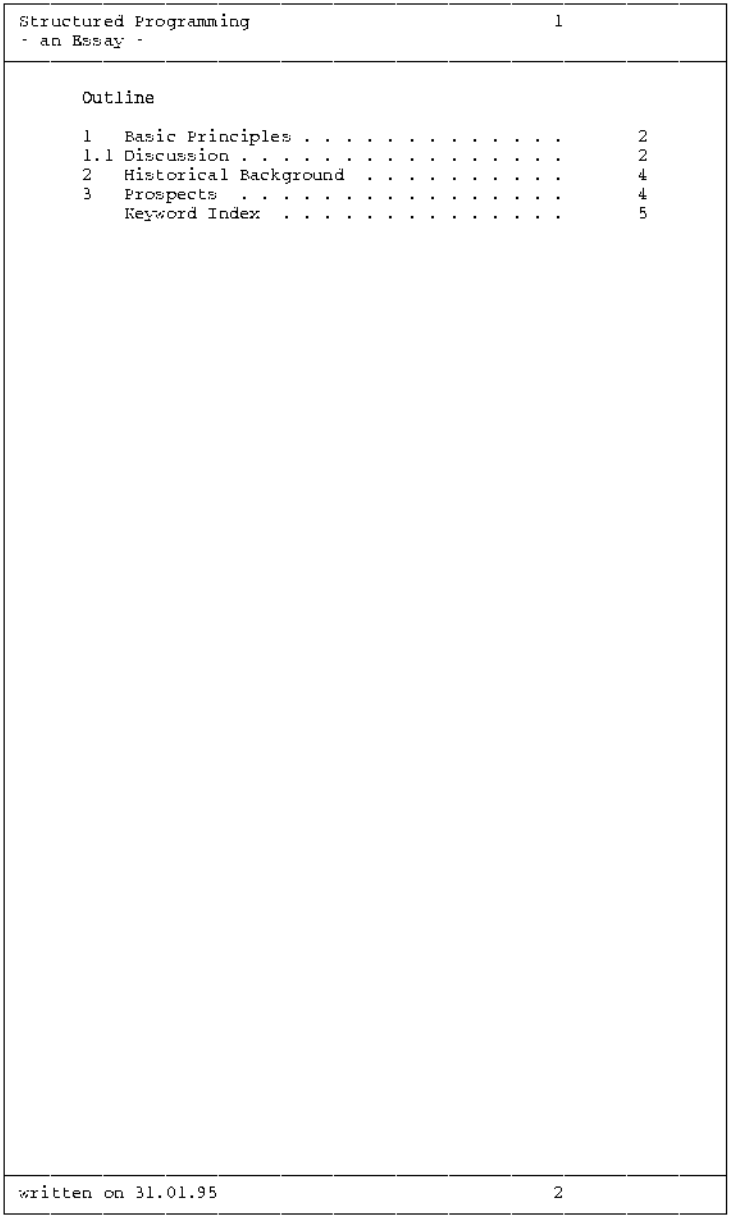


Figure 3-3: Table of contents created by TOM-DOC

```
..* GENERAL DEFINITIONS
..*
..OPF=Tomex.print;..* output file defined
..*
..* Page layout
..CPL=70;..* characters per line
..LPB=55;..* lines per body
..LPF=4;..* lines per footer
..LPH=5;..* lines per header
..SYL=y;..* syllabication (automatic word division)
..ML=5;..* margin, left
..MR=58;..* margin, right
..FC=3A;..* frame character defined
..*
..* Definitions
..*
..CSP=10;..* character spacing (for bold type and graphics)
..MB=@;..* metablank defined
..SF=#;..* umlaut marked
..EMF=[;..* emphasize flag (bold type on/off)
..ULF=';..* underline flag (underlining on/off)
..IF=];..* index flag
..HY={;..* hyphenator
..*
..* Header and footer inputs
..*
..HD=1 '+=========================================================================+'
..HD=2 'I Structured Programming                              &S1                I'
..HD=3 'I - an Essay -                                                           I'
..HD=4 '+=========================================================================+'
..HD=5 '!                                                                        !'
..FT=1 '!                                                                        !'
..FT=2 '+=========================================================================+'
..FT=3 'I written on &DATUM                                   &S2                I'
..FT=4 '+=========================================================================+'
..CHF=Y;..* code header and footer (dummy graphics)
..*
..* Table of contents
..*
..TCL
..TOC=Outline
..* title header for table of contents
..LL=Y;..* line leading (leading lines in table of contents)
..EMT;..* emphasize title headers (all titles in bold type)
```

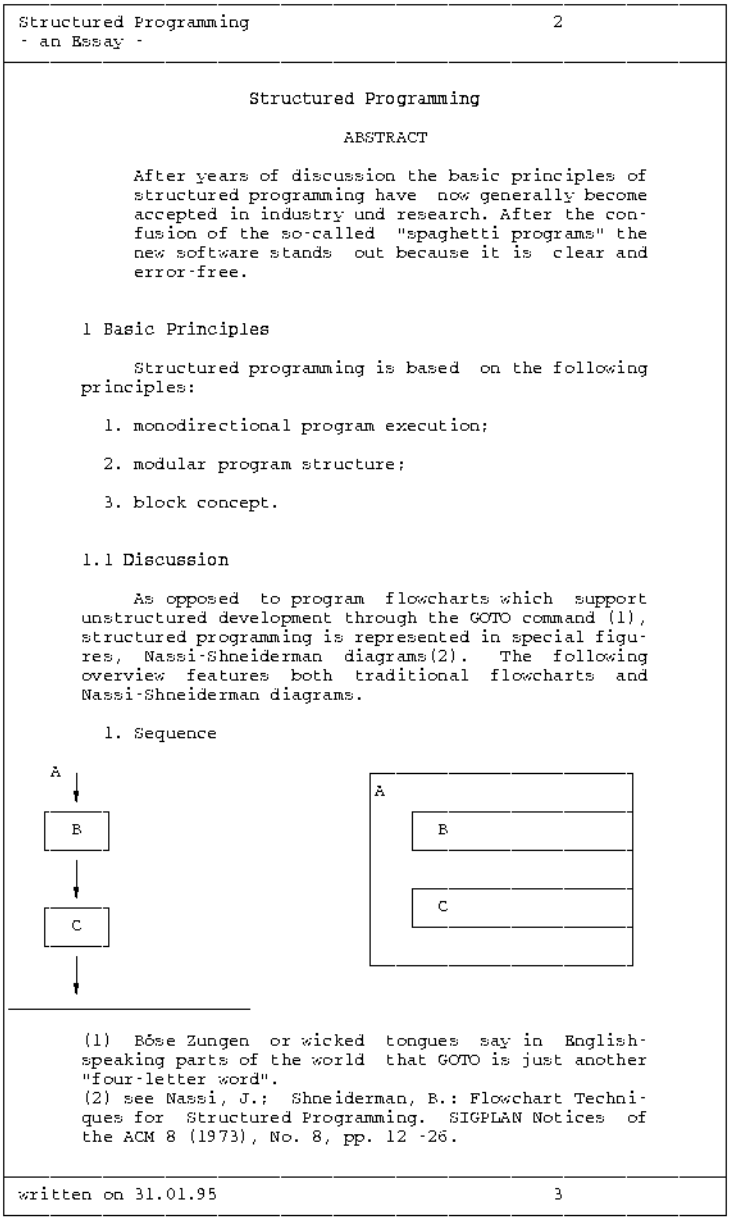


Figure 3-4: The first page of the edited text with dummy graphics and footnotes

```
..*
..* start of text
..*
..CE [Structured Programming[
..* title printed in bold type
..HM=5;..* left margin indented by 5 columns
..CE ABSTRACT
..* title header "Abstract" centered
..NL;..* blank line inserted
After years of discussion the basic principles of ]structured programming]
have now generally become accepted in ]industry] und research.
After the confusion of the so-called ]"spaghetti programs"] the ]new software]
stands out because it is clear and error-free.
..L0 Basic Principles
..* title header of the zero order
Structured programming is based on the following ]principles]:
..IN; monodirectional program execution;
..); modular program structure;
..); block concept.
..IX;..* an index enumeration is started with ..IN and terminated with ..IX
..L1 Discussion
..* title header of first order
As opposed to program flowcharts which support unstructured development
..SI=program flowchart
through the ]GOTO command] (&T1),
..FN
(&T1) Bo#se Zungen or wicked tongues say in English-speaking parts of
the world that GOTO is just another "four-letter word".
..* a footnote is inserted here
..FNX
structured programming is represented in special figures, {Nassi-Shneiderman
diagrams(&T1).
..* Nassi-Shneiderman is not to be divided
..FN
(&T1) see Nassi, J.; Shneiderman, B.: Flowchart Techniques for Structured
Programming. SIGPLAN Notices of the ACM 8 (1973), No. 8, pp. 12 -26.
..* another footnote was inserted here; note that the count variable remains
..* the same
..FNX
The following overview features both traditional ]flowcharts]
and ]{Nassi-Shneiderman diagrams].
..NL
..RL=14;..* page break if graphics do not fit on the page
..IN; ]Sequence]
..* index enumeration initialized
..NL=1;..ULF
..AS;..COD;..* AS mode and graphic conversion on
        A I                          +------------------------+
          v                          !A                       !
       +-----+                       !   +--------------------+
       !  B  !                       !   !  B                 !
       +-----+                       !   +--------------------+
          I                          !                        !
          v                          !   +--------------------+
       +-----+                       !   !  C                 !
       !  C  !                       !   +--------------------+
       +-----+                       !                        !
          I                          +------------------------+
          v
```

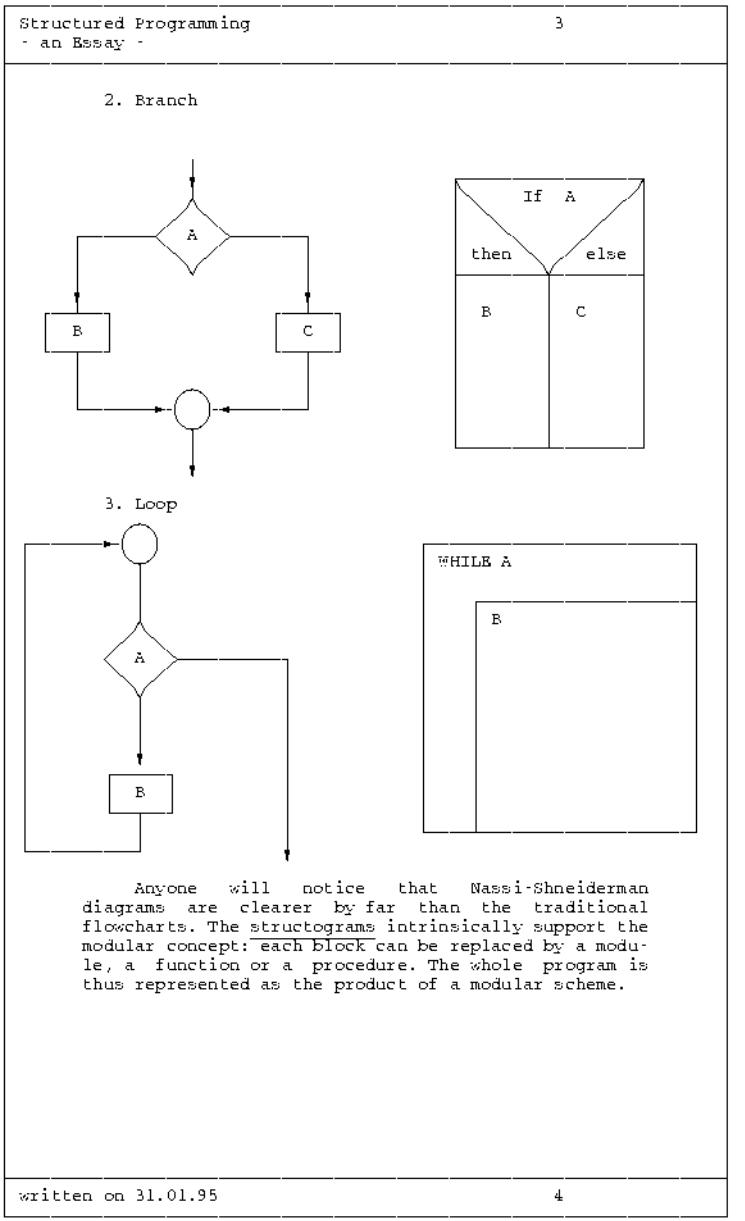
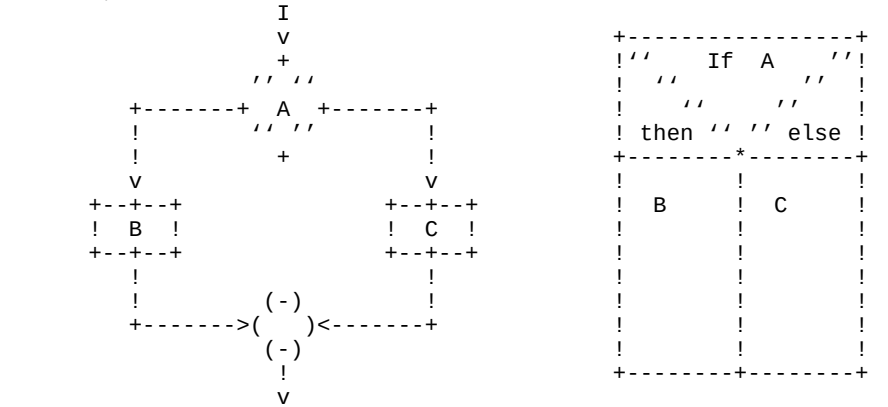
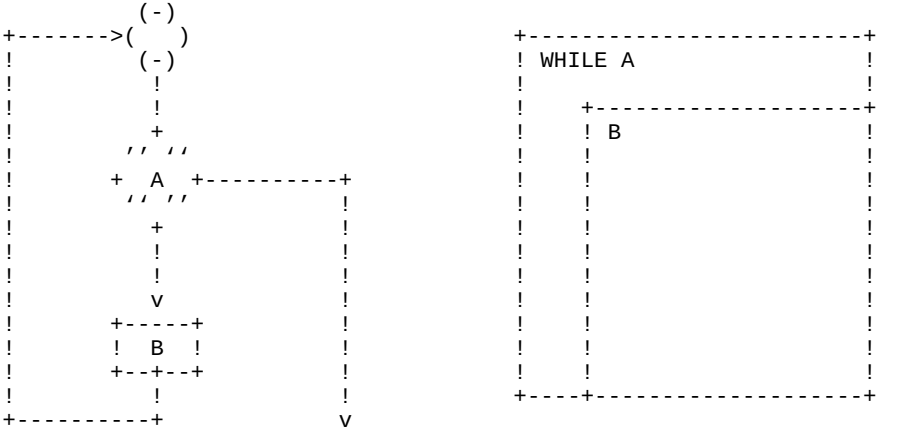


Figure 3-5: The second page of the edited text with dummy graphics

```
..AX;..COX;..** AS mode and graphic conversion off
..); ]Branch]
..NL=2
..AS;..COD
```



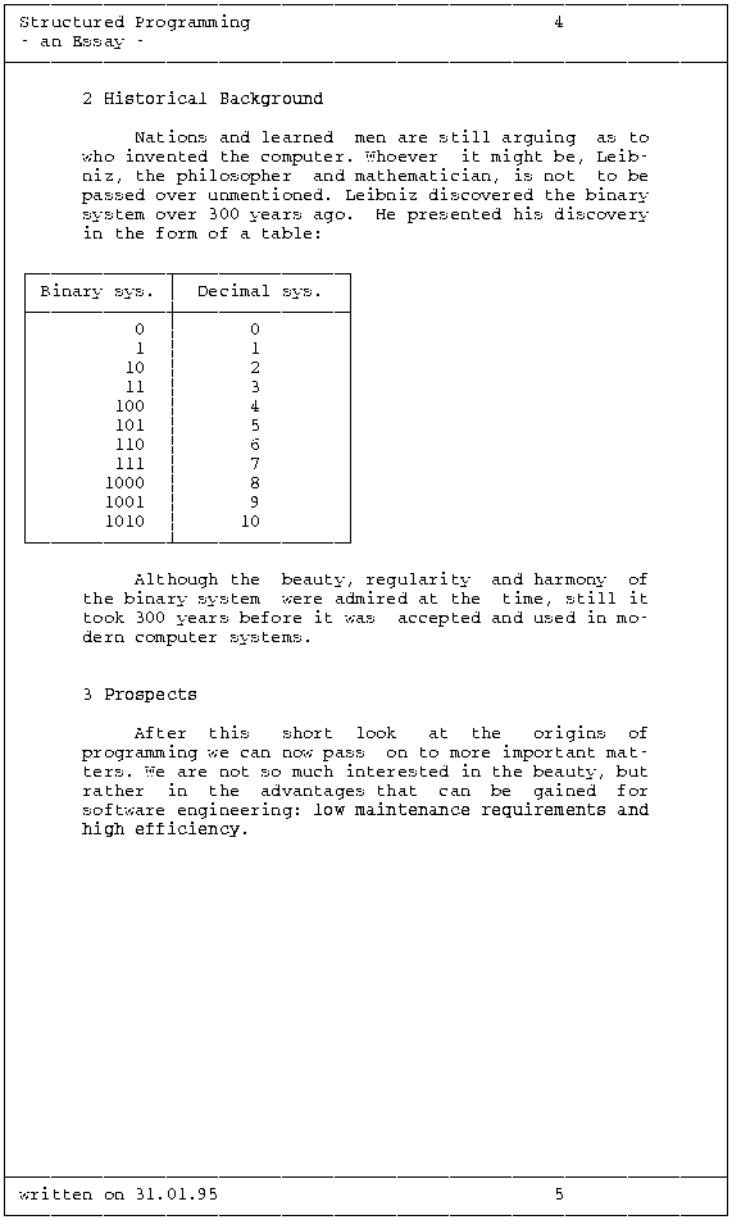

```
..AX;..COX
..); ]Loop]
..AS;..COD
```




```
..AX;..COX
..IX
..P;..ULF='
```

Anyone will notice that]Nassi-Shneiderman diagrams] are clearer by far than the traditional flowcharts. The ']structograms]' intrinsically support the modular concept: each block can be replaced by a module, a]function] or a]procedure]. The whole program is thus represented as the product of a modular scheme.

```
..NL=1
```



Structured Programming 4
- an Essay -

2 Historical Background

Nations and learned men are still arguing as to who invented the computer. Whoever it might be, Leibniz, the philosopher and mathematician, is not to be passed over unmentioned. Leibniz discovered the binary system over 300 years ago. He presented his discovery in the form of a table:

Binary sys.	Decimal sys.
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	9
1010	10

Although the beauty, regularity and harmony of the binary system were admired at the time, still it took 300 years before it was accepted and used in modern computer systems.

3 Prospects

After this short look at the origins of programming we can now pass on to more important matters. We are not so much interested in the beauty, but rather in the advantages that can be gained for software engineering: low maintenance requirements and high efficiency.

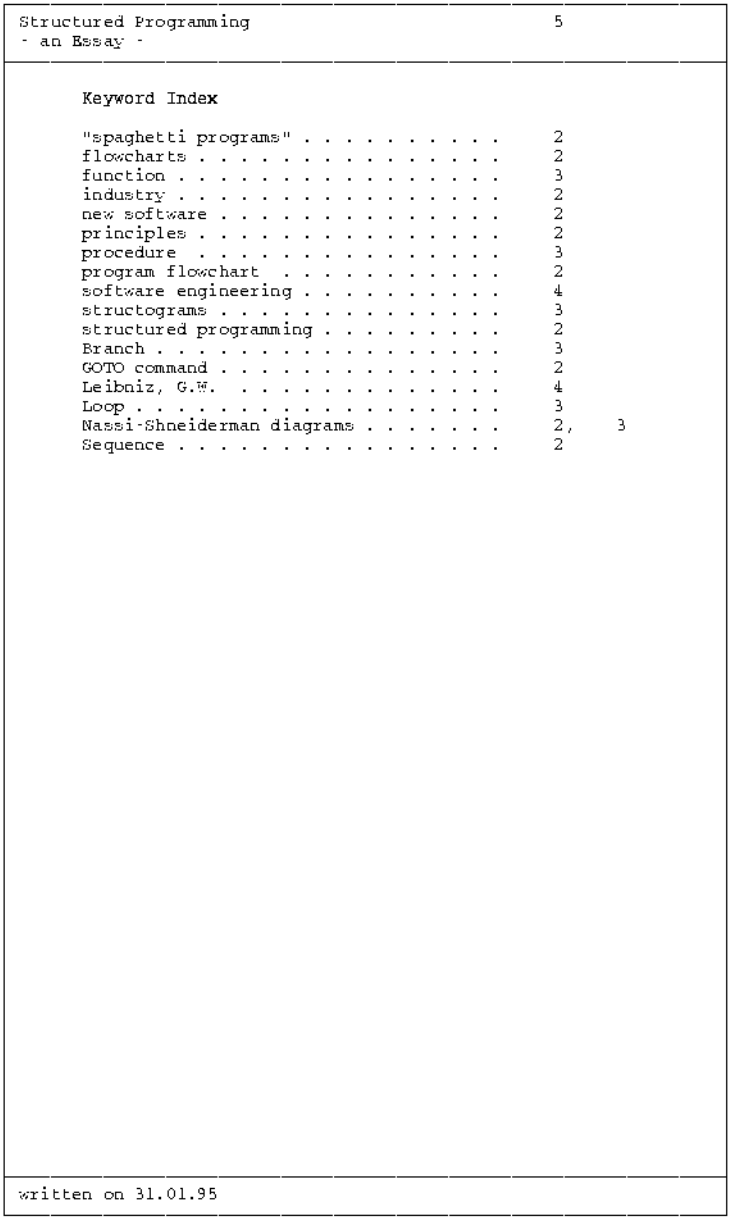
written on 31.01.95 5

Figure 3-6: The third page of the edited text with table and the "switch" file which was copied into it

```
..L0 Historical Background
Nations and learned men are still arguing as to who invented the computer.
Whoever it might be, Leibniz, the philosopher and mathematician, is not to
be passed over unmentioned.
..SI=Leibniz, G.W.
Leibniz discovered the binary system over 300 years ago.
He presented his discovery in the form of a table:
..NL;..COD;..AS
..THB
               +=============+================+
               I Binary sys. %  Decimal sys.  I
               +=============+================+
..THE
..TS=                  ------ to be continued ------
               I          0  %       0        I
               I          1  %       1        I
               I         10  %       2        I
               I         11  %       3        I
               I        100  %       4        I
               I        101  %       5        I
               I        110  %       6        I
               I        111  %       7        I
               I       1000  %       8        I
               I       1001  %       9        I
               I       1010  %      10        I
               +=============+================+
..TE
..AX
..P;..COX
Although the beauty, regularity and harmony of the binary system were admired
at the time, still it took 300 years before it was accepted and used in modern
computer systems.
..L0 Prospects
After this short look at the origins of programming we can now pass on to
more important matters. We are not so much interested in the beauty,
but rather in the advantages that can be gained for ]software engineering]:
[low maintenance requirements and high efficiency.[
..P
..SW=appendix
..HIX=Keyword Index
```

The ..SW File

```
..TH Appendix
After completion of the report we received this letter:
..NL
Dear Sirs,
..NL
It was with great interest that we learnt about your project.
We offer you our unconditional support in your venture.
If you should ever find yourselves in financial difficulties,
just let us know.
..NL
Yours sincerely,
```



```
Structured Programming                          5
- an Essay -
```

Keyword Index


```
written on 31.01.95
```

Figure 3-7: Keyword index created by TOM-DOC

3.8 Support of HIPO

3.8.1 What is HIPO?

HIPO is a procedure for designing software products. HIPO was first implemented at the beginning of the seventies to design a system which was to automate access to the New York Times archives. This was the first time that structured programming, up till then the subject of purely academic discussion, was put into practice. The success was indisputable: it took 13 months after release of the software before even one error could be found. After this initial success the HIPO process became much more widely known.

