The SuperPascal User Manual

PER BRINCH HANSEN [1]

School of Computer and Information Science
Syracuse University, Syracuse, NY 13244, USA

November 1993

Abstract: This report explains how you compile and run *SuperPascal* programs [Brinch Hansen 1993a].

1 Command Aliases

If you are using *SuperPascal* under Unix, please define the following command aliases in the file *.cshrc* in your home directory:

alias sc <*path name of an executable compiler sc*>
alias sr <*path name of an executable interpreter sr*>

2 Program Compilation

You compile a *SuperPascal* program by typing the command

sc

followed by a return. When the message

source =

appears, type the name of a program textfile followed by a return. After the message

code =

type the name of a new program codefile followed by a return.

Example:

sc
source = *sortprogram*
code = *sortcode*

If the compiler finds errors in a program text, the errors are reported both on the screen and in the textfile *errors*, but no program code is output.

3 Program Execution

You run a compiled *SuperPascal* program by typing the command

sr

followed by a return. When the message

code =

appears, type the name of a program codefile followed by a return. After the message

select files?

you have a choice:

1. If you type *no* followed by a return, the program will be executed with text input from the *keyboard* and text output on the *screen*.

2. If you type *yes* followed by a return, you will first be asked to name the input file:

input =

Type the name of an existing textfile or the word *keyboard* followed by a return. Finally, you will be asked to name the output:

output =

Type the name of a new textfile or the word *screen* followed by a return.

Examples:

sr
code = *sortcode*
select files? *no*

sr
code = *sortcode*
select files? *yes*
input = *testdata*
output = *screen*

4 Compile-time Errors

During compilation, the following program errors are reported:

- *Ambiguous case constant:* Two case constants denote the same value.
- *Ambiguous identifier:* A program, a function declaration, a procedure declaration, or a record type introduces two named entities with the same identifier.
- *Forall statement error:* In a restricted *forall* statement, the element statement uses a target variable.
- *Function block error:* A procedure statement occurs in the statement part of a function block.
- *Function parameter error:* A function uses an explicit or implicit variable parameter.
- *Identifier kind error:* A named entity of the wrong kind is used in some context. (Constants, types, fields, variables, functions and procedures are different kinds of named entities.)
- *Incomplete comment:* The closing delimiter } of a comment is missing.
- *Index range error:* The index range of an array type has a lower bound that exceeds the upper bound.
- *Number error:* A constant denotes a number outside the range of integers or reals.
- *Parallel statement error:* In a restricted parallel statement, a target variable of one process statement is also a target or an expression variable of another process statement.
- *Procedure statement error:* In a restricted procedure statement, an entire variable is used more than once as a restricted actual parameter.
- *Recursion error:* A recursive function or procedure uses an implicit parameter.
- *Syntax error:* The program syntax is incorrect.
- *Type error:* The type of an operand is incompatible with its use.
- *Undefined identifier:* An identifier is used without being defined.

5 Run-time Errors

During program execution, the following program errors are reported:

- *Channel contention:* Two processes both attemp to send or receive through the same channel.
- *Deadlock:* Every process is delayed by a send or receive operation, but none of these operations match.
- *False assumption:* An assume statement denotes a false assumption.
- *Message type error*: Two processes attempt to communicate through the same channel, but the output expression and the input variable are of different message types.
- *Range error:* The value of an index expression or a *chr, pred,* or *succ* function designator is out of range.
- *Undefined case constant:* A case expression does not denote a case constant.
- *Undefined channel reference:* A channel expression does not denote a channel.

6 Software Limits

If a program is too large to be compiled or run, the software displays one of the following messages and stops. Each message indicates that the limit of a particular software array type has been exceeded:

- *Block limit exceeded:* The total number of blocks defined by the program and its function declarations, procedure declarations, *forall* statements, and process statements exceeds the limit *maxblock*.
- *Branch limit exceeded:* The total number of branches denoted by all statements in the program exceeds the limit *maxlabel*.
- *Buffer limit exceeded:* The size of the compiled code exceeds the limit *maxbuf*.
- *Case limit exceeded:* The number of case constants exceeds the limit *maxcase*.
- *Channel limit exceeded:* The number of channels opened exceeds the limit *maxchan*.
- *Character limit exceeded:* The total number of characters in all word symbols and identifiers exceeds the limit *maxchar*.

- *Memory limit exceeded:* The program execution exceeds the limit *maxaddr*.
- *Nesting limit exceeded:* The level of nesting of the program and its function declarations, procedure declarations, parallel statements, and *forall* statements exceeds the limit *maxlevel*.
- *String limit exceeded:* The number of characters in a word symbol, an identifier, or a character string exceeds the limit *maxstring*.

The standard *software limits* are:

maxaddr	=	100000	maxchar	=	10000
maxblock	=	200	maxlabel	=	1000
maxbuf	=	10000	maxlevel	=	10
maxcase	=	128	maxstring	=	80
maxchan	=	10000			

If these limits are too small for compilation or execution of a program, the limits must be increased by editing a common declaration file and recompiling both the compiler and the interpreter [Brinch Hansen 1993b].

References

[1] Brinch Hansen, P. (1993a) The programming language SuperPascal. School of Computer and Information Science, Syracuse University, Syracuse, NY.

[2] Brinch Hansen, P. (1993b) The SuperPascal software notes. School of Computer and Information Science, Syracuse University, Syracuse, NY.