12.5. Page

`Page(f)` implies an implementation–defined effect on the textfile `f`, such that any text subsequently written to `f` will appear at the top of a new page when `f` is printed. If `f` is not empty, and the last component of its sequence is not an end–of–line marker, then `Page(f)` performs an implicit `Writeln(f)`. If the parameter list is omitted, the textfile program parameter `Output` is assumed. It is an error if `f` is undefined or if `f` is not in generation mode.

The effect of reading a file variable to which `Page` was previously applied is implementation–dependent.

13. Programs

A Pascal program consists of a program heading and a block.

Program = *ProgramHeading* ";" *Block* "." .

ProgramHeading = "`program`" *Identifier* [*ProgramParameterList*] .

ProgramParameterList = "(" *IdentifierList* ")" .

The identifier following the symbol `program` is the program name; it has no further significance inside the program. Each identifier in the program parameter list is called a program parameter, and denotes an entity that exists outside the program and that, therefore, is called *external*. It is through its program parameters that the program communicates with its environment.

When a program is activated, each program parameter is bound to the external entity that it represents. For those program parameters that are file variables, the binding is implementation–defined; for all other program parameters, the binding is implementation–dependent.

Each program parameter, with the exception of `Input` and `Output`, must be declared in the variable declaration part of the program's block. In the case of `Input` or `Output`, the occurrence of the identifier in the program parameter list has the effect of implicitly declaring the identifier in the program block to be a textfile, and implicitly performing a `Reset(Input)` or `Rewrite(Output)` at the commencement of each activation of the program.