HIPO stands for "Hierarchy plus Input Process Output". It is a procedure for the hierarchical structuring of informational tasks. HIPO employs the top-down design methods. Basically there are two phases of design:

1. **Hierarchy**

   The first phase is to draft a rough design including all the tasks and sub-tasks of the system. The top-down method of design is to be strictly adhered to in the process. The top level contains relatively general task descriptions. The next level gives more detailed descriptions of these tasks and so on down until a structure is obtained in the form of a hierarchical tree diagram. This then represents the hierarchical interrelationships in the system.

2. Input Process Output

The second phase is to define the sequence of the steps required to execute a sub-task. Here the main point is not how the system is to be executed later, but what the interrelationships are between input data, processing, and output data. These interrelationships are represented in table form.

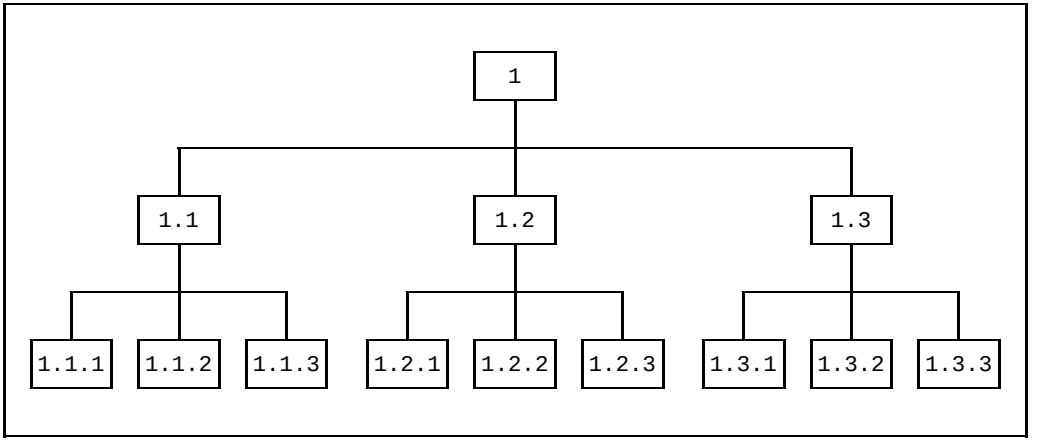


Figure 3-8: Tree structure

3.8.2 TOM-DOC Supports HIPO

HIPO is thus above all text-oriented. Coding comes at a much later stage with this process, so that the program designer can concentrate on the interrelationships between data and their consequences. In this respect TOM-DOC supports HIPO in important functions. TOM-DOC defines the hierarchical levels of the processes with ..Li commands. The tree structure of these processes is thus rendered in the form of the decimal classification of the table of contents.

Processes which are used more than once must appear at the appropriate positions in the tree diagram each time. In TOM-DOC this is done with the ..SW copy function.

The individual processes, including their inputs and outputs, are represented in HIPO by means of so-called IPO diagrams.

Input	Process	Output
Files Records Masks Fields	1. Function a 2. Function b 3. Function c ... References to other IPO diagrams References to extended descriptions References to other documents	Files Records Masks Fields

The user is free to determine both the contents and the format (A4 portrait/landscape ...) of the IPO diagrams over and above the basic schema.

TOM-DOC creates IPO diagrams as tables. In the case of large-scale IPO diagrams, it provides for automatic page breaks and copying of the IPO header. The generation of IPO diagrams can be accelerated and standardized by using the TOM-GEN generation functions.

Detailed descriptions are not included in the IPO diagrams. Instead they are stored externally as so-called "special descriptions" which are then referenced. If symbolic parameters are used for this, then not only the names of the documents but also their current classification and page number can be indicated.

The data flow in IPO diagrams is recorded by TOM-DOC in the form of a keyword index. To do this the input/output data of the IPO diagrams just have to be marked as keywords. This special keyword index can then be sorted as required using the ..ISF sort command.

3.8.3 Example

This example shows how to create a basic HIPO diagram using TOM-DOC. The task is to design a program for financial accounting. The draft hierarchical design is stored in a central control file in decimal classification. The ..*SW command* is used to call up the individual IPO diagrams from this central control file.

The input for the central control file is as follows:

```
..LPH=05
..LPF=02
..HD=1 ’               Specifications’
..HD=2 ’ &LN0   &LT0’
..HD=3 ’ &LC1   &LT1’
..HD=4 ’ &LC2   &LT2’
..FT=1 ’ Consulta  BW1, Mayer  FB.PFL  &Datum    Page:  &S3’
..LPB=038
..CPL=110
..SRC=N
..FC=40
..TOC=         Table of contents
..HIX=Documentation of use of data in IPO diagrams
..RIX=C
..IF=#
..ISF=(3,8),(1,1)
..SF=$
..PIN=0
..TL=2
..L0 FB.IPO.......Financial Accounting
..SW=FB.IPO
..L1 FBDB.IPO.....Receivables Accounting
..SW=FBDB.IPO
..L2 FBDBUPD.IPO..Update Receivables Master Data
..SW=FBDBUPD.IPO
..L2 FBDBACC.IPO..Enter Accounts Receivable
..SW=FBDBACC.IPO
..L2 FBDBINQ.IPO..Evaluate Accounts Receivable
..SW=FBDBINQ.IPO
..L1 FBSK.IPO.....General Ledger Accounting
..SW=FBSK.IPO
..L2 FBSKUPD.IPO..Update General Ledger Master Data
..SW=FBSKUPD.IPO
..L2 FBSKACC.IPO..Enter General Ledger Accounts
..SW=FBSKACC.IPO
..L2 FBSKINQ.IPO..Evaluate General Ledger Accounts
..SW=FBSKINQ.IPO
..L1 FBKB.IPO.....Payables Accounting
..SW=FBKB.IPO
..L2 FBKBUPD.IPO..Update Payables Master Data
..SW=FBKBUPD.IPO
..L2 FBKBACC.IPO..Enter Accounts Payable
..SW=FBKBACC.IPO
..L2 FBKBINQ.IPO..Evaluate Accounts Payable
..SW=FBKBINQ.IPO
..LPB=054
..CPL=074
..L1 FB.DAT.......Common Tables and Master Data
..SW=FB.DAT
..L1 FB.REQ.......Specialist/Technical Requirements
..SW=FB.REQ
..L0 FB.ANH.......Appendix
..SW=FB.ANH
```

First the header and footer lines are defined. The header is 5 lines long. The title is entered in the first line and then the following in header lines 2 to 4:

- the value (&LN0) and the title (&LT0) of the zero order (in this case the figure 1 and "Financial Accounting");
- the full decimal classification of the last title of the first order (&LC1) together with the text (&LT1);
- finally, the full decimal classification of the last title of the second order (&LC2) together with the text (&LT2).

The footer contains the company name, date and current page number. This is followed by the definition of the number of lines for the text body (..LPB) and the number of characters per line (..CPL). There is to be no line numbering (..SRC=N). The blank "_" (hexadecimal 40) is selected as frame character (..FC).

The ..TOC command sets the title of the table of contents and the ..HIX command does the same for the keyword index (in this case the documentation of use of data). The ..RIX command determines that the references in the keyword index refer to individual IPO diagrams which stand as sub-chapters in the table of contents.

The ..IF=# command selects the character for marking the words to be entered in the keyword index. The keyword index is to be sorted as of the third position to the length of eight characters, but also by the first position. All this is done with the ..ISF=(3,8),(1,1) command.

The ..SF=$ command defines the character used to mark umlauts. The ..PIN=0 command ensures that there is no indenting of new paragraphs and the ..TL=2 command implements a page break after each title header of the second order. This means that a new page is taken for each IPO diagram.

The other lines in the table of contents are made up in the same manner. First the level of decimal classification of the title is defined (with ..L0 to ..L2). After each title the ..SW command causes branching to another file in which the IPO diagram required has been stored. Here, the text of the titles includes the filename and a designation which is not only readable but also understandable.

The output of the table of contents then looks like this:

```
                S p e c i f i c a t i o n s

T a b l e  o f  c o n t e n t s

1      FB.IPO.......Financial Accounting                          1
1.1    FBDB.IPO.....Receivables Accounting                        2
1.1.1  FBDBUPD.IPO..Update Receivables Master Data                3
1.1.2  FBDBACC.IPO..Enter Accounts Receivable                    5
1.1.3  FBDBINQ.IPO..Evaluate Accounts Receivable                 6
1.2    FBSK.IPO.....General Ledger Accounting                     7
1.2.1  FBSKUPD.IPO..Update General Ledger Master Data            8
1.2.2  FBSKACC.IPO..Enter General Ledger Accounts                9
1.2.3  FBSKINQ.IPO..Evaluate General Ledger Accounts           10
1.3    FBKB.IPO.....Payables Accounting                          11
1.3.1  FBKBUPD.IPO..Update Payables Master Data                 12
1.3.2  FBKBACC.IPO..Enter Accounts Payable                     13
1.3.3  FBKBINQ.IPO..Evaluate Accounts Payable                  14
1.4    FB.DAT.......Common Tables and Master Data              15
1.5    FB.REQ.......Specialist/Technical Requirements          16
1.5.1  Quantitative Structure                                  17
1.5.2  Periodicity per Department Structure                    18
1.5.3  Operating Mode per Department Function                  19
1.5.4  Quality Requirements                                    20
1.5.5  Organization                                            21
1.5.6  Data Transfer                                           22
2      FB.ANH.......Appendix                                   23

Consulta  BW1, Mayer   FB.PFL   13.07.84        Page:   1-
```

The control commands and table frame were created by means of a generation function (see section 3.8.4). The individual inputs in the table were done with the editor. Please note that we have retained a number of German abbreviations in the following example.

Inputs and outputs are marked with "#".

```
..NL=2
..COD
..AS
..THB
IPO: FBDBUPD

+-------------------------------+-----------------------------------------------+-------------------------------+
!            Input              !                  Process                      !           Output              !
+-------------------------------+-----------------------------------------------+-------------------------------+
..THE
..TS=+-------------------------------+-----------------------------------------------+-------------------------------+
! #I:MANDANT.FIL#                !1. Set up FB customer master record          ! #O:FBMANDT.REC#                !
!    (customer data file)        !   (FB customer record)                      !                               !
!                                !                                             !  FB customer record           !
!  Access to the customer data   !   Set up because of a customer bill,        !                               !
!  entered and updated in the    !   payment, credit note                      !  - Name code                  !
!  BB for each NL and managed    !                                             !  - Match code                 !
!  in the ZA                     !   - Selection of FB-specific data           !  - Account number             !
!  (file and/or listing)         !     Data fields (name code, match code,     !  - Accounting area            !
!                                !     accounting area)                        !  - Reconciliation account     !
!                                !                                             !                               !
!                                !   - Definition of acc. no., reconc. acc.    !                               !
!                                !   - Predefined screen masks with check      !                               !
!                                !     routines and error messages             !                               !
!                                !                                             !                               !
! #I:FBMANDT.REC#                !2. Update FB customer record                 ! #O:FBMDTP01.FIL#               !
!                                !   FB-specific data only                     !                               !
!  Call FB customer record       !                                             !  Entry/error log              !
!                                !   - add; change; delete if no turnover      !                               !
!                                !     available                               !                               !
!                                !   - BB can issue deletion note              !                               !
!                                !   - Predefined screen masks with check      !                               !
!                                !     routines and error messages             !                               !
!                                !   - Logging of updates via automatic system ! #O:FBMDTP02.FIL#               !
!                                !     documentation                           !                               !
!                                !                                             !  Update log                   !
!                                !                                             !                               !
!                                !                                             !  - date; user;                !
!                                !                                             !    old/new contents           !
!                                !                                             !                               !
!                                !                                             !  - printout in ZA, monthly    !
!                                !                                             !                               !
!                                !3. Display FB customer record                ! #O:FBMDTM01.MSK#               !
!                                !                                             !                               !
!                                !   via match code or account number,         !  Screen display and/or        !
!                                !   accounting area                           !  printout (hard copy)         !
!                                !                                             !                               !
!                                !                                             !  - FB customer record         !
!                                !                                             !  - traffic volume             !
!                                !                                             !  - entry errors               !
!                                !                                             !                               !
!                                !4. Data transfer                             !                               !
!                                !                                             !                               !
!                                !   cf.  &RFFBRQ0(1).&RFFBRQ1(1).&RFFBRQ2(1)  !                               !
!                                !   Page &RFFBRQS                             !                               !
+-------------------------------+-----------------------------------------------+-------------------------------+
..TE
..AX
..COX
```

The ..NL=2 command inserts two blank lines before the table. The table itself is edited in AS mode, i.e. there is no automatic line break. In addition, the ..COD command was used to start dummy graphic conversion. The table header is enclosed between the ..THB and ..THE commands. The table section is marked with ..TS=+ and ..TE marks the end of the table. Both the graphic and the AS modes are switched off again afterwards.

The output of the table then looks like this:

```
           S p e c i f i c a t i o n s
1       FB.IPO.......Financial Accounting
1.1     FBDB.IPO.....Receivables Accounting
1.1.1   FBDBUPD.IPO..Update Receivables Master Data                Sheet: 1


1.1.1 FBDBUPD.IPO..Update Receivables Master Data

IPO: FBDBUPD
```

Input	Process	Output
I:MANDANT.FIL (customer date file) Access to the customer data entered and updated in the BB for each NL and managed in the ZA (file and/or listing)	1. Set up FB customer master record (FB customer record) Set up because of a customer bill, payment, credit note - Selection of FB-specific data Data fields (name code, match code, accounting area) - Definition of acc. no., reconc. acc. - Predefined screen masks with check routines and error messages	O:FBMANDT.REC FB customer record - Name code - Match code - Account number - Accounting area - Reconciliation account
I:FBMANDT.REC Call FB customer record	2. Update FB customer record FB-specific data only - add; change; delete if no turnover available - BB can issue deletion note - Predefined screen masks with check routines and error messages - Logging of updates via automatic system documentation	O:FBMDTP01.FIL Entry/error log O:FBMDTP02.FIL Update log - date; user; old/new contents - printout in ZA, monthly

```
Consulta  BW1, Mayer   FB.PFL  05.11.84       Page:   3+
```

```
            S p e c i f i c a t i o n s
1        FB.IPO.......Financial Accounting
1.1      FBDB.IPO.....Receivables Accounting
1.1.1    FBDBUPD.IPO..Update Receivables Master Data                Sheet: 2

IPO: FBDBUPD
```

Input	Process	Output
	3. Display FB customer record via match code or account number, accounting area 4. Data transfer see 1.5.6 Page 22	O:FBMDTM01.MSK Screen display and/or printout (hard copy) - FB customer record - traffic volume - entry errors

```
Consulta  BW1, Mayer   FB.PFL   05.11.84          Page:   4+
```

Finally we can see in the keyword index how the data in the IPO diagrams is used. The data index is sorted in ascending order of inputs and outputs. The IPO diagrams are indicated with their decimal classification.

Documentation of use of data in IPO diagrams

```
0:FBCHAP01.FIL  1.2.1
0:FBERRP01.FIL  1.2.1
I:FBKOPLA.FIL   1.2.1
I:FBMANDT.REC   1.1.1
0:FBMANDT.REC   1.1.1
0:FBMDTM01.MSK  1.1.1
0:FBMDTP01.FIL  1.1.1
0:FBMDTP02.FIL  1.1.1
I:FBSACHKO.FIL  1.2.1
0:FBSACHKO.FIL  1.2.1
I:MANDANT.FIL   1.1.1
```

3.8.4 Generation Task for IPO Diagrams

The generation function illustrated below supports the creation of IPO diagrams in A4 format (landscape). It can be modified with the aid of TOM-GEN to meet individual requirements.

```
        MACRO
        IPO
.*      Generation of IPO diagrams in DIN A4 format (landscape)
.IPO01  ANOP
&TS     SETC '..TS=+──────────────────────────────────'
&L11    SETC '+──────────────────────────────────'
&L12    SETC '+──────────────────────────────────────────────────────'
&L13    SETC '+──────────────────────────────────+'
&L21    SETC '!          Input                   '
&L22    SETC '!                Process                        '
&L23    SETC '!          Output                  !'
&L31    SETC '!                                  '
&L32    SETC '!                                                '
&L33    SETC '!                                  !'
.IPO02  ANOP
>>            Please specify IPO name (up to 8 digits)
>>&NAME
.IPO03  ANOP
....COD
....AS
....THB
 IPO: &NAME
&L11.&L12.&L13
&L21.&L22.&L23
&L11.&L12.&L13
....THE
.IPO04  ANOP
&TS.&L12.&L13
&ZAEHL  SETA  22
.LOOP   ANOP
&L31.&L32.&L33
&ZAEHL  SETA  &ZAEHL-1
        AIF   (&ZAEHL EQ 0).LOOPE
        AGO   .LOOP
.LOOPE  ANOP
&L11.&L12.&L13
.IPO05  ANOP
....TE
....AX
....COX
        MEND
```

4 Alphabetic List of TOM-DOC Commands

..*op Comment Line

The ..* command is used for comments in the input source. The text is placed directly after the asterisk * as an operand. It is neither edited nor output.

If you wish to enter a multiple line commentary, you have to enter the ..* command at the start of every line. A semicolon is considered part of the commentary. Thus it cannot be used for dividing commentary and commands.

..) Increment Current Index

The ..) command increments the current index level by 1. It is used with the ..*IN command* which switches on indexing.

..AS As-written-Mode On (1:1 output of the following text)
..AX As-written-Mode Off

The ..AS command switches the AS mode on, i.e. the automatic line break function is deactivated. The ..AX command returns you to the normal mode. Any text between the ..AS and ..AX commands is output exactly as it is entered. AS mode is necessary, for instance, for the creation of tables and when working with dummy graphics characters.

Note

– The length of the input line (default length = 72) must not exceed the value defined by the ..*CPL command* minus 2. Any text exceeding this length is truncated and a warning is output.

– The following commands are illegal in the AS mode.

..AS	*As-written-Mode On*
..AX	*As-written-Mode Off and return to text mode*
..CC=del	*Change Command Delimiter*
..CE	*Center Text*
..HM	*Hanging Margin*
..HP	*High Performance Printer Support*
..HY	*Hyphenator*
..IN;..);..IX	*Indexing Levels*
..JU;..JX	*Justify Mode On/Off*
..Li	*Level i Title*
..ML;..MR	*Set Left/Right Margin*
..P	*Paragraph with indentation*
..PSY	*Preset Syllabication*
..RL	*Reserve Lines*
..SLT	*Suppress Level in Table of Contents*
..TH	*Unnumbered Title*
..TL	*Title Level*

..CA=op Carry After On (modification marking)
..CX Carry After Off

This command serves to mark modifications in the text by setting flags of up to two characters at the right-hand margin of the printer listing. These characters are entered as the operand "op". All lines between the ..CA=op and ..CX commands are marked with the flag symbol.

– If you enter only one character as the operand, this character, prefixed by "*", is used to mark modifications. You can prevent this happening by entering the string "blank/character" (e.g. _#). If you enter more than two characters, TOM-DOC will ignore the superfluous characters.

– If a line is not to be marked as modified, the previous line must be terminated with ..*NL=0*, followed by entry of the ..CX command.

Example of modification marking

Input

```
..CA=??
The text following the ..CA command is marked on output
by two question marks at the right-hand margin until
modification marking is deactivated by means of the ..CX
command.

..CX
```

Output

```
The text following the ..CA command is marked    ??
on  output  by  two question   marks   at the    ??
right-hand margin until modification  marking    ??
is deactivated  by  means of the ..CX command.   ??
```

..CC=op Change Command Delimiter
opCC Command Delimiter Reset

The ..CC=op command serves to redefine the delimiter of TOM-DOC commands (default symbol = ..). Either 1 or 2 freely selectable characters may be specified.

