

Programming Standards – General

R. France, P. Koushik & B. Cline
7 February 1992

1. Consistency

- each piece of code should be written in an internally consistent style, whether or not it conforms to the following style guidelines.
- programmers altering others' code should use either the original programmer's style or the recommended style. They are encouraged to change deviant styles to the standard, but not to their personal style when it differs from the standard.

2. Modularity

- procedures should be no more than 30 lines long.
- compilation files should include no more than 300 lines.
- a compilation file should include procedures related in function and call dependencies.
- all modules requiring more than one compilation file must use Makefiles.

3. Documentation

- every file and every procedure must include header comments.
 - files: author's full name, date, purpose of procedures in file, history.
 - procedures: author, date, parameters (noting in or out), purpose of the procedure.
 - headers should include descriptions of algorithms when not clear from context.
 - any limitations or anticipated extensions should also be noted.
- each conceptual section of a procedure should have what it does explained in a comment.
 - readers should be able to understand the function of the code reading only the comments.
- any obscure code must include line comments explaining the obscurities.
 - similarly, any “language tricks” should include line comments.
- code that is altered, untested, or known to be buggy must include warning comments.
- if one programmer changes another's code, s/he should document hi/r changes with line comments including the date, hi/r initials, and the reason for the change.
- machine dependencies should be noted where they occur and in file headers.

4. White space

- statements in the body of a selector, a loop in an algorithmic language, or a choice in a declarative language must always be indented.
- indentations must be uniform (3-4 spaces is recommended).
- logically discrete actions within a procedure should be separated by blank lines.
- form feeds (control-L's) should be used to separate procedures (or procedure groups) in a file.

5. Naming conventions

- all names should be recognizable English words or phrases.
 - except for loop control, no variables should have single-letter names.
 - variable names should be descriptive of the object denoted; procedure names descriptive of the action or test performed.
- capital letters and underscores can be used in different combinations to provide recognizable styles for variables, constants, types, and procedure names.

6. Error handling

- no error condition should ever cause a system crash; local tests should always be performed for error conditions that can cause a crash. When an error condition is found, the procedure should always return with a distinctive value.
- error messages should always be printed at the lowest level at which the error is found.
 - they may also be printed by higher level modules, if more information is known there.
 - all error messages should include:
 - some indication of where in the code the error occurred (e.g., procedure name),
 - a description of the error that can be understood by people other than the programmer,
 - (wherever feasible) the values that caused the condition.
- it is also desirable that error messages fit on a single line (ha, ha).

7. Version Control

- multiple versions of a module should be kept in separate directories, with README files in the parent directory explaining version differences.
- each version directory should contain a README file explaining features of the version.
- periodic backups to tape or to another computer are essential. No single disk or computer should ever be considered safe.

8. Testing and quality assurance

- programming is not a solitary activity. Code walk-throughs and cross-testing are highly recommended.
- no module can be deemed completed until it is thoroughly tested and tidied up, and until all the paperwork – external documentation, user's manuals, archived copies, and so forth – is done.