CHAPTER 5 - DECISION-MAKING STRUCTURES AND BOOLEAN EXPRESSIONS

5.1

The sequential execution of a piece of software code refers to sequential stuctures that use linear flow of choices.

A control structure executes a set of statements only if determined circumstances occur: to this purpose a decision-making structure (=selection structure) must be applied.

A single-alternative decision-making structure is used when only one alternative execution path is possible: if the given condition is true, the alternative path is taken, otherwise the structure is excluded.

Decision-making structures are implemented by means of conditional statements (=instructions that deviate the sequential execution flow by adding alternatives)

A conditional statement is the if statement, it’s syntax for a single alternative decision-making structure is:

if condition: (this line is the clause)

statement1 (this is the block of code to

statement2 be executed if the condition

statement3 is true)

etc.

5.2

Boolean expressions are expressions that, when evaluated, only return a Boolean value (True or False), they make use of comparison operators (<,>,etc.).

True and False (Bool values) are of bool tyoe and not strings.

Comparison operators (=rational operators) are: == , > , < , >= , <= , != . They only act on variables and values of the same data type.

5.3

Double alternative decision-making structures imply two possible execution paths.

The general syntax is based on the if-else statement:

if condition:

statement1

statement2

etc.

else:

statement3

statement4

etc.

If-else conditions can be nested

5.4

An if-elif-else statement is a more readable if-else nested statement, the syntax is:

if condition1:

statement1

statement2

etc.

elif condition2:

statement3

statement4

etc.

…

elif conditionN:

statements

else:

statements

5.5

Boolean or logical operators allow expressing cumulative or alternative multiple conditions, they are:

- and —> works with two or more Boolean expressions, returns True if all expressions are true

- or —> works with two or more Boolean expressions, returns True if at least one expression is true

- not —> works with a Boolean expression, returns True if it is False

5.6

A pass statement is a void statement, which is a placeholder for the code which is going to be completed, avoiding that incomplete constructs may return an error and block the execution of the code.