Example of command delimiter redefinition

```
..CC=>
```

This command determines that any subsequent commands must be entered in the form >command.

The opCC command (in example: >CC) resets the command delimiter to the default value.

..CE_op Center Text

This command places the text specified under "op" in a line of the output file between an equal number of blanks at the left-hand and right-hand margins. Any blanks which may have been entered are regarded as normal characters and are thus also centered. The operand "op" must not exceed the length of the current line.

The text is handled as in AS mode. If there are umlauts in the text, they are converted to characters for the printer plus a blank. The blank is set at the end of the word containing the umlaut.

The command can only center the entered text. Values from system variables cannot be centered.

In the case of the &DATUM and &TIME system variables, the user must take into account the implicit length of the string (8 bytes).

..CH=del>del Change Graphic Delimiter (Modify dummy graphics character)

This command enables any character (except for ";" and "_") to be assigned the function of a dummy graphics character. You can thus inhibit conversion of characters if you wish these characters to appear in their original form in the graphics mode.

The characters are assigned using the symbol ">". The assignment is valid from when the command is given. Several assignments are possible in succession. The following format is thus permissible:

```
Z1>Z2,Z3>Z4,...
```

In the above, Z1 and Z3... are the new characters that you wish to use and Z2 and Z4 are the permanently defined dummy graphics characters. Input may be alphanumeric (length 1) or hexadecimal (length 2); ">" can only be hexadecimal. You can cancel graphic conversion explicitly by assigning a character its own function or an invalid value (e.g. X'FE').

During the conversion process, characters Z1, Z3,... are interpreted as characters Z2, Z4,... and represented accordingly. If Z2, Z4... are not permanently defined dummy graphic characters, the relevant pair of operands is ignored. You are not permitted to make two assignments to characters. If you do, the subsequent dummy graphics character inputs will not be converted.

If Z2 or Z4... is also to be converted, an unassigned dummy graphic character must be assigned to this character.

A ..CH command (without an operand) resets the assignments, and the dummy graphic characters are assigned their default functions.

The ..CH command affects whole pages since the TOM-DOC graphics module always outputs one completely edited page. The ..CH command should therefore only be reset after an explicit or implicit page break.

Note

- In order to treat an arbitrary input character in the same way as ">" during output, the corresponding hexadecimal value of ">" (6E) can be used.
- If dummy graphics characters are cancelled, the invalid values must not collide with control characters for the HP printer.
- The ..CH command is effective as of the page on which it is given, even if it follows the ..COD and ..COX commands.

– The ..CH command should only be used on the existing dummy graphic characters (see section 2.8.2 on page 54). No other characters are permitted here. The .. TR (Translate Input Character) command is available for this.

Example of dummy graphic character assignment for ">"

```
..CH=x>6E
```

causes any entered x to be treated as ">" (hexadecimal value: 6E) during graphics conversion.

Example of dummy graphic character cancellation

```
..CH=!>! or ..CH=FE>!
```

explicitly cancels the dummy graphic character "!" and excludes it from TOM-DOC graphic conversion.

Example of dummy graphic character redefinition

`..CH=/>I and ..CH=->=` and `..CH=m>%`

has the effect that the slash is interpreted as a capital I, the minus sign as an equals sign and the small "m" as a percent sign in all subsequent graphics conversions until ..CH without an operand is entered for resetting.

..CHF= { Y / N } Code Header and Footer (Graphics conversion header/footer)

The command ..CHF=Y (N=default value) enables headers and footers to be edited directly with dummy graphic representations. The user should bear in mind that these definitions have to be entered complete, including frame characters and blank lines, in order to ensure that the correct connecting lines are drawn. This is required in particular whenever a page frame is to be generated using dummy graphic characters. The command ..CHF=N resets graphics conversion in headers/footers.

Note

- The ..CHF command causes an implicit page feed.
- Graphics conversion is possible only with the ND,HP and LP 65 laser printers.

..CM=[i_[op]] Center Mark (Mid-page mark)

The ..CM command generates a center mark at the extreme left-hand paper edge which indicates where the holes should be punched in the document.

"i" is the number of the line on which the mark is to be placed. "op" is used to specify which character is used to mark the position (default is underscore "_").

..CM (without operand) marks the center of the page with an underscore.

To modify "op" only, enter

```
..CM=0_op
```

..COD Code Begin (Switch on graphics conversion)
..COX Code End (Switch off graphics conversion)

The ..COD command activates graphics conversion. It is entered immediately before the dummy graphic representations. The ..COX command deactivates graphics conversion. It is entered immediately after the dummy graphic representations.

Note

- Dummy graphic representations should be edited in the AS mode.
- Graphics conversion is possible only with the ND,HP and LP65 laser printers.

..CPL[=i] Characters Per Line (Default: 74)

This command defines the number of characters per line to be edited.

The following values are used unless otherwise specified:

```
left-hand frame character:        1
right-hand frame character:       1
blanks to the left of the text :  3
blank to the right of the text:   1
text (if CPL=74):                68 (generally: CPL - 6)
```

AS mode applies to all characters between the frame characters (i.e. to 72 characters if CPL=74).

Note

- The ..CPL command results in an implicit page feed. A subsequent ..*NP command* (new page) without an operand has no effect unless text has been output in the meantime.
- The sum of the "i" values in the *commands ..CPL, ..FM or ..FME* must not exceed 204. The following sequence is to be observed when entering these commands:

  ..FM must be entered *before* ..FME.
  ..FM and ..FME must be entered *before* ..CPL.
- ..CPL resets the right-hand frame to its default value.

..CSP[=i <op>] Character Spacing

You need the ..CSP command for bold type and for print editing for HP printers. Since the use of the command varies greatly depending on the printer type specified, a separate explanation for each printer type follows:

a) ND laser printers

..CSP=10	Character spacing: 10 characters per inch
..CSP=12	Character spacing: 12 characters per inch
..CSP=15	Character spacing: 15 characters per inch
..CSP	Character spacing: 10 characters per inch

b) HP laser printers

..CSP=10	Character spacing: 10 characters per inch
..CSP=12	Character spacing: 12 characters per inch
..CSP=15	Character spacing: 15 characters per inch
..CSP	Character spacing dependent on the default of the selected character set

c) LP65 laser printers

In the case of the 3365 laser printer, explicit character spacing cannot be specified, only additional character spacing. The spacing between two adjacent characters can be increased or reduced in steps of 1/440 inch.

You can use ..CSP[=i <op>] to set character spacing, where i can be a value from 0 to 999 and op can be either plus (+), which is the default, or minus (-).

"..CSP=i +" increases the character spacing, and "..CSP=i -" reduces it.

Example:

```
/PRINT filename,SPACE=E,RESOURCE=LP65(PAGEDEF=246,
DUPLEX=YES),FORM=form
```

Notes:

– When using the ..CSP command with either ..DEV=HP or ..DEV=LP65, you must note the following:

  The ..DEV=HP or ..DEV=LP65 command must be entered before the ..CSP command. The control character section at the beginning of the print line is generated as follows:

  (1) If ..CSP is entered with an operand, TOM-DOC generates the character spacing in the control character section with:

  X'04' for character spacing of 10 characters per inch (..CSP=10)
  X'08' for character spacing of 12 characters per inch (..CSP=12)
  X'0C' for character spacing of 15 characters per inch (..CSP=15)

  (2) If the ..CSP command is entered without an operand, TOM-DOC generates the value X'00' for the character spacing. The character spacing is dependent on the character set selected in the PRINT command; in other words, the size of the selected character set is taken into account automatically during the printing operation.

– All output files edited by TOM-DOC can be printed on BS2000 systems by means of the BS2000 /PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (the product RSO) and printed from there.

  As of SPOOL V3.0, it is still possible with the BS2000 product DPRINT to print the edited TOM-DOC output files on remote printers (e.g. including printers connected to SINIX systems) without RSO and the printer emulation on SINIX.

  You will find relevant information on the correct selection of devices

and paper sizes, the use of suitable character sets, the control of character and line spacing, and the monitoring of SPOOLOUT jobs in chapter 5, "Printing with TOM-DOC". This contains detailed information that helps you to use the extensive printing options for TOM-DOC files efficiently.

..DEV=op Device Type (output device)

By means of this command TOM-DOC supports the editing of output files for output to different printer types. The "op" operand specifies the desired printer type.

The following table shows you the printer types that are currently supported and the command operands required to select them:

op	Printer	Printer type
9001 / 9001S	TRANSDATA 9001-31 Printer *)	
9001-2 / 9001-2S	TRANSDATA 9001-2 Printer *)	
9003	TRANSDATA 9003 Printer	Ink-jet printer
9004	TRANSDATA 9004 Printer	Dot-matrix printer
9012	TRANSDATA 9012 Printer	
9013	TRANSDATA 9013 Printer	
9022	TRANSDATA 9022 Printer	Laser printer (single-sheet handling)
ND	ND 3350/3352 Laser Printer	
HP	HP 3351/3353 Laser Printer	
LP65	3365 Laser Printer	Laser printer

*) If the 9001 Printer is run on a 9750 Data Display Terminal or via the printer emulation on SINIX systems, 9001S or 9001-2S must be specified as the operand.

The operand can also be specified in the DOCL selection screen in the "DEVICE-TYPE" field. This makes it possible to overwrite an existing ..DEV command temporarily (see DOCL, section 2.4).

Note:

- The command must be entered before the first text input.
- It is generally not necessary to specify the ..DEV=ND command (it is the default); it is only generated or evaluated if specified in the DOCL selection screen (see DOCL, section 2.4).
- The effect of ..DEV=HP is identical to that of the ..HP command (see also the ..HP command in chapter 4).
- When the 90xx printer types are selected, you can manipulate type styles with the ..TD command (see also the ..TD command in chapter 4).
- The conversion of TOM-DOC dummy graphic representations with the ..COD and ..CHF commands is only possible and useful in the case of the ND, HP and LP65 laser printers.

All output files edited by TOM-DOC can be printed on BS2000 systems by means of the BS2000 /PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (the product RSO) and printed from there.

As of SPOOL V3.0, it is still possible with the BS2000 product DPRINT to print the edited TOM-DOC output files on remote printers (e.g. including printers connected to SINIX systems) without RSO and the printer emulation on SINIX.

You will find relevant information on the correct selection of devices and paper sizes, the use of suitable character sets, the control of character and line spacing, and the monitoring of SPOOLOUT jobs in chapter 5, "Printing with TOM-DOC". This contains detailed information that helps you to use the extensive printing options for TOM-DOC files efficiently.

..EM Emphasize On (bold-face print)
..EMX Emphasize Off
..EMF=del Emphasize Flag
..EMT Emphasize Title

These commands support bold face printing. Text segments which are to be emphasized by bold type can be written between the ..EM and ..EMX commands. Alternatively, the emphasize flag (..EMF=del) can be entered in the text immediately before and after the text segment to be emphasized. The start/end delimiters must always be entered in pairs. In edited text, the flag character is replaced by a blank. The flag is reset by entering the ..EMF command without operands. The ..EMT command causes all titles to be printed in bold type.

Note

- Emphasize flags in diagrams are only interpreted if the diagram is a simple graphic representation which can be converted by the TOM-DOC graphic module without overlay lines. Otherwise graphics conversion has priority over emphasis.
- The emphasize flag effect stops at the end of a line.
- Bold type is not supported for footnotes and tables.
- In the event of a clash, underlining has priority over bold type.
- The ..*CSP command* must be entered before specifying bold type (see also the ..*CSP and* ..*HP commands*).

..EN Exposed Numbers

This command causes title numbers to be printed before the text blocks. The decimal classifications are printed on the left-hand margin while the titles are aligned with the following text so that title and text are left-justified. To this end, the ..*ML=i command* must be issued to define a relative left-hand margin. This margin should be large enough to accommodate the largest decimal classification. If this is not the case, the title text is indented as required. The subsequent text again starts on the relative margin.

Example

```
1.      Title
1.1     Sub-title
1.1.1   Title on the second level

        The block of text you have typed appears
        after the title. This text is now followed
        by more. And so it goes on.
```

..FC=del Frame Control Character

This command serves to define the character used as the right-hand and left-hand delimiter of the text body and of the header and footer delimiters (unless these are explicitly defined by ..HD=i op and ..FT=i op). "op" is entered either in alphanumeric form with length 1, or in hexadecimal form with length 2; in the latter case, only uppercase letters are permitted for A-F.

..FM[=i] **File Margin (i = 0-20, default value = 10)**
..FME[=i] File Margin Even

These commands define the width of the filing margin. The ..FM command indents the filing margin i characters on every page. The ..FME command has the effect of indenting even-numbered pages by i characters. The commands are set between 0 and 20 before the first TOM-DOC text entry is made and remain valid until the end of the text. The default value for "i" is 10 columns. This command is used, for instance, for editing book texts so that the type area on an even page conforms to that of an odd page.

Note

- The sum of the values assigned under "i" in the *..FM, FME and ..CPL commands* (characters per line) must not exceed 204.
- The following sequence is to be observed when entering these commands:

  ..FM must be entered *before* ..FME.
  ..FM and ..FME must be entered *before* ..CPL.

..FN Footnote On
..FNX Footnote Off

The ..FN and ..FNX commands define footnotes which should appear in the lower part of the page. Any text between the two commands is interpreted as a footnote.

Note

– Footnotes should be entered immediately after the text to which they refer in order to make sure that they are output on the same page as the text.

– Count variables in footnotes are not incremented.

– If you use footnotes in your document, you should bear in mind that TOM-DOC always edits the texts entered by page, not by line. Since the layout of a page in TOM-DOC also depends on a great many other factors (e.g. ..LPH, ..LPF, ..LPB, ..CPL, ..RL, several or very long footnotes on a page), the result may not always be satisfactory.

  You should also take into account that the texts that are to appear as footnotes really are only text; there is no underlining and no bold type. For example, the use of the ..FN and ..FNX commands in index levels not yet concluded with the ..IX; command leads to an incorrect result.

– However, through the skillful use of the TOM-DOC commands through alternative coding, for example - you will virtually always be able to obtain an acceptable result. The following examples are designed to illustrate this.

Example 1: footnote definition

Input

```
Footnotes (&T1)
..FN
(&T1) Footnotes are explanatory texts
..FNX
are printed at the bottom of the page to
which they refer (see bottom of this page).
```

Output

Footnotes(1) are printed at the bottom of the page to which they refer (see bottom of this page).

(1) Footnotes are explanatory texts

Example 2: *Inclusion of ..FN and ..FNX in an index level that is not yet concluded*

Input

```
..IN;
TEXT WITH FOOTONTE (&T1)
..FN;
..HM=6 (&T1);
FOOTNOTE
..HM;
..FNX;
IS FORMATTED INCORRECTLY.
..);
```

Output

TEXT WITH FOOTNOTE (1) IS FORMATTED INCORRECTLY.

2. The index level is incremented and the opening of the index level with ..IN; is included in the footnote. This is wrong.

1.
(1) FOOTNOTE

You can avoid this error by means of alternative coding.

Example 3: *Avoiding the inclusion of ..FN and ..FNX in an index level*

Input

```
..CPL=50
..IN;
TEXT WITH FOOTNOTE (&T1)
IS FORMATTED CORRECTLY.
..IX                The opened index level is concluded here.
..FN;               ..FN is no longer in the index level.
..HM=6 (&T1)
FOOTNOTE
..HM;
..FNX;
```

Output

1. TEXT WITH FOOTNOTE (1) IS FORMATTED CORRECTLY.

_____ This is the desired result.

(1) FOOTNOTE

If you want more than one item in a list, add to the above coding.

Example 4: More than one item in index levels

Input

```
..IN;
TEXT WITH FOOTNOTE (&T1) IS FORMATTED CORRECTLY.
..IX
..FN;
..HM=6 (&T1)
FOOTNOTE
..HM;
..FNX;
..IN=2;                    Next item with an explicitly
                           set index
NEXT ITEM IS FORMATTED CORRECTLY.
..IX
```

Output

1. TEXT WITH FOOTNOTE (1) IS FORMATTED CORRECTLY.
2. NEXT ITEM IS FORMATTED CORRECTLY.

_____

(1) FOOTNOTE

..HD=i_op Headline Odd and Even (odd and even pages)
..HDE=i_op Headline Even (even pages only)
..FT=i_op Footline Odd and Even (odd and even pages)
..FTE=i_op Footline Even (even pages only)

These commands serve to define headers and footers and the text to be output on these lines. "i" defines the line number and "op" the text which is to appear on this line. If "op" is omitted, a blank line is generated. Headers and footers can be redefined using these commands between pages also. The new definition remains valid until it is changed again. If ..HDE/..FTE is not specified, the entry for ..HD/..FT also applies to ..HDE/..FTE. To print headers and footers on odd pages only, you must first define them using ..HD/FT and then specify blank strings for the even pages using ..HDE/..FTE (see Note).

There are several rules that should be observed when you specify the operand:

- If "op" is entered without apostrophes (’), it is left-justified.
- If "op" is entered with apostrophes (’), it is justified in AS mode.
- If a frame character was specified using the *..FC command*, the operand "op" is aligned with the margin and the frame character overwritten. You can prevent this by entering a blank "_" as the first character of the operand and enclosing the operand in apostrophes.

Note

- These commands cannot be terminated with a semicolon, as the semicolon is regarded as part of the text.
- If more than one definition is entered on one page for the same header/footer, i.e. in the event of redefinition before page feed, only the commands entered last before the page feed are interpreted.
- The ..HD/..FT commands have priority over the ..HDE/..FTE commands.
  Any ..HD/..FT command entered after an ..HDE/..FTE command and referring to the same header/footer resets any previously entered ..HDE/..FTE command. The "op" of the ..HD/..FT command entered last applies to the page on which the command was entered as well as to all subsequent pages.
  If the ..HDE/..FTE operand is to continue to apply to even-numbered pages, the ..HDE/..FTE command must be repeated.

- In the case of keyword indexes and tables of contents, the ..HD/..FT command entered last always applies (see the *..HIX and ..TOC commands*), even if the index/table of contents precedes the command.
- Headers/footers can be more easily defined in the prompt menu of the edit command *DOCL. The ..HD/..FT commands are issued implicitly in this case.
- Only as many headers and footers are generated as are specified by the operands of the ..LPH and ..LPF commands.
- Blanks are permitted before and after the equals sign.
- Index flags, bold type and underlining are not possible in headers and footers.

..HIX[=op] Header of Index

This command controls the output of a keyword index. It implicitly causes an automatic page break at the position in the text where it is entered. The value of the page count at the time the ..HIX command was entered is taken as the start value for the keyword index. The start value may have to be reset for subsequent text using the *..SET command*, taking into account the length of the keyword index. The keyword index is compiled at the end of the TOM-DOC editing process. The optional operand "op" is output as the index header. If no operand is specified for the header, TOM-DOC writes a standard text to the table of contents. Any commands which are to apply to the keyword index (e.g. header/footer definitions) must be entered at the end of the text input (except for *..QPN* Qualified Page Number).

Note

- For a description of how to mark text to be included in the keyword index, consult the sections on the *..IF (Index Flag) and ..SI (Set Keyword Index) commands*.
- The keyword index can be sorted by the *..ISF command*.
- References in the keyword index can be classified using the *..RIX command*.
- The keyword index should always come after the table of contents in your document. If it does not, the page numbering will not be continuous throughout the whole document and some numbers will be repeated.

..HM=i[_op] Hanging Margin

This command shifts the left-hand margin "i" columns to the right, relative to the text. The optional operand "op" is aligned with the original left-hand margin so that there is space for a keyword, for instance, next to the indented text. The length of the operand must not exceed the value "i", and "i" must not result in the right-hand margin being exceeded. Any subsequent text is indented "i" columns until this status is reset by another ..HM command.

Note

- An ..HM command without an operand resets the column counter to the relative left-hand margin (i.e. to the value of the last ..ML command).
- The *..ML, ..Li and ..TH commands* reset the ..HM status.
- A preceding *..NL=0 command* suppresses the blank line automatically generated by the ..HM command.
- If several "..HM=i op" commands are entered one after the other, you should reset the HM status by entering the ..HM command without operands. If you do not, you do not get a feed to the next line.
- The ..HM command does not work in AS mode.

Example of a hanging margin

Input

```
The text block starts in column 15.
..ML=20
The text block starts in column 20.
..HM=10 Point1:
This is the text under Point1, which may extend
over more than one line. The operand is shifted
to the previous margin in front of the text block
by means of the ..HM command.
```

Output

```
The text block starts in column 15.
    The text block starts in column 20.

    Point1:   This is the text under   Point1,   which
              may extend over more than one line.  The
              operand   is shifted to   the   previous
              margin   in  front of  the text block by
              means of the ..HM command.
```

..HP High Performance Printer Support (Laser printer)

This command supports the editing of TOM-DOC input files with control characters applying to ND2/3-HP laser printers (device types 3351 and 3353). The device types 3351 and 3353 do not support the overprint function of types 3350 and 3352, but they offer the option of using several character sets, each of which may comprise up to 255 characters.

Note

- The command must be issued before the first text input.
- This command has the same effect as the specification of ..DEV=HP (see section 4, ..DEV command).
- Always enter the command in conjunction with ..CSP.
- When editing with ..HP, make sure the code X'FF' in the TOM-DOC input file is not edited, for this code is used as an escape symbol for HP control characters and may lead to errors.
- All output files edited by TOM-DOC can be printed on BS2000 systems by means of the BS2000 /PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (the product RSO) and printed from there.

  As of SPOOL V3.0, it is still possible with the BS2000 product DPRINT to print the edited TOM-DOC output files on remote printers (e.g. including printers connected to SINIX systems) without RSO and the printer emulation on SINIX.

