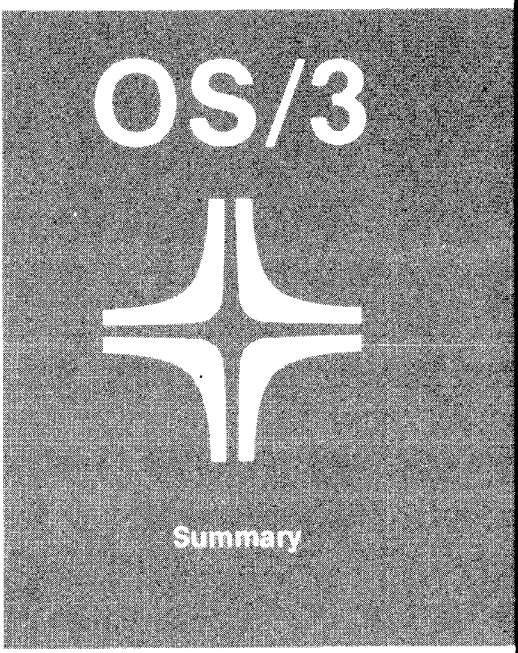
ESCORT



SPERRY UNIVAC

UP-8856
Rev. 1

H

This document contains the latest information available at the time of preparation. Therefore, it may contain descriptions of functions not implemented at manual distribution time. To ensure that you have the latest information regarding levels of implementation and functional availability, please consult the appropriate release documentation or contact your local Sperry Univac representative.

Sperry Univac is a division of the Sperry Corporation.

This document was prepared by Systems Publications using the SPERRY UNIVAC UTS 400 Text Editor. It was printed and distributed by the Customer Information Distribution Center (CIDC), 555 Henderson Rd., King of Prussia, Pa., 19406.

PRINTED IN U.S.A.

PAGE STATUS SUMMARY

ISSUE: UP-8856 Rev. 1
RELEASE LEVEL: 8.0 Forward

Part/Section	Page Number	Update Level
Cover/Disclaimer		
PSS	1	
Contents	1	
Text	1 thru 16	
User Comment Sheet		

Contents

STATEMENT FORMAT CONVENTIONS AND LANGUAGE RULES

The conventions and rules used to present statement formats are:

In the direct entry method, key in uppercase words exactly as shown. In the screen menu method, uppercase words designate words generated by ESCORT.

Lowercase words designate names, such as file or structure names, that you supply.

Brackets [ ] enclose optional parameters.

Braces { } indicate alternative parameter choices. When you use the parameter, you must pick one of the choices enclosed by braces.

Reserved words consist of all words used in ESCORT commands, statements, and clauses. You can't use these as variable field or file names, but you can use them in data you input or output or as alphanumeric constants.

ESCORT allows alphanumeric, numeric, and decimal constants.

ESCORT programs can be written in upper or lowercase, but when you're comparing or sorting alphanumeric data fields, lowercase words have greater value than their uppercase counterparts.

RESERVED WORDS

AND
BY
CHANGE
CLEARING
COMPUTING
CONTAIN
CONTAINS
CREATING
DATA
DELETE
DISPLAY
DOES
ELSE
END
ENDOFILE
ENDS
ENTER
EQ
EXIT
EXTENDING
FIRSTIME
FROM
GE
GT
IF
LE
LT
NE
NOT
OF
OR
OUTPUT
PRINT
PRINTER
SELECT
SORT
START
STARTS
SUBTOTALING
SUBTOTALS
TO
TOTALING
TOTALS
UPDATING
USING
WHILE
WITH
WORKAREA
WS
$DATE
$DAY
$LINE
$MONTH
$PAGE
$TIME
$TITLE
$YEAR

ESCORT COMMANDS LISTING

Program Commands	Structure Commands
CHANGE PROGRAM	CHANGE STRUCTURE
DISPLAY PROGRAM	DISPLAY STRUCTURE
DISPLAY PROGRAM NAMES	DISPLAY STRUCTURE NAMES
ENTER PROGRAM	ENTER STRUCTURE
PRINT PROGRAM	PRINT STRUCTURE
PRINT PROGRAM NAMES	PRINT STRUCTURE NAMES
RUN PROGRAM	SAVE STRUCTURE
SAVE PROGRAM	
Job Commands	**Other Commands**
CHANGE JOB	TERMINATE CURRENT ESCORT SESSION
DISPLAY JOB	
DISPLAY JOB NAMES	
ENTER JOB	
PRINT JOB	
PRINT JOB NAMES	
RUN JOB	
SAVE JOB	

