The C language has grown in popularity tremendously in recent years.

Its sources are found in 1972, in the Bell laboratories, in order to develop a version

laptop from the unix operating system. It is a structured programming language, but

very "close" to the machine.

Publication in 1978 of "The C programming language" by Kernighan and Ritchie: definition

classic C.

The development of C compilers by other houses made it necessary to define

a precise standard: the ANSI-C standard.

1983: Development by AT&T of C ++

1988: Second edition of the book "The C programming language"

1990: ANSI-C ++ standard

The success of C is due to the facts that:

- It is a universal language: C is not oriented towards a specific field of applications (at

opposite of FORTRAN: scientific applications, COBOL: commercial applications).

- It is a compact language: C is based on a core of limited functions and operators,

allowing the formulation of simple and effective expressions.

- It is near the machine: as it was initially developed to program the system

UNIX, it offers operators very close to those of machine language and functions which

allow a simple and direct access to the internal functions of the computer (for example the

memory).

- It is fast since it is close to the machine.

- It is portable: respecting the ANSI-C standard it is possible to use the same

program on any other operating system in possession of a C compiler. It is

which has now become the programming language for microcomputers.

- It is extensible: C does not only consist of standard functions, the language is

run by libraries of private functions or delivered by many houses of

development.

Disadvantages:

- The possibility of compact expressions leads to the risk of ending up with

incomprehensible programs (for others, but also for ourselves), hence the

need to include comments in programs.

- It is language close to the machine, so it is dangerous. Although it is a language of

structured programming, it does not require us to adopt a programming style (like,

for example PASCAL). The programmer therefore has a lot of freedom, but also

responsibilities: he must ensure that he adopts a clean, solid and

understandable.