  You will find relevant information on the correct selection of devices and paper sizes, the use of suitable character sets, the control of character and line spacing, and the monitoring of SPOOLOUT jobs in chapter 5, "Printing with TOM-DOC". This contains detailed information that helps you to use the extensive printing options for TOM-DOC files efficiently.
- The ..HP command is not expected to be supported in the next version of TOM-DOC. It will be replaced by the ..DEV=HP command.

..HY=del Hyphenator

This command defines a character to be used for the marking of potential word divisions. If, during justification, TOM-DOC determines that a word should be split up, the division is made at the marked place. If splitting up is not required, the hyphenator is ignored. The combination "del-" marks any existing hyphen as the point of division, ensuring at the same time that the hyphen is printed only once if the word is split. The hyphenator "del" before a word excludes this word from automatic word division. Entry of the ..HY command without an operand resets the hyphenator.

Examples of semi-automatic word division

The hyphenator is defined by means of ..HY=%.

The character string " hyphen%ation " is split up at the marked place if required for a line break.

The character string " semi%-automatic " is split up before "automatic" if required, the hyphen being printed only once at the end of the line.

The character string " %TOM-DOC " is excluded from automatic word division.

..IF=del Index Flag (Mark keyword)

The ..IF command defines a character to be used for marking words or character strings which are to be included in the keyword index. This index flag is input immediately before and after the word/string to be marked. In the output, "del" is replaced by a blank. It is therefore relatively simple to mark keywords for the index at a later stage by overwriting the blanks before and after a word by "del" (see the *..SI command*). Entry of an ..IF command without an operand resets the index flag.

Note

- No shifting of blanks occurs in AS mode.
- If the end of the keyword is not marked by a second "del", an end delimiter is set automatically at the end of the line.
- Any word/string marked by the index flag that exceeds 40 characters is truncated.
- The index flag cannot be used in the operands of the ..HD, ..HDE, ..FT and ..FTE commands.

..IN[=i] Index On
..) Increment Current Index
..IX Index Off

The ..IN command opens an index level and starts current numbering of this level with the digit specified for "i" or with the i-th letter of the alphabet. If the operand is omitted and no index level exists, numbering starts with index 1 on level 1. If an index level has already been defined, a new index level with the number of the existing index level incremented by 1 is opened. Up to 10 levels are possible. The entire text of an index level is indented 5 columns to the right, relative to the left-hand margin used up to that point or to the left-hand margin of the previous index level. The right-hand margin is shifted once 3 columns to the left the first time an ..IN command is issued.

The ..) command increments the current index level by 1. The ..IX command terminates the current index level, any subsequent ..) command refers to the previous level.

Note

- A blank line is generated before each ..IN, ..) and ..IX command.
- The following formats are defined for the index levels:
  (1) a digit from 0 through 999 followed by a dot for index levels 1, 5 and 9
  (2) a digit from 0 through 99 enclosed in parentheses for index levels 2, 6 and 10
  (3) a letter from A through Z followed by a dot for index levels 3 and 7
  (4) a letter from A through Z enclosed in parentheses for index levels 4 and 8
- The ..*Li command* (title) terminates all index levels.
- If the highest possible value is exceeded in one of the index values as a result of incrementation with the ..) command, the index level begins again with its initial value.

Example

Input

```
..IN;first item - first level
..);second item - first level
..IN=4;fourth item - second level
..IN;first item - third level
..IN=5;fifth item - fourth level
..IX;..*end fourth level
..IX;..*end third level
..);fifth item - second level
..IX;..*end second level
..);third item - first level
..IX;..*terminate enumeration
```

Output

```
1.  first item - first level

2.  second item - first level

     (4) fourth item - second level

           A. first item - third level

                (E) fifth item - fourth level

      (5) fifth item - second level

3.  third item - first level
```

Note

If you wish to cross-refer to certain items on a specific level, then you have to use the count variable &Tn for this level.

..ISF=op[,op] Index Sort Fields (op=(<sortpos>,<sortlen>))

The keyword index can be sorted. The ..ISF command serves to define sort position and sort length. Unless otherwise specified, the entire keyword index is sorted in ascending order. Up to 5 sort fields can be specified.

Example

The command ..ISF=(5,10) results in the words/strings marked by ..SI= (see the ..*SI command*) or the index flag (see the ..*IF command*) being sorted in the keyword index starting with the 5th character with a length of 10.

..JU Justify Mode On (Right-hand margin justification)
..JX Justify Mode Off

These commands serve to activate (default value) and deactivate automatic justification to the right-hand margin.

..Li_op Level i Title Header

This command serves to generate titles of levels 0 through 9 (i=0,1,...,9). The text of the title is specified as "op". If requested, these titles are subsequently automatically included in the table of contents during editing by means of the ..*TOC command*. Unless otherwise specified, 2 blank lines are inserted before and 1 paragraph indent after any title generated with this command.

Note

- Any previous ..NL=2 command has no effect unless text is output in between.
- An automatic page break takes place unless at least 5 more lines fit on the current page.
- The ..*Li command* terminates all index levels (see the ..*IN command*) and resets ..HM.
- The numbers of the titles generated with ..Li can be printed to the left of the text body by means of the ..*EN command*.
- A conditional page break before a title can be defined using the ..*TL command*.
- The "op" operand should always be specified.
- Counter and reference variables cannot be used in titles.

..LL= { Y / N } Line Leading On / Off (Leading lines)

The command ..LL=Y generates dotted leading lines between titles and reference specifications in the table of contents and between keywords and reference specifications in the keyword index. The default value is ..LL=N.

..LPB=i Lines Per Body (i = 10-160, default 44)
..LPF=i Lines Per Footer (i = 0-16, default 8)
..LPH=i Lines Per Header (i = 0-16, default 8)

These commands define the page layout of the text. The page layout can be redefined for each page in the source. Page layout is described in detail in sections 2.4 and 3.2.

Note

- The commands ..LPH, ..LPF and ..LPB result in an implicit page feed.
- Any subsequent ..NP command without an operand has no effect unless text has been output in between.

..LSP=i Line Spacing (Line feed in the text, i = 1-4)

This command causes a line feed of "i" lines after each line of the current text (the default is 1, a line feed to the next line).
"i=0" has the same effect as ..LSP=1.

..MB=del Metablank Character

This command defines a character which is converted to a blank in the output file. This serves to prevent required blanks from being deleted in the output file. Entry of the ..MB command without an operand resets the metablank character. Metablanks are used, for instance, for spaced print or for the arbitrary insertion of blanks.

Note

Words/strings containing a metablank character are not split up by a line break.

– The """ character (≙X'FF') must not be used as a metablank character because TOM-DOC uses this character internally as a terminating character for a string.

Example

The character string "100 000.-- DM" is to be output exactly in this form (i.e. including the two blanks) and must not be split up as a result of a line break. It is entered in the source as follows, assuming that "#" has been defined as metablank character (..MB=#):

```
100#000.--#DM
```

..ML[=i] Margin Left (i = 1,2,....,(MR-2))
..MR[=i] Margin Right (i = (ML+2)....(CPL-6))

These commands explicitly define the position of the right-hand and left-hand margins. Entry of the commands without operands resets the specified values.
(Default value ML: i=1; default value MR: i=CPL-6)

Note

– Any setting defined by ..MR is canceled by ..CPL.

– ..ML and ..MR are not valid in AS mode.

..NL[=i] New Line (Blank lines)

This command closes the current line and generates "i" blank lines. ..NL=0 positions to the next line without generating a blank line and continues output in this line. ..NL without an operand has the same effect as ..NL=1.

Note

..NL=1 has no effect whenever a previous command has already generated an implicit blank line.

..NP[=i] New Page (Empty pages)
..NPE New Page Even
..NPO New Page Odd

The ..NP=i command generates "i" blank pages. If issued without an operand it causes a page feed to the next page. ..NPE causes a page feed to the next even-numbered page, ..NPO to the next odd-numbered page. The commands ..NPE and ..NPO have no operands.

..NPT New Page In Table Of Contents

This command marks an intended page break in the **table of contents**.

..OPF=op Output File

This command allows you to specify an output file name in the source file when you start TOM-DOC with *DOCE, *DOCF or *DOCL. Outputs are consequently made to this file, i.e. TOM-DOC assigns SYSLST the file specified in ..OPF.

..P Paragraph

This command generates a new paragraph.
A blank line is inserted automatically. The next line is indented by the current value of the ..PIN command (default value = 5). A page feed is carried out unless at least 2 lines are left on the current page.

..PIN[=i] Paragraph Indentation Amount (i = 0-9, default value = 5)

This command defines the number of columns for the indentation at the beginning of a new paragraph.

..PSY= i [A|B] Preset Syllabication (default: 2) (length of split words)

This command defines the minimum length "i" of the first or last syllable of a word that is split off when the automatic division takes place. The default for "i" is 2. The value i can be set to any number between 1 and 99. The minimum length of the part of
the word after the hyphen (= last syllable) is defined by the operand A; operand B is for the minimum length before the hyphen (= first syllable). If no operands are set, the length of the first and last syllable is set to "i". The ..PSY command without operands sets the first and last syllable lengths to the default value 2.

Note

The automatic hyphenation program does not split off any single letters in front of a word, as German language hyphenation (see section 2.7.1) does not permit this.
For this reason, the input "i=1" is changed to the default value 2.

..QPN=op Qualified Page Number

The command ..QPN=op qualifies page numbers in tables of contents and keyword indexes. For this purpose, either of the system variables &Tn or &LN0 should be specified for "op" wherever a page count is reset by means of the ..*SET command*. (If a value other than 0 is specified for "n" in &LNn, the default value &LN0 <n=0> is set automatically.) Directly following the system variables, additional alphanumeric characters may be specified for "op", which form the prefix together with the system variable. The user should bear in mind that such a combined prefix consisting of a variable and alphanumeric characters should not exceed a length of 3 characters. The prefix defined with ..QPN=op is placed in front of the page numbers in tables of contents or keyword indexes. Instead of the system variables, up to three freely selectable alphanumeric characters may be specified for "op", which will then form the prefix. This permits qualified page numbering in tables of contents and keyword indexes.

Note

- Qualifications for headers and footers must be implemented by the user.
- The command implies a page feed.
- An ..NP command without an operand has no effect unless text is entered between the ..QPN and the ..NP commands.

Example

Input:
1. ..QPN=&LN0X
2. ..QPN=&TnX
3. ..QPN=XXX

X can be any alphanumeric character.

Examples of input

Example1

```
..QPN=&LN0-
..L0 chapter 1
..NP
..L1 chapter 1a
..NP
..L1 chapter 1b
..NP
..SET &S1=1
..L0 chapter 2
..NP
..SET &S1=1
..L0 chapter 3
..TOC
```

Example2

```
..L0 chapter 1
..NP
..L1 chapter 1a
..NP
..L1 chapter 1b
..NP
..SET &S1=1
..L0 chapter 2
..NP
..SET &S1=1
..L0 chapter 3
..TOC
```

Example3

```
..QPN=abc
..L0 chapter 1
..NP
..L1 chapter 1a
..NP
..L1 chapter 1b
..NP

..L0 chapter 2
..NP

..L0 chapter 3
..TOC
```

Examples of output

Example1: Table of contents with the ..QPN command produces the following prefixes before the page numbers:

```
1    Chapter 1                    1-1
1.1  Chapter 1a                   1-2
1.2  Chapter 1b                   1-3
2    Chapter 2                    2-1
3    Chapter 3                    3-1
```

Example2: Table of contents without the ..QPN command

```
1    Chapter 1                      1
1.1  Chapter 1a                     2
1.2  Chapter 1b                     3
2    Chapter 2                      1
3    Chapter 3                      1
```

Example3: Table of contents with the ..QPN command and an explicit prefix

```
1    Chapter 1                    abc1
1.1  Chapter 1a                   abc2
1.2  Chapter 1b                   abc3
2    Chapter 2                    abc4
3    Chapter 3                    abc5
```

..RIX=op Reference Of Index

The command ..RIX=op defines the references to be specified in the keyword index. The following entries for "op" are possible:

```
op  =  P          Page      (default value)
       C          Chapter
       CP         Chapter - Page
       PC         Page - Chapter
```

In the case of a combination of page and chapter, the main chapter is indicated.

..RL=i Reserve Lines (i=1,2,....,160) (conditional page feed)

This command causes a page feed to be executed only when there is no more space for "i" lines on the current page.

Reservation of "i" lines is only effective if the command ..NL=i is specified immediately before this command with the number of lines to be reserved.

The command ..RL=i ensures that the "i" lines remain on one page in a single data block.

If there is no previous ..NL=i command, this command does not reserve any lines.

..ROT=i Rotation (page rotation)

The command ..ROT=i rotates pages. You can set the angle of rotation to 0, 90, 180 and 270 degrees by means of the i operand. The command can be set anew for each print page. It can currently only be used for HP laser printers.

The operand ROTATION=YES must be specified in the PRINT command.

..RS= { Y / N } Reset System Variables

At the beginning of automatic table of contents and index generation, the system variables for titles (&LN, &LC, &LT) are reset. You can prevent this with the command ..RS=N. This preserves title variables - in headers and footers, for example set at the end of the input file.

You reset the default mechanism (i.e. that all variables are reset) by specifying ..RS=Y or ..RS (i.e. no operand).

..SET_sysvar=nnnn Set System Variable
..SET_&RFxxxxx=sysvar Set Reference Variable
..SET_&PA=i Set Page (page counter incrementation)

The ..SET command is used to insert symbolic parameters at any position in the text. A detailed description of the use of symbolic parameters can be found in section 2.6.

Formats of the different variables

1. For &S1, &S2, &S3, &P1, &P2 the following applies:
   ..SET sysvar=nnnn (nnnn = 4-digit number)
2. For &Tn, &LNi the following applies:
   ..SET sysvar=nnn (nnn = 3-digit number)
3. For &LTi the following applies:
   ..SET &LTi=op (op = up to 256 characters of text)
4. For reference variables the following applies:
   ..SET &RFxxxxx(nn)=sysvar

   (1) xxxxx may be either uppercase letters, digits or special characters with the exception of the characters ";", ">", "=" and "&" (or the current delimiter).

   (2) At least one and not more than five characters may be specified for xxxxx.

   (3) nn specifies the length (number of positions) of the variable (default value = 3). The user should ensure that the defined length of the variable permits a reference to be made to the full length of the system variable assigned.

   (4) A reference variable can be set no more than once per TOM-DOC run.

   (5) Reference variables are enclosed in blanks and thus set apart from the text.

   (6) Error messages referring to variables ("Unsatisfied Reference") show the last line as source reference, since they do not necessarily obtain their actual value until the end of the text.

Note

- If new symbolic parameters are to be defined for a new page, these must be set after entry of the ..NP command.
- When defining the value of system variable &LCi, the ..SET command must be issued for each individual level of &LNi.
- Symbolic parameters can be included in any character string other than blanks. They must not occur more than once in the same character string except in headers and footers and in AS mode.
- The operand &PA=i increments the current value of the page counter by the amount i. The operand thus effects the system variables &S1, &S2, &S3, &P1 and &P2. The operand &PA=i can only be used in the ..SET command; it cannot occur in the text as a reference.

..SF[=del] Special Flag (applies to German umlauts only)

This command serves to define a flag for the marking of umlauts. Since the umlauts available on terminal keyboards do not always generate the code defined for the printer, umlauts may be marked using this special flag. The flag is replaced by a blank at the end of the word. This may not be desirable, in particular in AS mode. In this case, it is recommended to work in the editor code mode. Not specifying an operand resets the special flag.

Note

If you will be printing on TRANSDATA 90xx printers (RSO printer), a conversion table must be specified in the PRINT command as of RSO V2.0A (see section 5).

Example of flagging umlauts

Input

```
..SF=$
Sie ko$nnen die Stras$e ungefa$hrdet u$berqueren.
```

Output

Sie können die Straße ungefährdet überqueren.

..SI=op Set Keyword Index (Enter keyword)

The command ..SI=op serves to define keywords which either do not occur in the current text, or do not occur in the form required for the keyword index. The "op" is then included in the keyword index (see also the..IF command).

Note

- Any word/string defined in "op" that exceeds 40 characters is truncated in the keyword index.
- "op" may consist of digits, letters and special characters with the exception of semicolon.
- Keywords containing umlauts only appear in the correct place in the index if they are generated using the ..SF or ..IF command or the CODE statement in EDT.

Example of keyword definition

The command ..SI=keyword causes the string "keyword" to be included in the keyword index when the index is being edited with the *..HIX command*.

..SLT=i Suppress Level In Table Of Contents (Title output)

This command excludes titles of the level "i" (as well as any lower-order levels) from the table of contents. Unnumbered titles (see the *..TH command*) are assigned to level 9.

..SRC= { Y / N } Print Source Line (Line numbering)

As a result of the command ..SRC=Y, the reference to the corresponding line number in the source file is printed at the left-hand margin of the output file.

Note

- Members/files copied into the source file using ..SW are not resolved, i.e. the reference for all output lines generated in this way is to the input line of the ..SW command.
- In certain cases, the mechanics of buffered output cause the source line numbers to be 1 below the correct value.

..SW= { op[,plamtype] / op[,[plamtype],version] } Switch Input (switch input file or element)

..SWL=op Switch Input Library

These commands enable the user to process more extensive documents with TOM-DOC without overloading the editor's virtual memory. Externally stored text segments can be copied into the TOM-DOC input file. For ..SW=op, "op" must be a member of an FMS library, a member of a PLAM library, or a BS2000 file. If an element from a PLAM library is specified for the ..SW command, type can be specified as well. The types "S", "M", "D", "P", "J" and "X" or the maximum 8-character entry of a user-defined type are permitted. If you do not make an entry, the default type "S" is used.

If a version is specified, a member of an LMS or FMS library managed by version can be accessed.

The name of the appropriate FMS/PLAM library is to be specified as "op" in the ..SWL command. ..SW commands may also be nested: more than one ..SW/..SWL command may occur within the member or file being read in (maximum nesting level: 9). For BS2000 files which have been copied in and members of an FMS library managed according to versions, there is no nesting option.

Note

- In the case of editing with *DOC, these commands are not supported.
- When copying BS2000 files, nesting is not possible; i.e. as long as the ..SW input from a BS2000 file is not complete, no other input from a BS2000 file is permitted.
- When copying FMS members, nesting is not possible; i.e. as long as the ..SW input from an FMS member is not complete, no other input from an FMS member is permitted.
- The ..SW command is useful if you wish to store definite standard formats for your texts in a file or a library. ..SW can then be used to insert these format commands in your current text.
- If a plam type and a version are specified, the versioned input element of the selected type is searched for in an LMS library.

  If `..SW=op,,version` is specified, the omission of the plam type and

the entry of a comma mean it is not clear whether a versioned member is to be searched for in an LMS library or an FMS library. In this case, FMS functions are used before LMS functions to access the member.

Because of this search strategy, you should never specify the plam type, which is irrelevant anyway, when copying in members from an FMS library.

To copy in versioned FMS members, please use the syntax ..SW=op,,version, as you have done up to now.

If the member specified in the ..SW command is not found in the library specified in the ..SWL command, TOM-DOC automatically looks for a BS2000 file with the relevant member name.

TOM-DOC then copies this file into the text without a warning.

..SYL= { Y / N } Syllabication (Automatic hyphenation)

Automatic hyphenation is activated/deactivated by means of the ..SYL command (default value = Y).

..SYM=del Symbol For Variable (Change variable symbol)

This command serves to redefine the delimiter of variables (default value = &).

..TCL[=i] Table of Contents Length (Insert table of contents)

This command achieves correct continuous numbering, without having to explicitly set a page counter, of the continuation page of a table of contents not located at the beginning of the output file. The "i" operand serves to specify the expected number of pages for the table of contents. If the specified number differs from the number of pages calculated by TOM-DOC, consistent pagination is effected in an automatic trailer run.

Note

- If /EXEC TOM.DOC is used to call the phase, the trailer run only takes place if a BS2000 file has been assigned with SYSDTA.
- The ..TCL command must precede the ..TOC command.

..TDi TRANSDATA Printer Support

A number of different type styles are available for the decentralized TRANSDATA printers, as well as for the laser printers connected centrally to BS2000 processors. The printers supported by TOM-DOC are listed in the description of the ..DEV command. The commands required to select the various print types are contained in the following table:

Printer type	Command					
	TD1	TD2	TD3	TD4	TD5	TD9
9001/9001S	———	bold	italics	RESET	———	wide
9001-2/9001-2S	———	———	———	RESET	———	wide
9003	black	red	italics	RESET	ZV2	wide
9004	———	bold	shaded	RESET	———	———
9012	———	bold	———	RESET	———	wide
9013	———	bold	———	RESET	———	wide
9022	———	shaded	———	RESET	———	———

Explanations

black -- black print
red -- red print
italics -- italic print
shadow -- shadow print
raster -- raster print
reverse -- reverse print
bold -- bold print
wide -- wide type (double character width)
ZV2 -- switchover to second character set
RESET -- reset all functionss (reset printer settings to default values)
---- -- no effect (no functions are executed)

Note

- It is absolutely necessary that the ..DEV command be entered prior to this command and a 90xx TRANSDATA Printer be specified as device type.
- A combination of print styles is possible.
- Simple texts can also be underlined, with the following exceptions:

  a) Titles which make use of TD commands *cannot* be underlined.
  b) Certain *restrictions* apply to underlining wide type.
- The various type styles are not supported in footnotes, tables, headers, footers, the table of contents or index.
- TOM-DOC commands cannot be used to set character or line spacing. (Default values are used.) The appropriate default values can be defined by selecting suitable character sets using RSO (parameter CHARS of the /PRINT command).
- All output files edited by TOM-DOC can be printed out on BS2000 systems using the /PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (using the product RSO) and printed out from there.

  As of SPOOL V3.0, it is still possible with the BS2000 product DPRINT to print out the edited TOM-DOC output files on remote printers (e.g. including printers connected to SINIX systems) without RSO and the printer emulation on SINIX.

