CHAPTER 1 - INTRODUCTION TO COMPUTER PROGRAMMING

1.1

A program is a set of instructions that a computer must follow systematically to perform a certain task.

Computational thinking is the set of mental processes involved in the formulation of a problem and its solution, so that a machine can actually execute them.

1.2

All computers use the Von Neumann Architecture, whose main components are:

- CPU (Central Processing Unit) : performs basic operations specified in the program

- memory : necessary to store both data and instructions

- input/output devices

- BUS : communication channels connecting all components

The CPU speaks and understand machine language only; it’s binary (0,1) and each digit represents a bit of information.

Computer memory is divided into bytes (=8 bits) which stores a single datum.

The most widespread encoding system for characters is ASCII (American Standard Code for Information Interchange), which is now being replaced by unicode.

1.3

Programs are usually stored in the mass memory.

When the program is run it is copied from the mass memory to the central memory (RAM), the processed by the CPU which executes all information in sequence.

So that the CPU can execute the program, it must be translated from source code to machine language in one of two ways:

1. Compilation - after the software is developed it is turned into an executable file. Execution times are faster but if errors have occurred the whole program must be redesigned. Example: C

2. Interpretation - translation happens instruction by instruction. It is slower but easier to correct if errors have occurred. Example: Python.

1.4

Programming languages are mainly divided in:

1. Declarative - describe reality and declare the required objective

2. Procedural - define algorithms which lead to the goal step by step

1.5

The program development cycle is:

1. Requirements (=objectives to be achieved)

2. Analysis (=understand the requirements)

3. Design (=create algorithms needed to satisfy the requirements)

4. Implementation (=coding)

5. Testing (=run the software)

6. Debugging (=find mistakes, fix, go back to point 3, repeat)

7. Execution (=publish the software, use it)

1.6

An algorithm is the finite sequence of coded actions that leads to the solution of a problem, crucial features:

-the sequence of statements must be finite

-the procedure must lead to a result

-the statements must be materially executable

-the statements must be expressed unambiguously

They are represented by flow charts and divided into blocks, with different shapes:

-oval (=beginning and end blocks)

-rhomboids (=input and output elements)

-rectangles (=execution elements with one or more elementary actions)

-diamonds (elements of decision or choice)

-arrows (=flow of execution of the algorithm)