PROGRAM STATEMENT FORMATS

CHANGE DATA Program Statement

```
CHANGE DATA OF master-file [structure]
 [FROM{WS[{structure        }]}]
      {      {structure,form}}
      {input-file[structure]  }
 [USING input-file-1 [structure],...,]
  [ input-file-5 [structure]        ]
```

WORKAREA structure [data replacement statements]

SUBTOTALING fieldnames ["literal expressions"]

TOTALING fieldnames ["literal expressions"]

```
IF{conditions}clauses [ELSE clauses] END
  {FIRSTIME  }
  {ENDOFILE  }
```

COMPUTING data replacement statements

WHILE conditions clauses END

CLEARING fieldnames

```
PRINT[{fieldnames   ["literal expressions"]}]
     [{SUBTOTALS                           }]
     [{TOTALS                              }]
     [{structure                           }]
```

```
DISPLAY[{fieldnames   ["literal expressions"]}]
       [{SUBTOTALS                           }]
       [{TOTALS                              }]
       [{structure                           }]
```

```
OUTPUT TO{WS     }{{structure     }}
         {PRINTER}{{structure,form}}
```

EXIT

PROGRAM STATEMENT FORMATS (cont)

DELETE DATA Program Statement

```
DELETE DATA OF master-file [structure]
{FROM {WS [{structure       }]}}
{     {    {structure,form  }}}
{     {input-file [structure]}}
{IF conditions               }
```

ENTER DATA Program Statement

```
ENTER DATA FROM WS [{structure       }]
                   {structure,form  }
```

WORKAREA structure [data replacement statements]

SUBTOTALING fieldnames ["literal expressions"]

TOTALING fieldnames ["literal expressions"]

```
IF {conditions} clauses [ELSE clauses] END
   {FIRSTIME  }
   {ENDOFILE  }
```

COMPUTING data replacement statements

WHILE conditions clauses END

CLEARING fieldnames

CREATING output-file [structure]

EXTENDING output-file [structure]

```
PRINT [{fieldnames ["literal expressions"]}]
      [{SUBTOTALS                         }]
      [{TOTALS                            }]
      [{structure                         }]
```

```
DISPLAY [ { fieldnames  ["literal expressions"] } ]
        [ { SUBTOTALS                           } ]
        [ { TOTALS                              } ]
        [ { structure                           } ]

OUTPUT TO { WS      } { structure      }
          { PRINTER } { structure,form }

EXIT
```

SELECT DATA Program Statement

```
SELECT DATA OF master-file [structure]
 [FROM { WS [ { structure      } ] }  ]
       [       { structure,form } ]  ]
       { input-file [structure]   }  ]

[USING input-file-1 [structure],...,
  input-file-5 [structure]           ]
```

WORKAREA structure [data replacement statements]

SUBTOTALING fieldnames ["literal expressions"]

TOTALING fieldnames ["literal expressions"]

```
IF { conditions } clauses [ELSE clauses] END
   { FIRSTIME   }
   { ENDOFILE   }
```

COMPUTING data replacement statements

WHILE conditions clauses END

CLEARING fieldnames

CREATING output-file [structure]

PROGRAM STATEMENT FORMATS (cont)

EXTENDING output-file [structure]

UPDATING input-file [structure]

OUTPUT TO {WS / PRINTER} {structure / structure,form}

PRINT [{fieldnames ["literal expressions"] / SUBTOTALS / TOTALS / structure}]

DISPLAY [{fieldnames ["literal expressions"] / SUBTOTALS / TOTALS / structure}]

EXIT

SORT DATA Program Statement

SORT DATA OF master-file [structure]
[BY field1,...,field6]
CREATING output-file [structure]

PROGRAM STATEMENT AND CLAUSE CROSS-REFERENCE

Clauses	ENTER DATA	CHANGE DATA	SELECT DATA	SORT DATA	DELETE DATA
BY				X	
CLEARING	X	X	X		
COMPUTING	X	X	X		
CREATING	X		X	X	
DISPLAY	X	X	X		
END	X	X	X		
EXIT	X	X	X		
EXTENDING	X		X		
IF	X	X	X		X
OUTPUT TO	X	X	X		
PRINT	X	X	X		
SUBTOTALING	X	X	X		
TOTALING	X	X	X		
UPDATING			X		
USING		X	X		
WHILE	X	X	X		
WORKAREA	X	X	X		