  You will find relevant information on the products and their versions required on the BS2000 systems supported by TOM-DOC, the BS2000 commands, the correct selection of devices and paper sizes, the use of suitable character sets, the control of character and line spacing, and the monitoring of SPOOLOUT jobs in chapter 5, "Printing with TOM-DOC". This contains detailed information that helps you to use the extensive printing options for TOM-DOC files efficiently.
- The ..TD5 functionality (switchover to second character set) is not implemented for the 9012 and 9013 printers.

..TE Table End

This command terminates the last table section or the entire table (see *..THB, ..THE and TS commands*).

..TH_op Unnumbered Title Header

This command generates an unnumbered title which is included in the table of contents.

Note

- The ..TH command terminates all index levels and resets the ..HM command.
- It causes an automatic page feed unless five more lines will fit on the current page.

..THB Table Header Begin
..THE Table Header End

Tables often extend over more than one page. TOM-DOC carries out the appropriate table breaks together with the page breaks. For this purpose, it must be supplied with descriptions of the table header, the individual table sections and the table end (see the *..TE and ..TS commands*).
The description of the table header starts with the ..THB command and ends with the ..THE command. The table header is printed for the first time at the position where it is defined and repeated after each page break until table end.

Note

- It is advisable to generate the table header in AS mode (see the *..AS command*).
- The *..NP command* is not permissible between *..THE* and *..TE*.

..TL[=i] Title Level (i = 0,1,2....9)

This command results in a page break in the text before a numbered title of level "i" or a lower level, and also controls line feed in the table of contents. Entered without an operand it serves to reset the TL function. If the ..TL command is omitted, a blank line is inserted before every title in the table of contents. If ..TL = i is issued, blank lines are inserted only before titles of level "i" or a lower level.

Note

The last "i" value is applied when creating the table of contents.

..TOC[=op] Table Of Contents

This command controls the output of a table of contents. It implicitly causes an automatic page break at the position in the text where it is entered. The value of the page count at the time the ..TOC command was entered is taken as the start value for the table of contents. The table of contents is compiled at the end of the TOM-DOC editing process. The optional operand is output as the table of contents header. Any commands which are to apply to the table of contents (e.g. header/footer definitions) must be entered at the end of the text input (except for ..QPN and ..NPT). The table of contents contains a reference to the keyword index, if one exists.

Note

The ..*TCL command* produces consistent pagination including the table of contents.

..TR=del>del Translate Input Characters

This command permits any input character to be converted into any output character. The command has the following format: C1>C2 , A>B, 9># etc. Alphanumeric characters must not exceed length 1; characters with length 2 are interpreted as hexadecimal notation. Entry of the ..TR command without operands resets the "translate" mode.

Conversion is executed with ..TR. If a converted character has a special meaning as a flag, it is converted as a flag. If this conversion is not possible, the character is removed from the input stream.

..TRL Translate Lower Case Letters

This command converts all lowercase letters (including umlauts) to uppercase in the subsequent text. ..TR (without operand) resets the "translate" mode.

..TS[=op] Table Section

A table section starts after the description of the table header (see the *..THB command*) or after the end of the previous table section. The ..TS command indicates the end of a table section and the start of the next one. If possible, a complete table section is always printed on one page; if this is not possible, a table break is carried out. If "op" is specified, it is printed as a separation line after a forced table break. Any operand specified remains valid until table end (see the *..TE command*); it should therefore be entered before the first table section.

Note

It is recommended to generate table sections in AS mode (see the *..AS command*).

..UL Underline On
..ULX Underline Off
..ULF=del Underline Flag
..ULT Underline Title Header

These commands support underlining. Text segments which are to be emphasized by underlining can be written between the ..UL and ..ULX commands. Alternatively the underline flag (..ULF=del) can be entered in the text immediately before and after the text segment to be emphasized. The start/end delimiters must always be entered in pairs. The underline flag only affects the line in which it is entered. In the edited text, it is replaced by a blank. It is reset by entering the ..ULF command without an operand. The ..ULT command causes all titles to be underlined.

Note

- Text segments marked with underline flags in diagrams are only interpreted if the diagram is a simple graphic representation which can be converted by the TOM-DOC graphic module without overlay lines. Otherwise graphic conversion has priority over underlining.
- Underlining is not supported for footnotes and tables.
- In the event of a clash, underlining has priority over bold type.
- The TRANSDATA 90xx Printers currently represent underlining by overprinting the line with a second line filled with the underscore character X’6D’.

5 Printing with TOM-DOC

5.1 General

A large number of printer types are available under both the BS2000 and SINIX operating systems for printing TOM-DOC files.

Note

If you use the product DPRINT(BS2000) as well, even more options are available to you for printing TOM-DOC files.

The BS2000 product DPRINT(Distributed Print Services) allows files to be printed in a network of BS2000 and SINIX/UNIX systems linked to each other via the TCP/IP, ISO or NEA(TRANSDATA) transport protocol.

You can use Distributed Print Services as of the following software versions:

- BS2000/OSD-BC V1.0 with SPOOL V3
- DCAM V11.0, PDN V8.0C
- FT-BS2000 V5.0B and FTAC V2.0A

Printing is generally initiated under BS2000 (/PRINT command). The software product RSO (REMOTE SPOOL OUTPUT) under BS2000 then passes the job on to the printers connected under BS2000 or SINIX.

a) Connecting TRANSDATA printers to BS2000 processors: A TRANSDATA printer can be connected either directly via a printer terminal controller or to a 9750 or 9752 Data Display Terminal (bypass operation).

b) Connecting TRANSDATA printers to SINIX processors: If a TRANSDATA printer is connected to a SINIX processor which has a communication link to a BS2000 processor, it is merely necessary to install the software product EMDS (9750 emulation + printer emulation) for execution. A print file can then be passed directly to the SINIX printer via RSO (BS2000) using printer emulation.

5.2 Printers and Resources for TOM-DOC

TOM-DOC(BS2000) currently supports the following printers for the BS2000 and SINIX operating systems:

```
9001 /     TRANSDATA 9001-31 Printer *)   Ink-jet printer
9001S                                     Ink-jet printer
9001-2 /   TRANSDATA 9001-2 Printer *)    Ink-jet printer
9001-2S                                   Ink-jet printer
9003       TRANSDATA 9003 Printer         Dot-matrix printer              **)
9004       TRANSDATA 9004 Printer         Daisy-wheel printer
9012       TRANSDATA 9012 Printer         Ink-jet printer
9013       TRANSDATA 9013 Printer         Dot-matrix printer
9022       TRANSDATA 9022 Printer         Laser printer (single-
                                          sheet handling)
ND         3350/3352 ND Laser Printer     High-speed laser printer **)
HP         2090-2/2140-2/3351/3353        High-speed laser printer **)
           HP Laser Printer               High-speed laser printer **)
LP65       3365 Laser Printer             Laser printer             **)
```

*) If the 9001 Printer is run at a 9750 Data Display Terminal or via the printer emulation on SINIX systems, 9001S or 9001-2S must be specified as the printer type in the ..DEV command.

**) The 2xxx, 33xx and 9003 types are only available under BS2000.

You can only output TOM-DOC files on ND laser printers of types 3350 and 3352 in BS2000 V10. These printers are supported in this version for the last time. As of BS2000/OSD V1, these printer types are replaced by HP laser printers of types 2090-2, 2140-2, 2240-2, 3351 and 3353.

Thus, if you want to output the edited TOM-DOC files on a system running BS2000/OSD V1 or a subsequent version, you have to insert the TOM-DOC command ..HP or ..DEV=HP at the beginning of your TOM-DOC input text so that the TOM-DOC output list is generated for the HP printer family.

If you use an HP printer with BS2000 V10, the same applies, of course. An ..HP or ..DEV=HP command is mandatory if you want to generate a list for an HP printer.

All output files edited by TOM-DOC can be printed on BS2000 systems by means of the BS2000 PRINT command or forwarded to SINIX systems via REMOTE-SPOOL jobs (the product RSO) and printed there.

When using SPOOL V3.0A, which can be used as of BS2000/OSD V1, the BS2000 product DPRINT can be used for printing on remote printers in a network.

The following table indicates how you can use the print resources in the BS2000 operating system and with the SPOOL version you are using:

Resources		BS2000 Operating system V10	OSD V1	OSD V2
Command	PRINT	X	X	X
	PRINT-FILE		X	X
	PRINT-DOCUMENT			X
	DPRINT product		X (*)	X (*)
Spool	2.5B	X		
	2.6A	X		
	2.7A / 2.7B		X	
	3.0A		X (*)	X (*)
Printer	ND	X (+)		
	HP	X	X	X
	LP65	X	X	X
Character sets	POST.NDFILE	X		
	POST.HPFILE	X	X	
	POST.PRFILE			X (!)
	POST.RSOFILE	X	X	X

(+) The ND printer is supported for the last time in BS2000 V10 together with Spool V2.7A.

(*) The character set files HPFILE, NDFILE and SPOOLFILE (for BAND and line printers) must be converted to the PRFILE format by the PRM program for use with SPOOL V3.0.

(!) The conversion of the character set files HPFILE, NDFILE and SPOOLFILE by the PRM program for use with SPOOL V3.0 takes place in the POST.PRFILE file.

Example

To convert POST.HPFILE, please use the following commands:

```
/MOD-SDF-OPT PROC-DIA=*YES
/MOD-JOB-SW ON=1
/START-PRM
//OPEN-PR-FILE MYIMAGELIB.HPFILE,,*UPDATE
//CLOSE-PR-FILE
//END
/
```

5.3 The PRINT Commands

The table on page 157 indicates the command to use when you want to output to one of the supported printers. Since the SPOOL version used plays a decisive role in the use of a PRINT command, you should always consult this table in the following examples.

5.3.1 Printing out to ND Printers

TOM-DOC files can be output to 3350 and 3352 ND Laser Printers by means of the PRINT command:

```
/PRINT filename,SPACE=E
```

a) The print parameters *CONTROL* und *CHARS* If files containing bold print (..CSP and ...EM) are to be printed, you must switch over to particular character sets used by TOM-DOC. The command must read as follows:

```
          /PRINT filename,CON[TROL]=PHYS,CHARS=(DL,DM)
          (for a small character set with 8 lpi and 15 cpi)
or
          /PRINT filename,CON[TROL]=PHYS,CHARS=(DI,DJ)
          (for a medium-sized character set with 8 lpi and 12 cpi)
or
          /PRINT filename,CON[TROL]=PHYS,CHARS=(DE,DF)
          (for a large character set with 6 lpi and 10 cpi)
```

b) The print parameter *IMAGE* The character sets DI and DJ or DE and DF are loaded from the SPOOL printer control file $TSOS.NDFILE to the buffer of the laser printer. You also have the option of defining or copying your own character sets and storing them in a special character set file. This file name must conform to the format *$userid.xxxx.NDFILE*, where *xxxx* comprises 1-4 characters and is specified as an operand of the IMAGE parameter:

```
/PRINT filename,CON[TROL]=PHYS,CHARS=(DI,DJ),IMAGE=POST
```

The character sets DI and DJ are then loaded from the POST.NDFILE file under the user's own ID. The service program *PRSERVE* (see SPOOL description, part 2) can be
used to generate user-specific character sets or to modify or copy existing character sets.

c) The print parameters *FORM* and *LOOP* Printouts are normally printed by the laser printer on standard paper (such as 9-inch*315 mm recycled paper: FORM=STD). If the printout is to be on another type of paper (such as white paper, DIN A4 format), the FORM parameter must be specified as well:

```
/PRINT filename,CON[TROL]=PHYS,CHARS=(DE,DF),FORM=form-name
```

The form name for white paper must be entered in the SPOOL parameter file. Enter the SHOW-SPOOL-FORMS (S-S-F) command to display the form name. Selecting a specific form simultaneously establishes a preset value for line spacing (LOOP parameter). It may be necessary to modify the preset line spacing using the LOOP parameter. If you want to print on white paper (12-inch) with a small character set and narrow line spacing, select the following command:

```
/PRINT filename,CON[TROL]=PHYS,CHARS=(DI,DJ),FORM=LDOC,LOOP=C8
```

Possible operand values for the LOOP parameter are listed below:

LOOP=96 (9-inch paper + 6 lines/inch)
LOOP=98 (9-inch paper + 8 lines/inch)
LOOP=C6 (12-inch paper + 6 lines/inch)
LOOP=C8 (12-inch paper + 8 lines/inch)

5.3.2 Printing out to HP Printers

TOM-DOC files can be output to HP laser printers by means of the PRINT, PRINT-FILE and PRINT-DOCUMENT commands.

For semigraphical representation on HP laser printers there is a fifth character set. Since the CHARS parameter of the PRINT commands permits a maximum of 4 character sets to be specified, these 5 character sets are grouped together in a character set pool. This must be specified in the CHARS-POOL parameter instead of the CHARS parameter in the PRINT commands.

All the character sets are in the POST.HPFILE or POST.PRFILE file together with the corresponding character set pools. The following applies:

```
TOME character set pool -> DE1,DE2,DE3,DE4,DE5 character sets
                           (character spacing: 10 characters per inch)

TOMI character set pool -> DI1,DI2,DI3,DI4,DI5 character sets
                           (character spacing: 12 characters per inch)

TOML character set pool -> DL1,DL2,DL3,DL4,DL5 character sets
                           (character spacing: 15 characters per inch)
```

5.3.2.1 Printing with the PRINT Command

TOM-DOC files can be printed on HP laser printers by means of the PRINT command:

```
/PRINT filename,CHARS-POOL=TOME,DEVICE=(HP),CONTROL=PHYS
```

– The CONTROL, CHARS and DEVICE print parameters are required so that TOM-DOC files can be output correctly on HP laser printers with underlining, bold type or dummy graphic representations. For a selected font size, you must be able to switch between several character sets: DL1, DL2, DL3 and DL4 (small character sets); DI1, DI2, DI3 and DI4 (medium-sized character sets); and DE1, DE2, DE3 and DE4 (large character sets).

  So that the switchover control characters generated by TOM-DOC are interpreted correctly, the parameter DEVICE=(HP) must also be specified. The following commands must be used:

```
/PRINT filename,CONTROL=PHYS,CHARS=(DL1,DL2,DL3,DL4) -
                ,DEVICE=CENTRAL(HP)
(for the small character set with 8 lpi and 15 cpi)

/PRINT filename,CONTROL=PHYS,CHARS=(DI1,DI2,DI3,DI4) -
                ,DEVICE=CENTRAL(HP)
(for the medium-sized character set with 8 lpi and 12 cpi)

/PRINT filename,CONTROL=PHYS,CHARS=(DE1,DE2,DE3,DE4) -
                ,DEVICE=CENTRAL(HP)
(for the large character set with 6 lpi and 10 cpi)
```

When working with the pools:

```
/PRINT filename,CHARS-POOL=TOML,DEVICE=(HP),CONTROL=PHYS
(for the small character set with 8 lpi and 15 cpi)

/PRINT filename,CHARS-POOL=TOMI,DEVICE=(HP),CONTROL=PHYS
(for the medium-sized character set with 8 lpi and 12 cpi)

/PRINT filename,CHARS-POOL=TOME,DEVICE=(HP),CONTROL=PHYS
(for the large character set with 6 lpi and 10 cpi)
```

– The IMAGE and CHARS or CHARS-POOL print parameters

  By default, the character sets DL1,DL2,DL3,DL4, DI1,DI2,DI3,DI4 and DE1,DE2,DE3,DE4 are loaded from the SPOOL printer control file $TSOS.HPFILE or $TSOS.PRFILE in the memory of the laser printer. They must be adopted at installation of TOM-DOC. If this is not usual in your computer center, you can store them and also your own character sets in a private character set file. In these cases, the name of this file must be $userid.xxx.HPFILE or $userid.xxx.PRFILE. xxxx consists of 1 to 4 characters and must be specified as an operand of the IMAGE parameter:

  ```
  /PRINT filename,CHARS-POOL=TOME,DEVICE=(HP),CONTROL=PHYS,IMAGE=POST
  ```

  In this example, the character sets of the TOME character set pool are loaded from $userid.xxx.HPFILE or $userid.xxx.PRFILE.

  You can create your own character sets or modify or copy existing character sets with the SPOOL programs PRSERVE (see the description of SPOOL Version 2) and PRM (see the description of SPOOL Version 3).

  The character sets and character set pools are supplied with TOM-DOC in POST.HPFILE and POST.PRFILE.

– The print parameters FORM, LOOP and ROTATION

  Printouts on the HP laser printers are generally on standard paper (e.g. 9 inch * 315 mm recycled paper: FORM=STD). If you want to use different paper for printing (e.g. white 9-inch paper), you have to specify the FORM parameter as well:

  ```
  /PRINT filename,CONTROL=PHYS,CHARS-POOL=TOMI,DEVICE=(HP),FORM=form-name
  ```

  The form-name for white paper must be entered in the SPOOL parameter file. It can be queried with the SHOW-SPOOL-FORMS command. When a specific form is selected, a default character density is set (LOOP parameter). Sometimes it is necessary to change the default character density. You do this by means of the LOOP parameter. If you want to print on white paper (12 inch) with a small character set and narrow line spacing, you use the following command:

  ```
  /PRINT filename,CONTROL=PHYS,CHARS-POOL=TOMI,DEVICE=(HP) -
                  ,FORM=form-name,LOOP=C08
  ```

The operand values of the LOOP parameter can be as follows:

```
LOOP=906 (9-inch paper + 6 lines per inch)
LOOP=908 (9-inch paper + 8 lines per inch)
LOOP=C06 (12-inch paper + 6 lines per inch)
LOOP=C08 (12-inch paper + 8 lines per inch)
```

You will often want to print the TOM-DOC output file with page rotation. To do this, you require the ROTATION parameter, which allows page rotation of 0, 90, 180 and 270 degrees (see also the TOM-DOC commands ..ROT and ..CSP).

To implement this, the POST.HPFILE and POST.PRFILE character set files also contain loops, which are required when the TOM-DOC output involves page rotation. You can specify these in your PRINT command together with the ROTATION parameter.

```
/PRINT filename,CONTROL=PHYS,CHARS-POOL=TOME,DEVICE=(HP) -
                ,FORM=form-
                name,LOOP=(908,B06),ROT=90,IMAGE=POST
```

Skillful combination of the FORM, LOOP and ROTATION parameters in a PRINT command makes it possible to have very varied printed output. However, you should also note that not every combination is worthwhile.

You can use the following LOOP parameters for page rotation:

```
LOOP=B06 (9-inch paper, landscape format, 6 lines per inch)
LOOP=B08 (9-inch paper, landscape format, 8 lines per inch)
```

Here are some examples of how you can print your documents:

POOL	LOOP	ROTATION-LOOP	ROTATION	FORM
TOME	C06	No	No	12 inch
TOME	908	B06	90	9 inch
TOMI	908	B08	90	9 inch
TOMI	908	No	No	9 inch
TOML	908	B08	90	9 inch
TOML	908	No	No	9 inch

5.3.2.2 Printing with the SDF Commands PRINT-FILE and PRINT-DOCUMENT

Note

To understand this chapter better, you should read section 5.3.2.1, "Printing with the PRINT Command", first.

The SDF commands PRINT-FILE and PRINT-DOCUMENT differ from the PRINT command described above in both their semantics and syntax on account of their SDF notation.

Nevertheless, the technical description in "Printing with the PRINT Command" also applies to the PRINT-FILE and PRINT-DOCUMENT commands. An example is therefore given for each command below.

The interpretation of the parameters of the PRINT command described above (IMAGE, FORM, LOOP, ROTATION, CONTROL, DEVICE, etc.) in the SDF notation of the PRINT-FILE and PRINT-DOCUMENT commands should be self-explanatory.

Example

The IMAGE=POST parameter of the PRINT command must be written in the PRINT-FILE command as "USER-PARAMETER-FILE=POST" and in the PRINT-DOCUMENT command as "USER-RESOURCE-FILE=POST".

You can read about the PRINT-FILE and PRINT-DOCUMENT commands and their additional features in the manual entitled "User Commands (SDF Format)".

5.3.2.3 Printing with the PRINT-FILE Command

You can output TOM-DOC files on HP laser printers with the PRINT-FILE command:

```
PRINT-FILE FILE-NAME=filename,SPOOLOUT-NAME=text -
           ,LAYOUT-CONTROL=*PARAMETERS(LOOP-NAME=908 -
           ,ROTATION-LOOP-NAME=B08,CONTROL-CHARACTERS=*PHYSICAL -
           ,CHARACTER-SETS=*POOL(POOL-NAME=TOMI) -
           ,USER-PARAMETER-FILE=POST,ROTATION=90)
```

5.3.2.4 Printing with the PRINT-DOCUMENT Command

You can output TOM-DOC files on HP laser printers with the PRINT-DOCUMENT command:

```
PRINT-DOCUMENT FROM-FILE=filename -
               ,DOCUMENT-FORMAT=*PAGE-FORMAT(CONTR-MODE=*PHYS) -
               ,RESOURCE-DESCRIPTION=*PAR(CHAR-SETS=(DE1,DE2,DE3,DE4) -
               ,USER-RESOURCE-FILE=POST) -
               ,TO-PRINTER=*PAR(PRINTER-TYPE=*HP-PRINTER)
```

5.3.3 Printing on LP65 Printers

You can output TOM-DOC files on LP65 laser printers with the PRINT or PRINT-FILE command:

```
PRINT filename,SPACE=E
```

Unlike the laser printers supported up to now, it is not possible with the 3365 to load all the print resources to the printer's memory from the host. Instead, they must be loaded into the printer via floppy disk.

