

Tecniche di programmazione in chimica computazionale

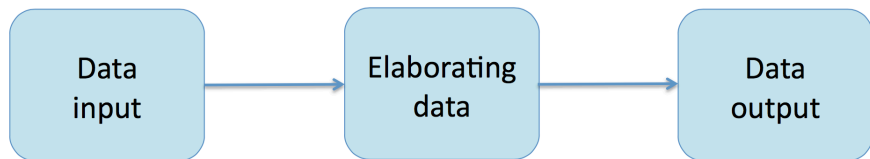
Introduction to Fortran

Emanuele Coccia

Dipartimento di Scienze Chimiche e Farmaceutiche

General flowchart

Goal for a computer: **executing** a set of **instructions** to solve a given **numerical problem**



- **Reproducible syntax** to write instructions for a computer

- **Reproducible syntax** to write instructions for a computer
- **Separation** between the physical support and instructions

- **Reproducible syntax** to write instructions for a computer
- **Separation** between the physical support and instructions
- **Semantics**: assign a meaning to the syntactic forms of the language

- **Computer Science:** best method to translate a scientific model into a code

- **Computer Science**: best method to translate a scientific model into a code
- Start from the general instructions: **input**, **do operations**, **output**

- **Computer Science**: best method to translate a scientific model into a code
- Start from the general instructions: **input**, **do operations**, **output**
- From **the general problem** to the identification of the main sections

- **Computer Science**: best method to translate a scientific model into a code
- Start from the general instructions: **input**, **do operations**, **output**
- From **the general problem** to the identification of the main sections
- Simplify towards smallest pieces of code (**stepwise refinement**)

Define an algorithm

- To solve a problem, the proposed approach has to be:

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - ① clear and univocally defined

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - 1 clear and univocally defined
 - 2 effective

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - 1 clear and univocally defined
 - 2 effective
 - 3 finite

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - 1 clear and univocally defined
 - 2 effective
 - 3 finite
- Two steps:

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - ① clear and univocally defined
 - ② effective
 - ③ finite
- Two steps:
 - ① Develop or choose an algorithm

Define an algorithm

- To solve a problem, the proposed approach has to be:
 - ① clear and univocally defined
 - ② effective
 - ③ finite
- Two steps:
 - ① Develop or choose an algorithm
 - ② Coding the algorithm

- From **high-** to **low-level** (machine) language: the role of the compiler

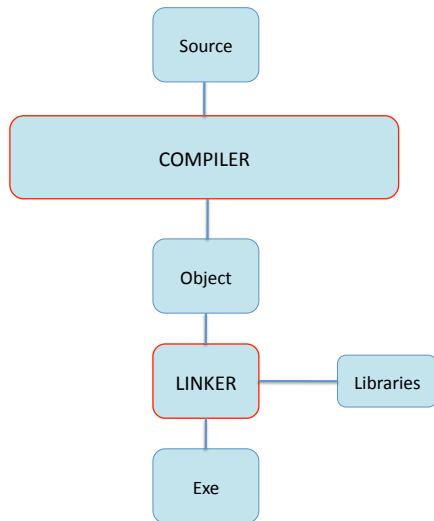
Compiling a code

- From **high-** to **low-level** (machine) language: the role of the **compiler**
- From a **source code** (e.g., name.f90) to an **executable**

Compiling a code

- From **high-** to **low-level** (machine) language: the role of the **compiler**
- From a **source code** (e.g., name.f90) to an **executable**
- Two main steps:
 - 1 Translating the source code into an **object** file, containing meta-instructions
 - 2 Converting the **object** (and possibly other ones in **libraries**) into an **executable**

Compiling a code



- **Syntax** errors: recognised by the compiler and (usually) easy to fix

Errors in programming

- **Syntax** errors: recognised by the compiler and (usually) easy to fix
- **Semantic** errors: use debugging options to find and fix them, **structured programming** to avoid them

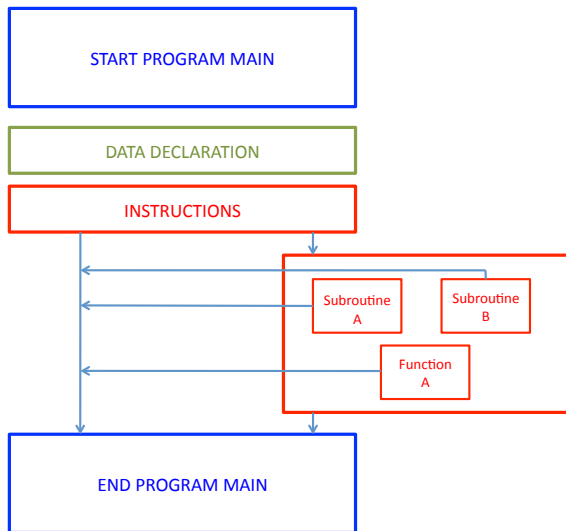
- First high-level language

- First high-level language
- Largely used for scientific and technical purposes

- First high-level language
- Largely used for scientific and technical purposes
- Born in the mid-50s: FORMula TRANslator

- First high-level language
- Largely used for scientific and technical purposes
- Born in the mid-50s: FORMula TRANslator
- Standard releases:
 - FORTRAN4
 - FORTRAN66
 - FORTRAN77
 - Fortran90
 - Fortran95
 - Fortran03
 - Fortran08
 - ...

Generic (Fortran) code



- Open a terminal

Our first code

- Open a terminal
- Type `vi hello_world.f90`

Our first code

- Open a terminal
- Type `vi hello_world.f90`
- Let us do it together (`hello_world.f90`)

- `ifort -c hello_world.f90` → produces `hello_world.o`

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:
 - ① `-o object (executable)`

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:
 - 1 `-o` object (executable)
 - 2 `-c` compile only

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:
 - 1 `-o` object (executable)
 - 2 `-c` compile only
 - 3 `-pg` profiling

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:
 - 1 -o object (executable)
 - 2 -c compile only
 - 3 -pg profiling
 - 4 -O_n optimisation level

Compiling and running

- `ifort -c hello_world.f90` → produces `hello_world.o`
- `ifort -o hello_world.x hello_world.f90` → produces the executable `hello_world.x`
- (Some) compiler options:
 - 1 `-o` object (executable)
 - 2 `-c` compile only
 - 3 `-pg` profiling
 - 4 `-O`n optimisation level
 - 5 ...

- Two types of files: **binary** or **text**
 - Binaries can be programs or data files written by programs or the operating system
 - Text files are those that can be read by us human beings
- Two categories of files: **programs** or **data**
- Either a file contains a set of instructions to be executed by the CPU (a program) or it contains information (data)