CLAUSE FORMATS

Clause	Format
BY	BY field-1,..., field-6
CLEARING	CLEARING fieldnames
COMPUTING	COMPUTING data replacement statements
CREATING	CREATING output-file [structure]
DISPLAY	DISPLAY [{ fieldnames ["literal expressions"] / SUBTOTALS / TOTALS / structure }]
END	END
EXIT	EXIT
EXTENDING	EXTENDING output-file [structure]
IF	IF { conditions / FIRSTIME / ENDOFILE } clauses [ELSE clauses] END
OUTPUT TO	OUTPUT TO { WS / PRINTER } { structure / structure,form }
PRINT	PRINT [{ fieldnames ["literal expressions"] / SUBTOTALS / TOTALS / structure }]

CLAUSE FORMATS (cont)

Clause	Format
SUBTOTALING	SUBTOTALING fieldnames ["literal expressions"]
TOTALING	TOTALING fieldnames ["literal expressions"]
UPDATING	UPDATING input-file [structure]
USING	USING input-file-1 [structure],..., input-file-5 [structure]
WHILE	WHILE conditions clauses END
WORKAREA	WORKAREA structure [data replacement statements]

ARITHMETIC OPERATORS AND RULES

Arithmetic Operators	
+	Addition
-	Subtraction
*	Multiplication
/	Division
:	Modulo (a division operation where the result is the remainder portion)

In arithmetic expressions, multiplication, division, and modulo are performed before addition and subtraction.

The sequence for performing operations of the same level is left to right.

Expressions enclosed within parentheses are evaluated first. When using nested parentheses, evaluation begins with the innermost set and works outward.

RELATIONAL OPERATORS

Alphabetic	Symbolic	Meaning
EQ	=	Equal to
NE	< >	Not equal to
GT	>	Greater than
LT	<	Less than
GE	>=	Greater than or equal to
LE	<=	Less than or equal to

CHARACTER STRING OPERATORS

STARTS WITH "string"
ENDS WITH "string"
CONTAINS "string"
DOES NOT START WITH "string"
DOES NOT END WITH "string"
DOES NOT CONTAIN "string"

LOGICAL OPERATORS

In addition to the relational and special character string operators, there are two logical operators, OR and AND, which are used to connect one or more conditionals.

SYSTEM UTILITY FIELDS

ESCORT provides eight system utility fields you can specify in any ESCORT program. These fields are always prefixed by a dollar sign ($).

Field	Length	Function
$DATE	8A	Gives you the current system date (mm/dd/yy)
$DAY	2N	Gives you the current system day
$LINE	3N	Sets the system printer to a specified line number before printing
$MONTH	2N	Gives you the current system month
$PAGE	4N	Sets the system page number to a specific page number
$TIME	5A	Gives you the current system time (hh:mm)
$TITLE	40A	Allows you to provide a title line for printed reports
$YEAR	2N	Gives you the current system year

STRUCTURE DESIGNATIONS

In the LENGTH/TYPE parameter, the following symbols are used to designate field type:

A - alphanumeric

N - numeric

V - assumed decimal

P - packed decimal

B - binary

In the KEYS parameter, the letter K, followed by a number 1 through 6, is used to designate key values.

EDIT CODES

Symbol	Function
Z	Suppress leading zeros
*	Replace leading zeros with asterisks
,	Insert commas where applicable
-	Insert sign after least significant digit. If the number is: Positive - the sign inserted is a blank character Negative - the sign inserted is a minus symbol

EDIT CODES (cont)

Symbol	Function
/	Insert slash character where applicable in date field. In 4-digit date field, one slash is inserted. In 6-digit date field, two slashes are inserted.
*$	Fixed dollar sign is assumed when the asterisk is used in the first position of the field.
$	Floating dollar sign is assumed when the asterisk is omitted.

NOTE:

When using a form to output your data, the slash, sign, comma codes, and the decimal point are characters that are added to the field and should be allowed for in your output.

For example, on a form, to output a 5V2 field with a comma, you must allow for nine positions: eight for the number and decimal point and one for the comma.

FUNCTION KEY TABLE

Key	Structure Processor	Program Mode	Tutorial Mode	Run-Time Processor
F1	End of input, return to master menu	Not used	Not used	Not used
F2	Cancel display output	Not used	Present a menu of subjects for which tutorial mode HELP is available	Not used
F3	Not used	Cancel current screen and return to previous screen	Abort current tutorial processing and return to beginning of tutorial mode	Not used
F4	Abort structure and return to menu	Terminate free-form input and return to previous menu	Abort tutorial processing and return to command mode	Terminate data input or workstation and return to caller

NOTES:

1. *In the run-time F4 key description, caller refers to the ESCORT processor that called for a run-time function. The processor can be either program mode or tutorial mode. Structures do not use run-time functions.*

2. *The XMIT key (instead of F4) in program mode will also terminate input and return you to the previous menu.*