A disk is therefore shipped with TOM-DOC V3.2A. This disk contains various print resources required by TOM-DOC to support the 3365 laser printer:

```
SET00246.PCL Print control file (DIN A4 paper, loadable character sets
             SET00991.C00 and SET30993.C00)

SET00247.PCL Print control file (DIN A3 paper, loadable character sets
             SET00991.C00 and SET30993.C00)

SET00248.PCL Print control file (DIN A4 paper, loadable character sets
             SET00992.C00 and SET30994.C00)

SET00249.PCL Print control file (DIN A3 paper, loadable character sets
             SET00992.C00 and SET30994.C00)

SET00991.C00  Character set file (DE character set)

SET00992.C00  Character set file (DI character set)

SET30993.C00  Character set file (DF character set)

SET30994.C00  Character set file (DJ character set)

SET00000.CAT  Conversion table
```

Loadable character sets are not currently available for DL and DM. The existing character sets for the LP65 printer are shipped as files on the FL5396 floppy disk:

```
File name             Description

SET00246.PCL          Print control file
SET00247.PCL          Print control file
SET00248.PCL          Print control file
SET00249.PCL          Print control file
SET00991.C00          Character set file
SET00992.C00          Character set file
SET30993.C00          Character set file
SET30994.C00          Character set file
SET00000.CAT          Conversion table
```

– The RESOURCE and PAGEDEF print parameters

  In order to be able to output on LP65 printers TOM-DOC files edited with underlining, bold type or dummy graphic representations, it must be possible to switch between several character sets (SET00991, SET00993 or SET00992, SET00994). So that the switchover control characters generated by TOM-DOC are interpreted correctly, the RESOURCE parameter must also be specified. The following commands must be used:

```
/PRINT filename,SPACE=E,RESOURCE=LP65(PAGEDEF=248)
(for a small character set with 12 cpi)
```

  oder

```
/PRINT filename,SPACE=E,RESOURCE=LP65(PAGEDEF=246)
(for a large character set with 10 cpi)
```

– The print parameter DUPLEX=YES | NO | TUMBLE | STD

```
/PRINT filename,SPACE=E,RESOURCE=LP65(PAGEDEF=246 -
                ,DUPLEX=YES)
```

  This command prints a TOM-DOC file with a large character set (10 cpi) on both sides of the paper (DIN A4). Page rotation is implemented here with the DUPLEX parameter rather than with the TOM-DOC command ..ROT.

  You use TUMBLE to print on both sides of the paper, which is now rotated from bottom to top (along the X-axis) rather than from left to right (along the Y-axis).

```
/PRINT filename,SPACE=E,RESOURCE=LP65(PAGEDEF=246 -
                ,DUPLEX=TUMBLE)
```

  This command prints a TOM-DOC file with a large character set (10 cpi) on both sides of the paper (DIN A4) and rotates the page along the X-axis.

  DUPLEX=NO means SIMPLEX mode, in which printing is on only one side of the paper. DUPLEX=STD means that the file is printed as defined in the PCL file (DUPLEX or SIMPLEX).

– The FORM and LOOP print parameters

  If you specify the name of a form, a LOOP record is assigned implicitly. This LOOP record must be contained in the $TSOS.SPOOLFILE for the output device (LP65 printer of type 3365).

  Also refer to the ’BS2000 user Commands’ manual.

5.3.4 Printing out to TRANSDATA 90xx Printers (RSO Printers)

BS2000 files can be output to various TRANSDATA printers (of the type 90xx). This is done using the software product RSO (REMOTE SPOOL OUT), a BS2000 subsystem which supports various printers by means of communication processors and front-end processors. TOM-DOC currently supports the following printers: 9001-2, 9001-31, 9004, 9012, 9013 and 9022. The following points must be heeded when outputting TOM-DOC files to RSO printers:

- *Defining* RSO printers - *Starting* RSO printers - *Operating* RSO printers

a) Defining RSO printers The system administrator must first define an RSO printer using the ADD-SPOOL-DEVICE (A-S-D) command, thereby entering this printer in the SPOOL parameter file and establishing a particular user ID as device administrator. Device administrators and non-priviledged users can use the SHOW-SPOOL-DEVICES (S-S-D) command to obtain information on RSO devices.

b) Starting RSO printers Once an RSO printer has been defined, the system administrator or device administrator uses the following command to activate the printer:

```
/SDVC DEV=device-name,FORM=form-name
```

Note

- The device name designates the printer to be activated. The SHOW-SPOOL-DEVICES (S-S-D) command is used to obtain a list of defined printers.
- The form name designates the paper format/form to be used on the selected printer. The SHOW-SPOOL-FORMS (S-S-F) command is used to obtain a list of available forms.

Once the command has been successfully executed, the printer is in status "A". The device administrator can use the STA[TUS] R[EMOTE] command to obtain information on the status of an RSO device.

c) Operating RSO printers The following command is used to output a TOM-DOC file to the appropriate RSO printer:

```
/PRINT filename,SPACE=E,DEV[ICE]=device-name,FORM=form-name,
                CON[TROL]=LOG[ICAL]
```

Note

- The file name designates the TOM-DOC file to be printed.
- The device name designates the RSO printer selected.
- The form name designates the paper format/form used on the RSO printer.
- The CONTROL=LOGICAL parameter is necessary only if the ..TD command was used in the TOM-DOC source file. As of RSO Version 2.0A, the CONTROL=PHYSICAL parameter can also be used instead of the CONTROL=LOGICAL parameter:

  ```
  /PRINT filename,SPACE=E,DEV[ICE]=device-name,FORM=form-name,
                  CON[TROL]=PHYS[ICAL]
  ```

  Unlike for output with CONTROL=LOGICAL, no LOOP record is evaluated with CONTROL=PHYSICAL.
- If umlauts were generated in a TOM-DOC source using ..SF or the CODE statement of the editor, a code conversion table must be specified in the PRINT command as of RSO version 2.0A:

  ```
  /PRINT filename,SPACE=E,DEV[ICE]=device-name,FORM=form-name,
                  TRANSLATION-TABLE=(DOC,POST)
  ```

  This command transforms umlauts via the DOC conversion table located in the POST.RSOFILE file. The DOC conversion table is supplied with TOM-DOC in the POST.RSOFILE file.

You can use the following commands to check on the status of your print job:

```
/STA[TUS] L[IST],TYPE=7
```

or

```
/STA[TUS] E[NVIR],TYPE=7
```

6 Appendix

6.1 List of Commands in Functional Groups

The following tables contain the TOM-DOC commands sorted according to function. This will help you quickly find the commands you need to complete a task.

6.1.1 Input/Output

Command	Function	Effect
..OPF=op	File name of output file	first command applies
..SRC=[Y / N]	Specify line numbers of the source file	until reset
..SW=op, plamtyp, version	Name of a file, PLAM member, Version of a member of the FMS or LMS library	once
..SWL=op	Name of a library	once

6.1.2 Printer Interface

Command	Function	Effect
..HP	Control character generation for 3351 or 3353 Laser Printer	first command applies
..TDi	Control character generation for TRANSDATA printer	until reset
..DEV=op	Select output device	first command applies
..ROT=i	Page rotation	every printed page

6.1.3 Graphics Conversion

Command	Function
..AS ..AX	Switch on 1-to-1 mode Switch off 1-to-1 mode
..CH=del>del	Redefine dummy graphics character
..CHF=[Y / N]	Graphics conversion for headers and footers
..COD ..COX	Switch on graphics conversion Switch off graphics conversion

6.1.4 Table of Contents and Index

Command	Function	Valid in AS mode
..HIX=op	Header of keyword index	yes
..IF=del	Mark keywords	yes
..IN[=i] ..) ..IX	Open index level Increment current index by 1 Close index level	no no no
..ISF=opo,opo	Sort keyword index	yes
..Li_op	Title numbering	no
..LL=[Y / N]	Generate leading lines	yes
..NPT	New page in table of contents	yes
..RIX=op	Qualify references in the keyword index	yes
..SI=op	Define keywords	yes
..SLT=i	Suppress titles ≤ level i	no
..TH_op	Unnumbered title	no
..TL=i	Conditional page break before titles	no
..TOC[=op]	Generate table of contents	yes

6.1.5 Titles and Footnotes

Command	Function	Valid in AS mode
..EMT	Print titles in bold face	yes
..EN	Expose numbers of titles	yes
..FN ..FNX	Start footnote End footnote	yes yes
..HIX[=op]	Header of keyword index	yes
..Li_op	Numbering titles	no
..TCL=i	Continuous page numbering	yes
..ULT	Underline titles	yes

6.1.6 Tables

Command	Function
..AS ..AX	Switch on 1-to-1 mode Switch off 1-to-1 mode
..TE	End table processing
..THB ..THE	Table header begin Table header end
..TS[=op]	Table section

6.1.7 Mark Text

Command	Function
..* op	Comment
..CA=op	Modification marking
..CM=[i_[op]]	Center mark for mid-page

6.1.8 Page Layout

Command	Function	Valid in AS mode
..CPL=i	Define characters per line	yes
..CSP={ 10 | 12 | 15 }	Set character spacing	yes
..FC=del	Frame character for left and right margins	yes
..FM=i ..FME=i	Width of filing margin Width of filing margin for even pages	yes yes
..FT=i_op ..FTE=i_op	Define footer i Define footer i for even pages	yes yes
..HD=i_op ..HDE=i_op	Define header i Define header i for even pages	yes yes
..HM=i_op ..HM	Indent left-hand margin i characters Reset left-hand margin	no no
..JU ..JX	Switch on right-hand margin justification Switch off right-hand margin justification	no no
..LPB=i ..LPF=i ..LPH=i	Lines per body Lines per footer Lines per header	yes yes yes
..ML=i ..MR=i	Set left-hand margin Set right-hand margin	no no

6.1.9 Text Layout

Command	Function	Valid in AS mode
..AS	Switch off line break function (1-to-1 mode)	yes
..AX	Switch on line break function	yes
..CE	Center text	no
..EM	Switch on bold print function	yes
..EMF=del	Define bold print flag	yes
..EMX	Switch off bold print	yes
..EMT	Titles in bold print	yes
..LSP=i	Line feed in the current text	yes
..NL=i	Blank lines	yes
..NP=i	i blank pages	yes
..NPE	Change to next even page	yes
..NPO	Change to next odd page	yes
..NPT	Page feed in table of contents	yes
..P	Paragraph indented	yes
..PIN=i	Define width of indent	yes
..RL=i	Conditional page break	no
..TL=i	Conditional page break before titles	no
..TRL	Convert to uppercase letters	yes
..UL	Underline on	yes
..ULF..del	Underline flag	yes
..ULX	Underline off	yes
..ULT	Underline titles	yes

6.1.10 Hyphenation

Command	Function	Valid in AS mode
..HY=del	Semi-automatic hyphenation	no
..PSY=i[_{A|B}]	Define minimum length of prefix or suffix (range of values 1 to 99)	no
..SYL={Y|N}	Automatic hyphenation	no

6.1.11 Redefine

Command	Function	Valid in AS mode
..CC=del	Change command delimiter ..	no
..CH=op	Change dummy graphic character	yes
..MB=del	Change metablank character	yes
..SF=del	Mark umlaut	yes
..SYM=del	Change delimiter for variable (&)	yes
..TR=del>del	Convert input character	yes

6.1.12 Dummy Graphic Characters

Character	Function
-	normal horizontal line
=	bold horizontal line
%	dashed horizontal or vertical line
!	normal vertical line
I	bold vertical line
’	slanting line, right
‘	slanting line, left
+	connector for all directions
*	symbol to join centric lines
#	symbol for horizontal/vertical connections
(	round corner or left upward curve
)	round corner or right upward curve
?	downward curve
v	arrowhead downwards
^	arrowhead upwards
<	arrowhead to the left
>	arrowhead to the right

6.1.13 TOM-TI Commands for Text Editing

Command	Function	Transfer key
*DOC[S] ln1-ln2	Summary interactive text editing	DÜ1
*DOCE	Editing of a complete output text in batch mode without auxiliary functions	DÜ1
*DOCF	Interactive editing of a complete output text	DÜ1
*DOCL	Editing of text (in batch mode) and use of auxiliary functions for layout via input in screen mask (interactive mode), automatic buffering in work area 3 or transfer of generated commands to work area 0	DÜ1
*DOCL NO[SCREEN]	Same function as *DOCE command	DÜ1

6.1.14 Significance of TOM-TI Work Areas

Area	Contents
0	Text provided with edit control statements
1	Interactively edited text (commands *DOC and *DOCF)
2	Error messages (e.g. errors in edit control statements)
3	Layout information for print editing (*DOCL command)

6.1.15 Symbolic Parameters

Para-meter	Format	Length	Meaning (entry in page header or footer)
&S1	nnnn	1-4	Page counter, right-justified
&S2	nnnn	1-4	Page-counter+1, right-justified
&S3	nnnnx	2-5	Page counter, right-justified with end identifier +/-
&P1	nnnn	1-4	Page counter, left-justified
&P2	nnnn	1-4	Page counter+1, left-justified
&DATUM	dd.mm.yy	8	Current date (European format)
&DATE	mm/dd/yy	8	Current date (American format)
&TIME	hh:mm:ss	8	Time of day

Variable	Meaning
&Tn	Count variable n=1...6, up to 3 digits, left-justified
&LNi	Value of individual level i of last title, up to 3 digits, left-justified
&LCi	Full decimal classification of last title on level i (where i = 0 to 9), up to 40 digits
&LTi	Text of last title on level i (where i = 0 to 9)
&PA=i	Increment page counter
&RFx(i)	Reference variable (set to a specific value of a system variable by means of the ..SET command) x: alphanumeric identification (1 to 5 uppercase characters) i: number of positions for variable (up to 99, default value = 3)

6.2 TOM-DOC MESSAGES

This section contains an alphabetic list of all TOM-DOC messages, indicating possible causes of error and recovery actions as well. The messages/warnings are always preceded by a reference to the relevant source line number, the relevant command and, at the start of a line, an abbreviation indicating the type of error, where:

```
W    warning
E    syntax error
S    semantic error
I    illegal command
```

```
..CPL + ..FM IS GREATER 204
```

Meaning
The sum of the values defined with ..CPL, ..FM und ..FME exceeds the permitted value (204).

Action
Alter current values.

```
APPROPRIATE DEVICE TYPE DEFINITION IS MISSING
```

Meaning
The ..TDi command was used to select a type style without the appropriate device type having been specified.

Action
Use the ..DEV command to specify an appropriate device type before entering any text or an initial ..TDi command.

```
ERROR ON FILE / ELEMENT
```

Meaning
A file or an FMS/PLAM member is to be accessed by means of ..SW but the file management system reports an error (other than "not found").

Action
Check file/library.

```
ERROR ON COPY-LIBRARY
```

Meaning
The library specified in the ..SWL command does not exist or cannot be accessed.

Action
Check and correct file names or catalog entries (shareable, password etc.)

EXTRA BOLD PRINT NOT ALLOWED IN THIS PLACE

Meaning
The ..CSP command occurs more than once or after the first text input in the source.

Action
Make sure that the ..CSP command occurs before the first text input and no more than once per source to be edited.

HANG MARGIN EXCEEDS RIGHT MARGIN

Meaning
The value specified in the ..HM command exceeds the right-hand margin defined with ..MR.

Action
Either redefine the right-hand margin or specify a smaller value for the operand of the ..HM command.

ILLEGAL COMMAND

Meaning
The input source contains an invalid TOM-DOC command.

Action
Check command syntax or orthography.

ILLEGAL COMMAND COMBINATION

Meaning
The TOM-DOC source contains an invalid combination of commands (e.g. bold type without ..CSP command).

Action
Correct command(s).

ILLEGAL OPERAND

Meaning
A command contains an illegal operand.

Action
Correct operand (see command syntax).

INDEX LEVEL EXCEEDED

Meaning
An attempt has been made to open more than the permitted number of 10 index levels.

Action
Redefine index levels.

INDEX LEVEL NEGATIV

Meaning
An attempt has been made to close an index level which has not previously been opened by ..IN.

Action
Either open index level with ..IN as required or eliminate the ..IX command which caused the error.

INVALID DEVICE-TYPE

Meaning
The device type specified as an operand in the ..DEV command is not supported by TOM-DOC.

Action
Specify valid device type as operand.

INVALID VERSION

Meaning
FMS reports the specification of an invalid version number (separated from the member name by a comma) in the ..SW command.

Action
Correct version number specification.

KEYWORD INDEX NOT ALLOWED

Meaning
Footnotes must not contain any keywords.

Action
Erase ..SI command or keyword flag (..IF) in footnote.

LEFT MARGIN EXCEEDS RIGHT MARGIN

Meaning
The value specified in the ..ML command is greater than the value specified in the ..MR command.

Action
Enter either a larger value for the ..MR operand or a smaller value for the ..ML operand.

```
LEFT MARGIN 0 NOT ALLOWED
```

Meaning
The specification of 0 for the operand of the ..ML command is illegal. The permitted range of values is 1,2...(MR-2).

Action
Correct value specification of ..ML command.

```
LINE TRUNCATED
```

Meaning
In AS mode: some lines exceed the number of columns specified in the ..CPL command (default value: ..CPL=74).

Action
Either increase value for CPL or split up relevant lines in AS mode.

```
MAXIMUM SIZE OF LINES PER PAGE EXCEEDED
```

Meaning
The sum of the values specified in ..LPH, ..LPB and ..LPF must not exceed 160.

Action
Alter values.

```
MISSING START OF A FOOTNOTE
```

Meaning
An ..FNX command has been entered to close a footnote that does not have a defined beginning.

Action
Either define beginning of footnote by means of an ..FN command or, if no footnote is required, erase the ..FNX command.

```
NO ELEMENT / VERSION FOUND
```

Meaning
The member specified with ..SW cannot be found in the library specified with ..SWL.

Action
Check member name or library.

`NO OPERAND ALLOWED`

Meaning
An operand has been specified for a command for which no operands are allowed (possible reason: the semicolon which is required to separate the command from the current text is missing, with the result that the text is interpreted as a command operand).

Action
Correct command (enter semicolon or eliminate operand).

`NO VALUE FOUND`

Meaning
No value is assigned to a variable defined in the input.

Action
Set variable in the text by means of the ..SET command.

`NO VERSIONS IN ELEMENT`

Meaning
A version number has been specified in an ..SW command, but the specified member does not contain any versions.

Action
Erase version number.

`NOT ENOUGH SPACE FOR SOURCE REFERENCE`

Meaning
The command ..SRC=Y has been issued although the value specified for the filing margin in the ..FM command is smaller than 4 columns, i.e. no references to source line numbers (at least 4 columns) can be printed.

Action
Increase the value specification in the ..FM command or enter ..SRC=N or erase ..SRC=Y.

`ODD BUFFER LENGTH - JUSTIFY NOT POSSIBLE`

Meaning
Applies to wide type: the number of columns between the left-hand and right-hand margins is an odd number. The text can therefore not be aligned.

Action
Correct values specified by ..ML and ..MR.

`ODD BUFFER LENGTH OR INDENTATION AMOUNT`

Meaning
Applies to wide type: either the length of the buffer for titles or the indentation amount

specified with ..HM is an odd value, with the result that alignment is not possible.

Action
Correct values.

```
ONLY 10 - 160 LINES PER BODY ALLOWED
ONLY 0 - 16 LINES PER FOOTER ALLOWED
ONLY 0 - 16 LINES PER HEADER ALLOWED
ONLY 0 - 9 FOR PARAGRAPH INDENTATION AMOUNT ALLOWED
ONLY 1 - 16 FOR HEADLINE / FOOTLINE ALLOWED
ONLY 10 OR 12 CHARACTERS PER INCH ALLOWED
ONLY 0 - 20 ALLOWED FOR FILE MARGIN
```

Action
The valid range of values of the commands concerned is specified in the messages; correct the values entered accordingly.

```
OPEN ERROR ON TOM-DOC-DEVICETABLE
```

Meaning
The TOM-DOC-DEVICETABLE file could not be found (installation error)

Action
Inform the system administrator

```
OUTPUT-FILE IS LOCKED
```

Meaning
The specified output file cannot be opened.

Action
Check file (attributes) and assign different output file if necessary.

```
PRECEDING "..IN" MISSING
```

Meaning
A ..) command has been issued to increment an index level which has not previously been opened with ..IN.

Action
Open index level with ..IN as required.

```
RIGHT MARGIN EXCEEDS HIGHEST POSSIBLE VALUE
```

Meaning
The value of the operand of the ..MR command exceeds the range of values permitted: (ML+2)...(CPL-6).

Action
Correct value specification.

SORT PARAMS EXCEED 1-40

Meaning
A single sort field specification or the sum of the sort field specifications in the ..ISF command exceeds the permitted keyword length of 40.

Action
Correct values.

SYNTAX ERROR IN OPERAND

Meaning
A command operand does not comply with the permissible range of values.

Action
Enter correct operand.

TEXT LENGTH EXCEEDS INDENTATION AMOUNT

Meaning
The text operand of the ..HM command is greater than the previously specified indentation value.

Action
Correct the ..HM command; the following applies to "..HM=i op": "op" must not exceed the value specified for "i".

TOO MANY FOOTNOTE LINES
TOO MANY TABLE HEADER LINES

Meaning
The number of lines specified for a footnote or a table header exceeds the number of lines per text body (..LPB command).

Action
Either reduce footnote or table header lines, or increase value in ..LPB.

UNABLE TO OPEN FILE / OTHER FILE STILL OPENED

Meaning
An ..SW command was issued within a BS2000 file already being read in with ..SW (nested ..SW input of BS2000 files is not permitted).

Action
Store the relevant BS2000 file as a member in an FMS or PLAM library if possible. Otherwise the ..SW command must be replaced with those input lines it was to read.

UNSATISFIED REFERENCE

Meaning
The text contains an undefined reference variable. Since reference variables can be set until the end of text editing, the last line of the input file is always output as the source line reference in this case.

Action
Enter a correct value for the reference variable at the relevant position in the text (see the ..SET command).

VARIABLE ALREADY SET

Meaning
A reference variable has been set more than once in the text.

Action
Use different reference variables for different values.

VARIABLE BUFFER OVERFLOW

Meaning
The internal buffer for reference variables is full; additional reference variables are not buffered and cannot be set.

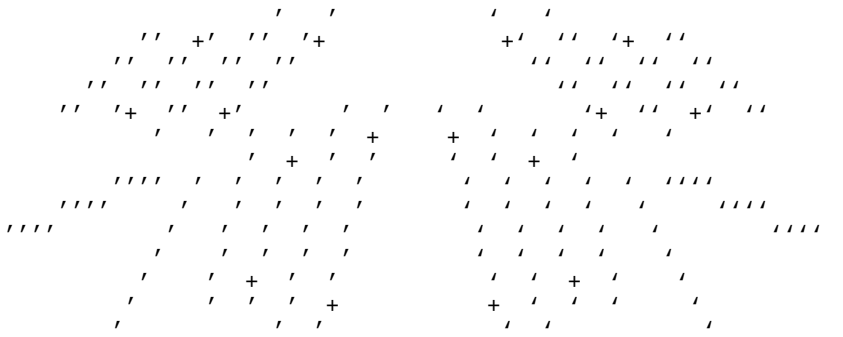
Action
Reduce number of reference variables.

6.3 Further Examples of TOM-DOC Graphic Module

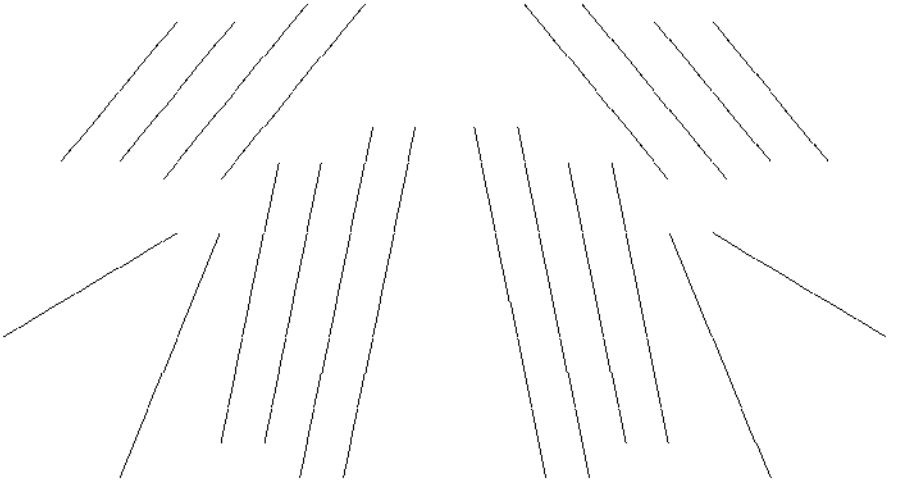
6.3.1 Sloping Lines

Illustrations of inputs/outputs show all the different oblique lines that the TOM-DOC graphics module recognizes from the characters " ’ " and " ‘ ".

Input



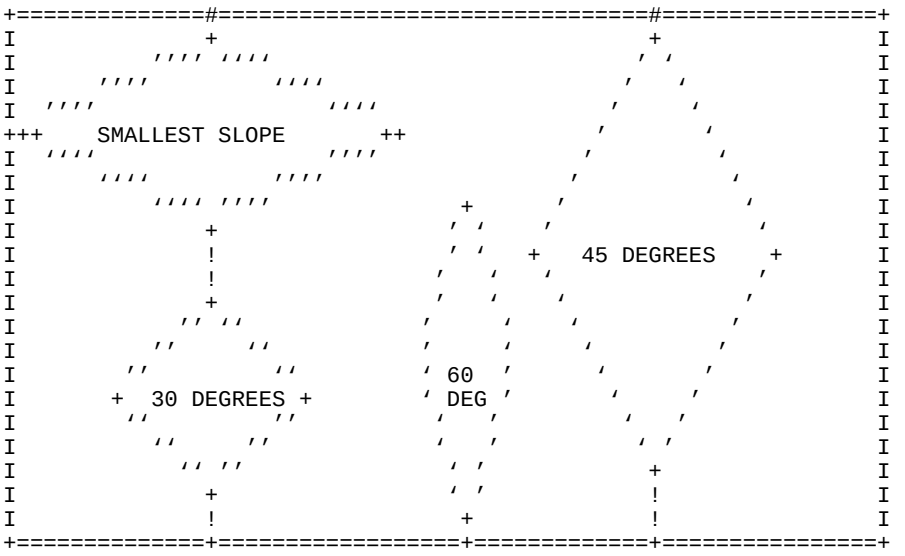
Output



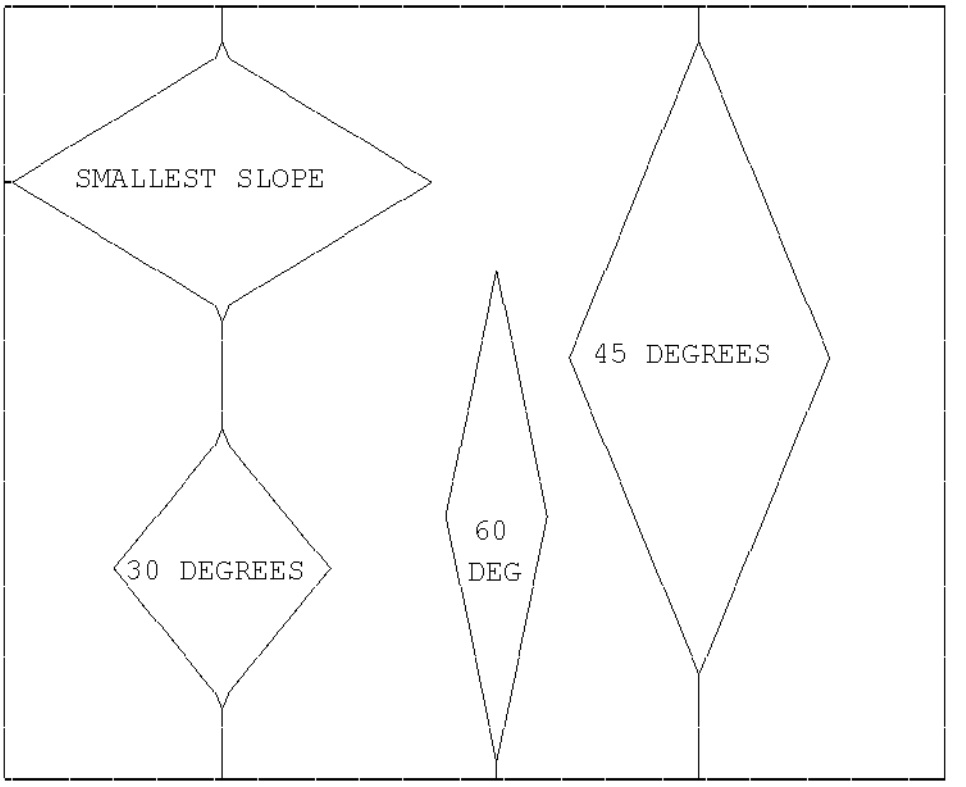
6.3.2 Junction Points

6.3.2.1 Junction Points of Lines Sloped at the Same Degree

Input



Output

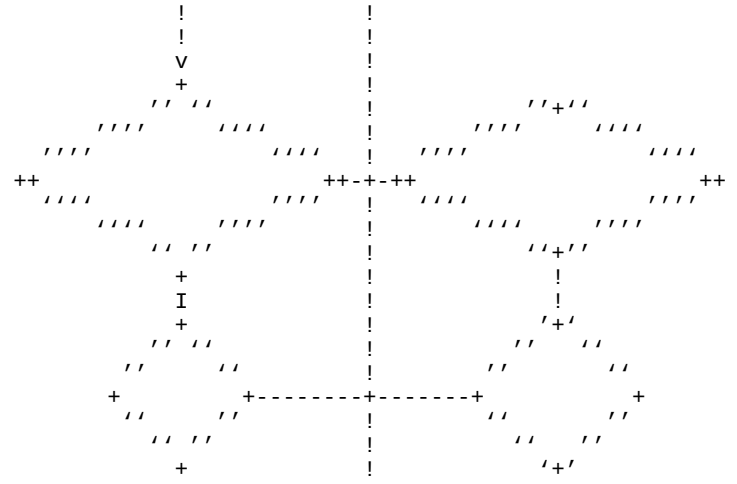


Special cases: junction points of lines sloped at the same degree

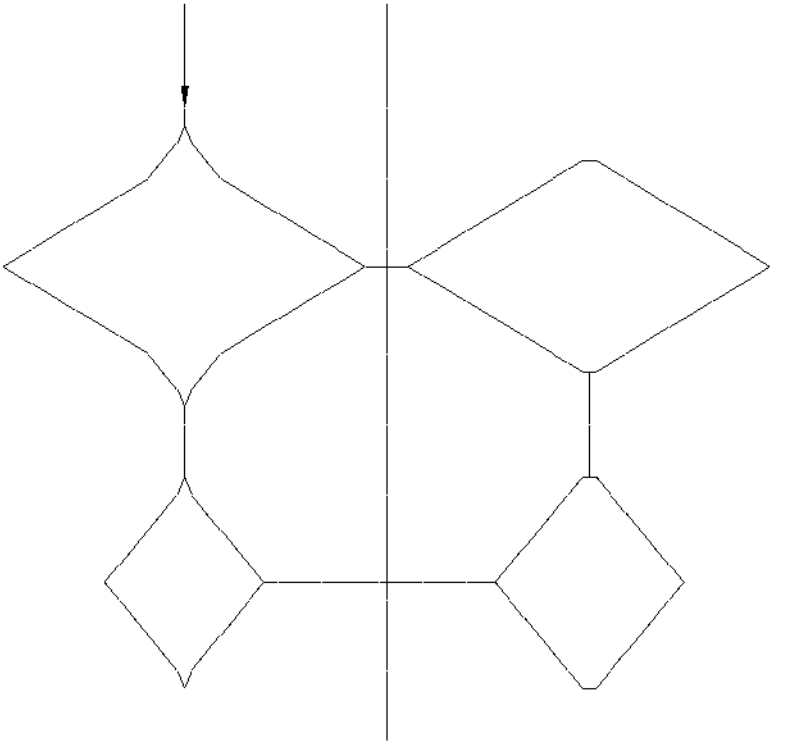
In the figures below,
any symbols can be used for
upward and downward connections.

In the figures below, only "!"
and arrows ("v" and "^")
can be used for upward and
downward connections.

Input



Output



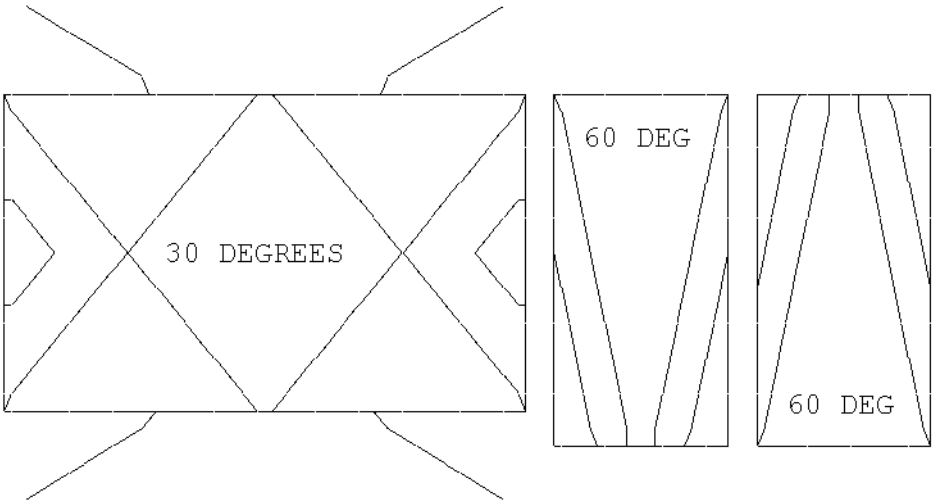
6.3.2.2 Joining Frames and Eccentric Lines

The 30- and 60-degree sloping lines are **eccentric**, i.e. they do not pass through the center. Since such a line may meet a horizontal or a vertical line, you must ensure that the lines will join.

Input

```
  ''''                          ''''
      ''''                  ''''
+---------+------+-+------+---------+ +-----------+ +--+-+-+-+--+
!''           ''   ''           ''! ! !'         '! ! ' ' ' '  ' !
!  ''       ''       ''         ''  ! ! !' 60 DEG  '! ! ' '   ' ' !
+'   ''    ''           ''    ''   '+ ! '       ' ! !'  '   '  ' !
! ''   ''''   30 DEGREES  ''''   '' ! ! '       ' ! !' '     ' ''!
! ''   ''''                 ''''   '' ! +  '     '  + +  '     '  +
+'   ''    ''           ''    ''   '+ !' '     ' '! ! '       ' !
!  ''       ''       ''         ''  ! !'  '   '  '! ! '       ' !
!''           ''   ''           ''! ! ' ' '   ' ' ! !' 60 DEG  '!
+---------+------+-+------+---------+ ! '  ' '  ' ! !'          '!
      ''''                  ''''      +--+-+-+-+--+ +-----------+
  ''''                          ''''
```

Output



6.3.3 Angles

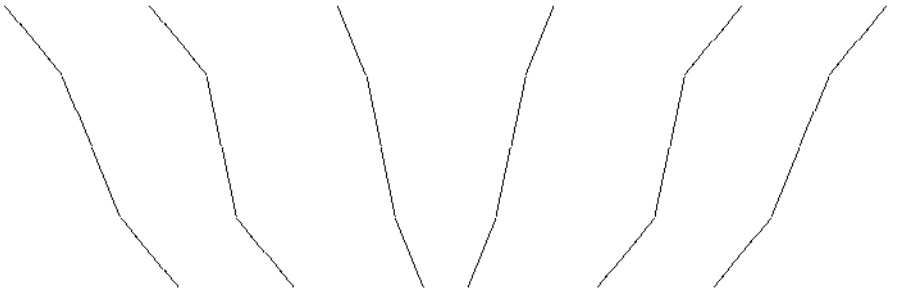
The following example shows all the angles that the TOM-DOC graphics module is able to convert. The graphics module cannot produce any other angles.

6.3.3.1 Angles of 30, 45 and 60 Degrees, EXAMPLE 1

Input

```
''          ''          '           '           ''          ''
  ''          ''          '           '           ''          ''
    '           '           '           '           '           '
     '           '           '           '           '           '
      '           '           '           '           '           '
       '           '           '           '           '           '
        ''          ''          '           '           ''          ''
          ''          ''          '           '           ''          ''
```

Output

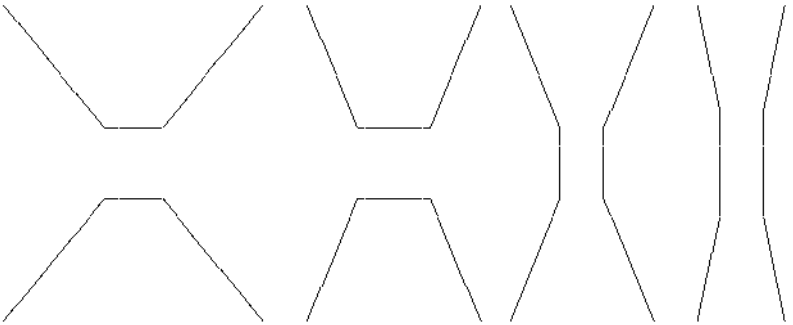


6.3.3.2 Angles with Vertical or Horizontal Lines, EXAMPLE 1

Input

```
''              ''   '          '  '        '   '    '
  ''          ''      '        '    '      '    '    '
    ''      ''         '      '      '    '      '  '
      '----'            +----+        +  +       !  !
                        +----+        !  !       !  !
      '----'            +----+        +  +       !  !
    ''      ''         '      '      '    '      '  '
  ''          ''      '        '    '      '    '    '
''              ''   '          '  '        '   '    '
```

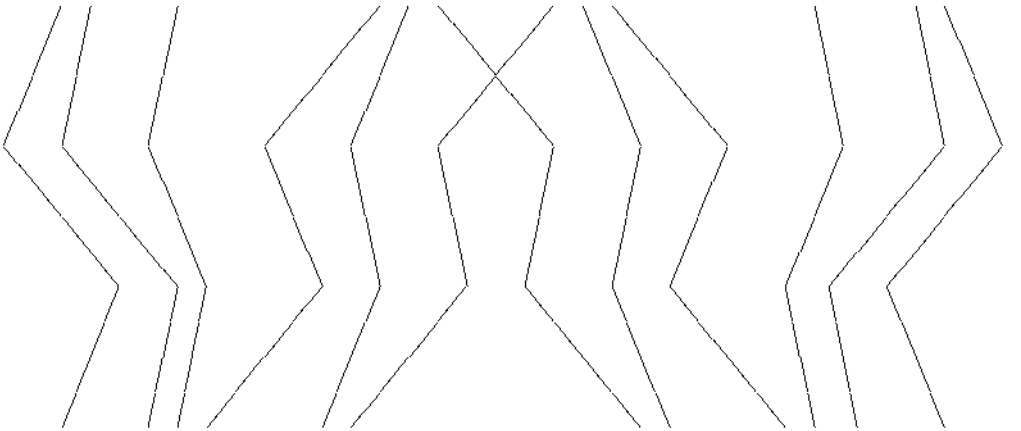
Output



6.3.3.3 Angles of 30, 45 or 60 Degrees, EXAMPLE 2

Input

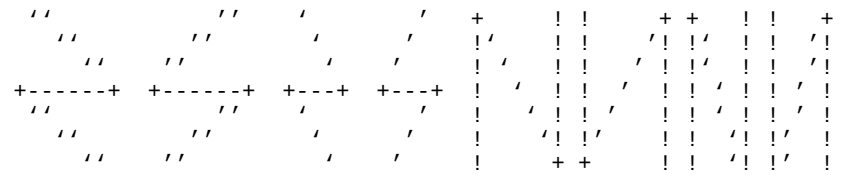
Output



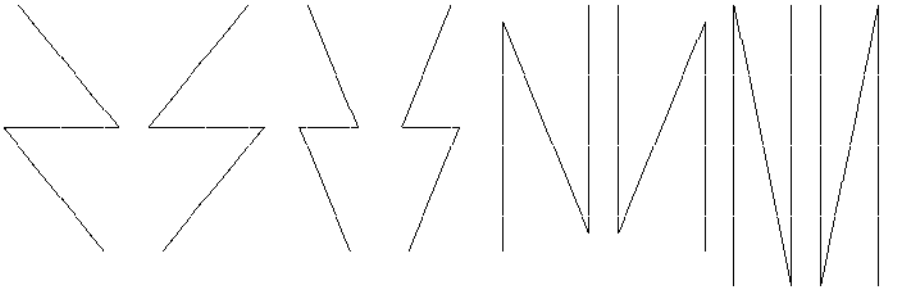
6.3.3.4 Angles with Vertical or Horizontal Lines, EXAMPLE 2

Input

```
Eingabe
```

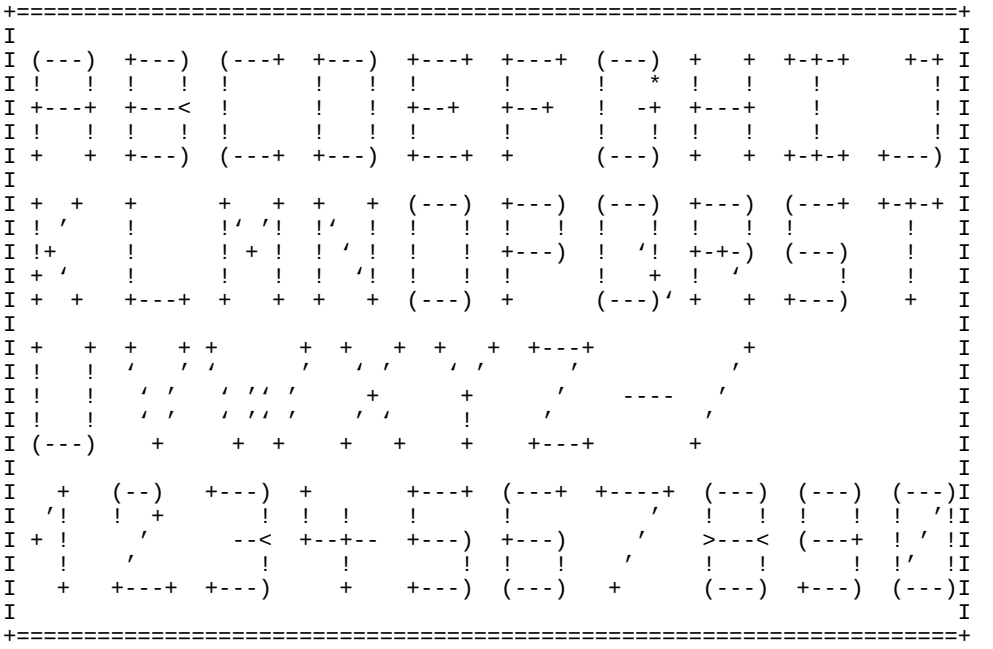


Output

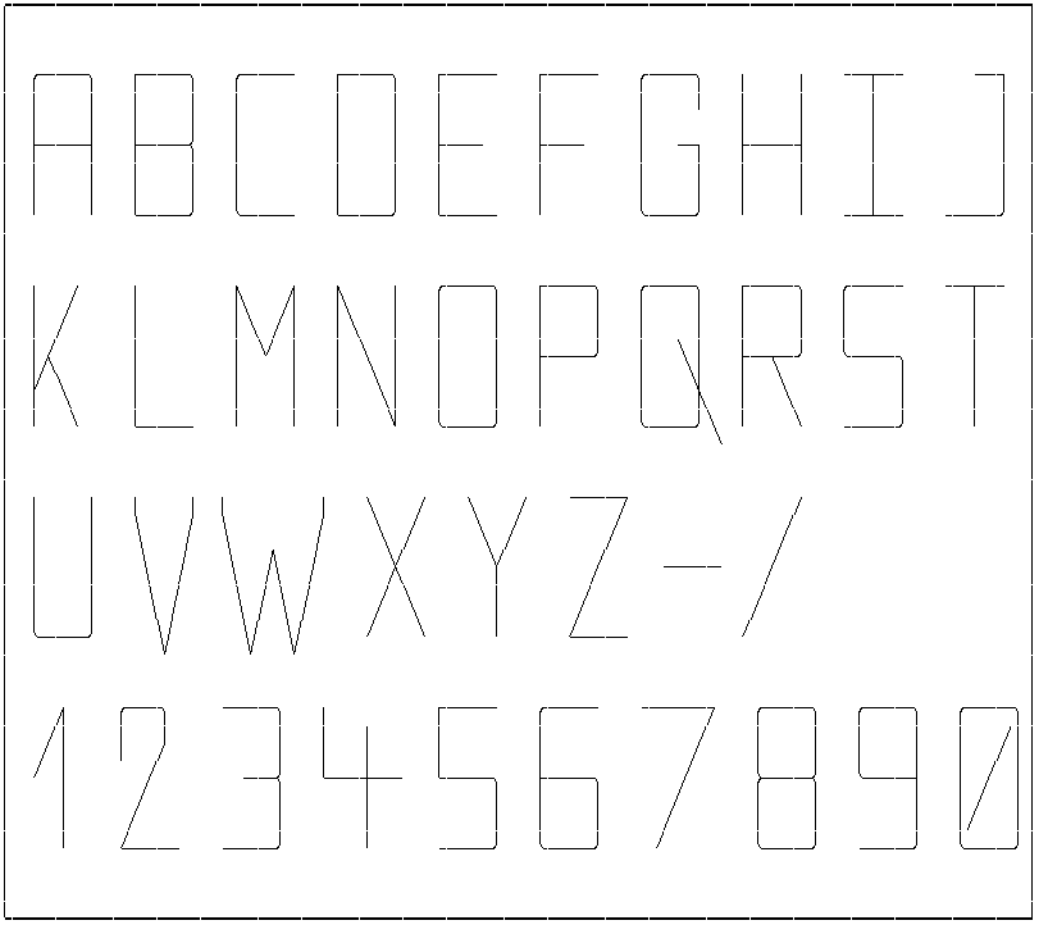


6.3.4 Letters

Input

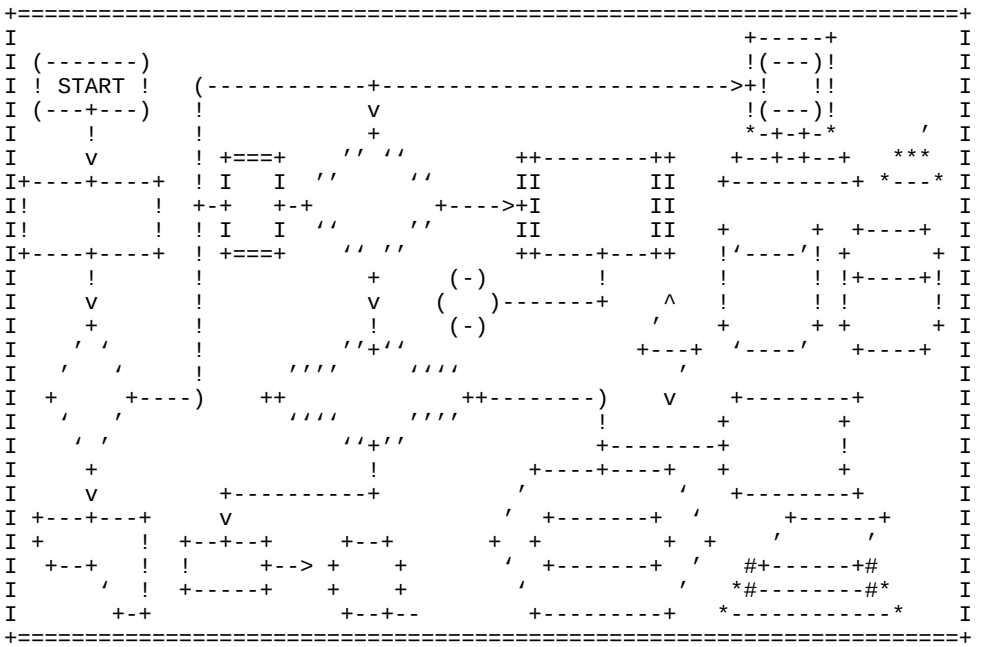


Output

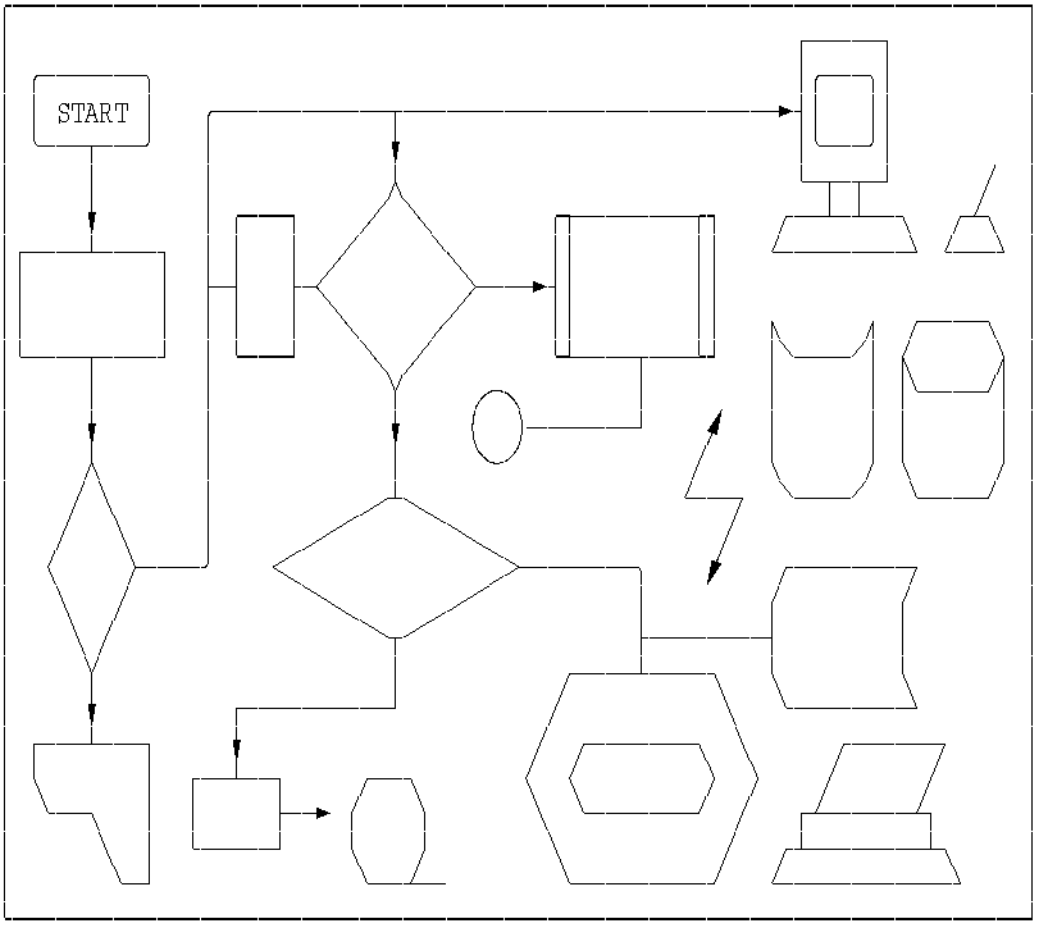


6.3.5 Flowchart Elements

Input



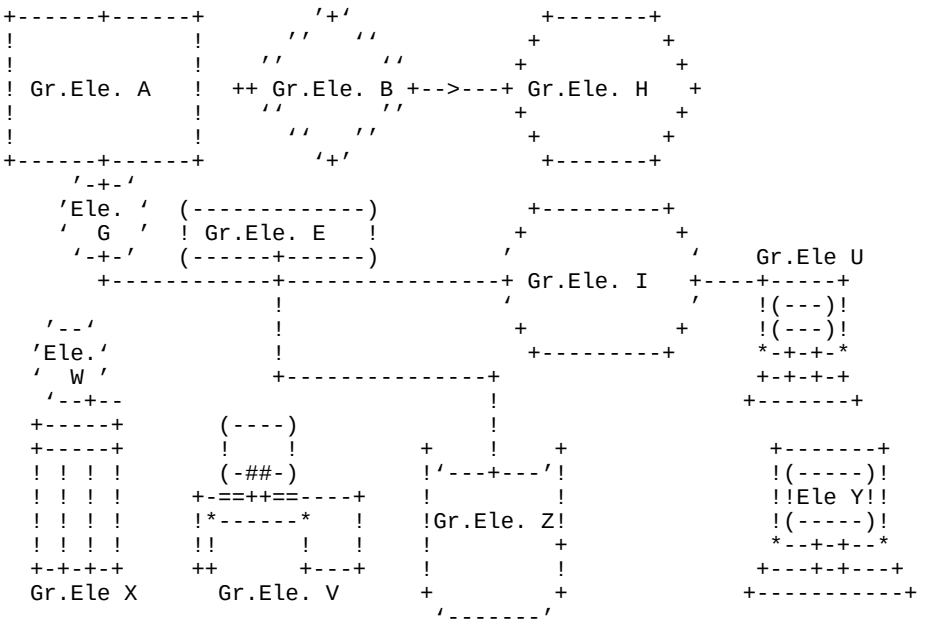
Output



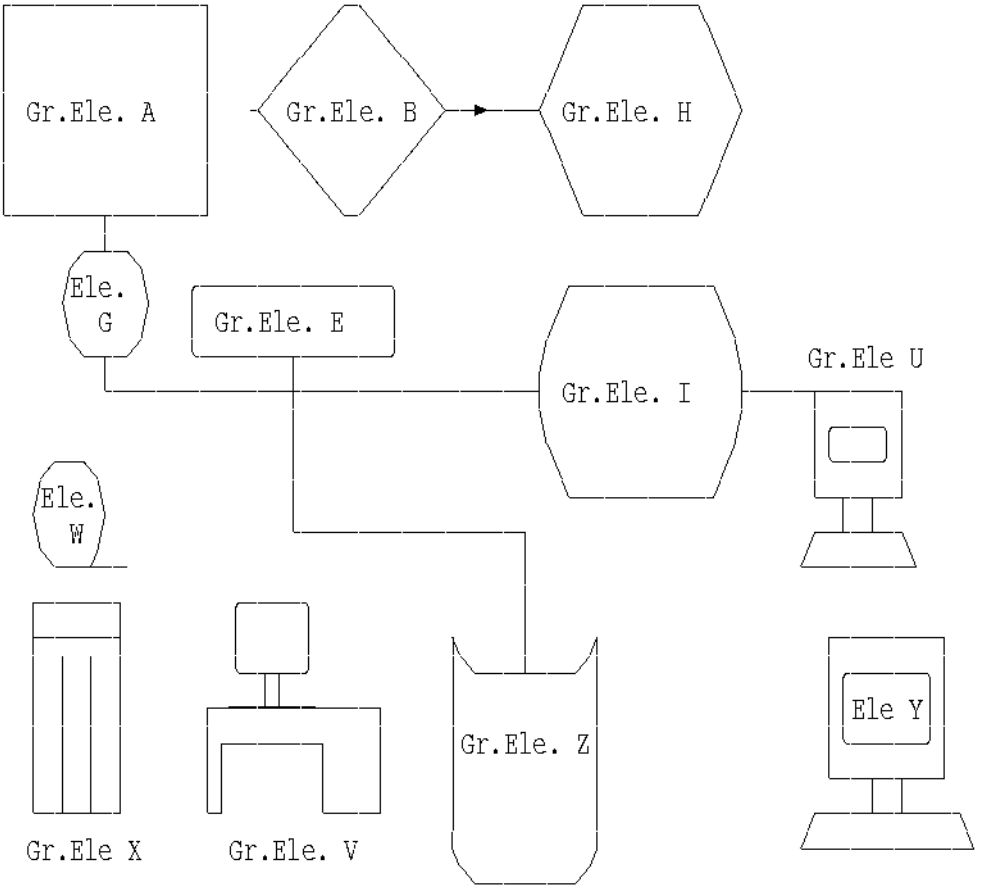
6.3.5.1 TOM-GEN Graphics Elements

The task is to generate PICTURE (see TOM-TI User's Guide).

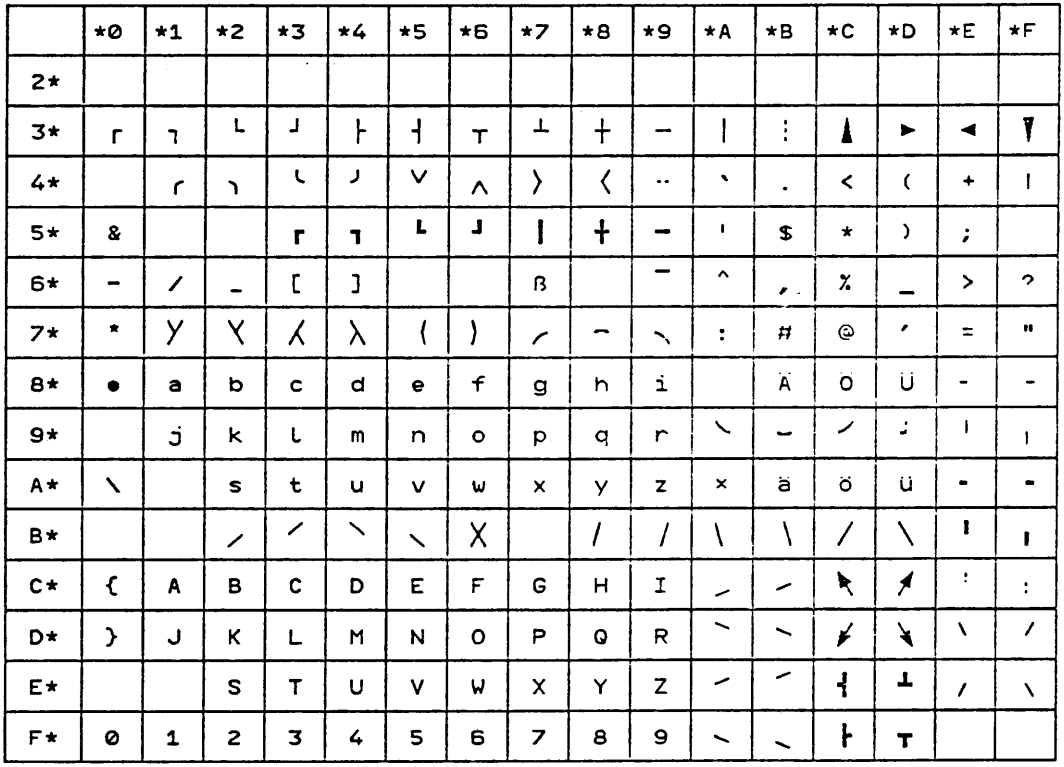
Input



Output


Gr.Ele. A
Gr.Ele. B
Gr.Ele. H
Ele.
G
Gr.Ele. E
Gr.Ele. I
Gr.Ele U
Ele.
W
Gr.Ele X
Gr.Ele. V
Gr.Ele. Z
Ele Y

6.4 The ’DE’ Character Set



	*0	*1	*2	*3	*4	*5	*6	*7	*8	*9	*A	*B	*C	*D	*E	*F
2*																
3*	┌	┐	└	┘	├	┤	┬	┴	┼	─	│	⋮	▲	►	◄	▼
4*		┌	┐	└	┘	∨	∧	⟩	⟨	¨	`	.	<	(	+	¦
5*	&			┌	┐	└	┘	¦	┼	─	'	$	*	)	;	
6*	-	/	_	[	]			ß		¯	^	,	%	_	>	?
7*	*	Y	Y	X	X	(	)	⌒	¯	⌒	:	#	@	'	=	"
8*	●	a	b	c	d	e	f	g	h	i		Ä	Ö	Ü	-	-
9*		j	k	l	m	n	o	p	q	r	⌒	⌣	⌒	⌣	¦	¦
A*	⌒		s	t	u	v	w	x	y	z	×	ä	ö	ü	-	-
B*			/	/	⌒	⌒	X		/	/	\	\	/	\	¦	¦
C*	{	A	B	C	D	E	F	G	H	I	⌒	⌒	▲	▲	:	:
D*	}	J	K	L	M	N	O	P	Q	R	⌒	⌒	▼	▼	\	/
E*			S	T	U	V	W	X	Y	Z	⌒	⌒	┤	┴	/	\
F*	0	1	2	3	4	5	6	7	8	9	⌒	⌒	├	┬		

Figure 6-1: The ’DE’ (German/English) character set

The hexadecimal value of a character is defined by its position within the table. Thus, ’81’ is the character "a" and ’3A’ the frame character "¦".

Glossary

Delimiter
The delimiter ("del") in this manual is a single character used as a marker.

Escape character
A special control character used to "escape" from the text and control output to the printer or on the screen. The escape characters in TOM-DOC are recognized by the two dots.

Filing margin
The filing margin is not to be confused with the margin of the text. If a sheet of paper has to be perforated, the filing margin is the margin in which the holes are punched. The width of the text margin has to be added to that of the filing margin to obtain the overall margin width.

Flag
A signal character with which processes can be controlled. In this manual a flag is understood to be a marker used to replace a full command.

Flowchart
A flowchart consists of different symbols (lozenges, rectangles, circles, arrows, etc.) with which the execution of a program is illustrated. Flowcharts are often criticized as being confusing because there is no set direction for the program flow. A clear alternative to the traditional flowchart is the -> structogram.

HIPO technique
HIPO is the abbreviation for "Hierarchy, Input, Process, Output". It is a software development technique based on the top-down design principle. The individual modules of the final program are designed according to the data flow. The result is a tree structure from which program segments of increasingly detailed structure can be accessed.

Index
In this manual the following is to be understood under index:

1) the keyword index;
2) the current number in an enumeration.

Integer
An integer is to be understood as a whole number, e.g. 1,2,3...1000 etc.

Leading line
A leading line links the text of an entry in a table of contents to the corresponding page number.

Metablank
The metablank is a blank character which is retained unmodified in the print file. It is not deleted, nor is it replaced by a line break.

Operand
In this manual an operand is understood as a text to be specified along with a command.

Phase
A program which can be executed independently.

Prompting screen
A prompting screen requests inputs from the user. The input fields are usually highlighted in a different color.

Qualified page number
Qualified page numbering is, for example, the numbering of pages by chapters (1-1, 2-1 etc.).

Reference variable
A reference variable is used to make reference to specific points in the text. The text points can be defined by chapter numbers, page numbers, chapter headings or by an item in an enumeration.

String
A string is an arbitrary sequence of characters. A word is a string, for example. A number, however, is not a string.

Structogram
A structogram is also known as a Nassi-Shneiderman diagram. Structograms are used to illustrate modular program structures. As opposed to the -> flowchart, which shows the chronological order of the program, the structogram shows the individual program blocks in which the processes execute in one direction only. Contrary to flowcharts, structograms only need two symbols: a rectangle for straight flows and a triangle for decisions.

System variable
The value of a system variable is preset by the system. System variables include the date and time and the page numbers of a document. System variables can be modified to a limited extent only.

References

9750 Data Display Terminal
Operating Manual

DPRINT (BS2000)
Distributed Print Services
User Guide

EDT (BS2000)
Reference Manual

BS2000/OSD
Control System Command Language

TOM-GEN (BS2000)
Generator Module
User’s Guide

TOM-M (BS2000)
Monitor
User’s Guide

TOM-REF (BS2000)
Reference Management
User’s Guide

TOM-TA (BS2000)
Text Analysis
User’s Guide

TOM-TI (BS2000)
Tool Interface
User’s Guide

SILB (BS2000)
Hyphenation Routine
User’s Guide

SPOOL (BS2000)
User Guide

TOM-DOC (SINIX)
Text Editing
User Guide

Ordering manuals

The manuals listed above and the corresponding order numbers are to be found in the List of Publications issued by Siemens Nixdorf Informationssysteme AG. New publications are listed in the Druckschriften-Neuerscheinungen (New Publications.)

You can arrange to have both of these sent to you regularly by having your name placed in the appropriate mailing list. Your local office will help you.

Index

A

B

C

D

Contents

TOM-DOC V3.2 (BS2000/OSD)

User Guide

Target group
Programmers, application designers and all users who are involved in producing documents required for development projects.
Contents
TOM-DOC is a well suited programm for producing documents required for development projects, such as studies, specifications, performance descriptions and manuals.

The manual describes the commands and functions of TOM-DOC like
– editing functions for typographical mark outs,
– editing functions for automatic
  classification of a text into chapters, sub-chapters, sections, etc., line and page breaks, automatic hyphenation in a number of national languages and automatic generation of table of contents and keyword index
– generating graphics.

Edition: **April 1995**

File: **TOM_DOC.PDF**

Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format *...@ts.fujitsu.com*.

The Internet pages of Fujitsu Technology Solutions are available at *http://ts.fujitsu.com/*...
and the user documentation at *http://manuals.ts.fujitsu.com*.

Hinweise zum vorliegenden Dokument

Zum 1. April 2009 ist Fujitsu Siemens Computers in den alleinigen Besitz von Fujitsu übergegangen. Diese neue Tochtergesellschaft von Fujitsu trägt seitdem den Namen Fujitsu Technology Solutions.

Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.

Kontakt- und Supportadressen werden nun von Fujitsu Technology Solutions angeboten und haben die Form *...@ts.fujitsu.com*.

Die Internetseiten von Fujitsu Technology Solutions finden Sie unter *http://de.ts.fujitsu.com/*..., und unter *http://manuals.ts.fujitsu.com* finden Sie die Benutzerdokumentation.