Revision

- Unlike the natural languages like English, a programming language is an artificial language that is used to instruct a computer to do a task.
- Programming languages can be categorised into low-level and high-level languages.
    - The machine language and assembly language are considered low-level languages, and the rest are called high-level languages.
    - *The machine language is the only language that can be directly executed* – all other languages must be translated into the machine language to execute.
- It is also generally agreed that programming languages can be divided into four generations:
    - The first generation programming language is often referred to as the machine language.
    - The second generation language has a one-one relationship to the machine language, and needs a program called “assembler” to translate into the language that can be directed executed by computer.
    - The third generation language is often referred to as “imperative” language, and is platform independent.
    - The fourth generation languages are referred to as the declarative languages.
- In terms of how a language is translated to the machine language, programming languages can also be categorised into compiled and interpreted languages.
    - For a compiled language, the whole program is compiled (translated) before the output is produced. The implication is that the whole program must syntax error-free before the it can be compiled.
    - For an interpreted language, the program is interpreted one line at a time, until finish or an error is encountered.
- The program that is used to translate high-level language to machine language is called a compiler or interpreter.
- A programming language can be said to be either static typing or dynamic typing.
    - For a static typing language, you must specify a type to create a variable.
    - For a dynamic typing language, you don’t have to because a variable is merely a reference or pointer.
- The problem solving process can be divided into the following four stages (or steps)
    - Analysis: to have a clear understanding of what’s given and what’s required, i.e., to define the requirements and specifications.
    - Design: to develop appropriate algorithms.
    - Implementation: choose a programming language and develop a solution.
    - Testing: execute the solution using a set of selected data.
- Programming paradigm is a way of organising code in an application. The following describes the differences between procedural and object oriented programming.
    - Procedural programming

        - A program is divided into a set of procedures (AKA. routines or sub-routines)
        - Procedures and data are separate
        - Top-down approach
    - Object Oriented Programming (OOP)
        - A program is divided into a set of objects
        - Methods (procedures) and data are bundled together to form an object
        - Bottom-up approach
- The benefits of using functions are:
    - Maintainability
    - Reusability
    - Simplicity
    - Cleaner variable namespace
    - Easier debugging
- The following describes the characteristics of the Python language (note, some features are not Python specific):
    - Python is an interpreted language.
    - Python is a dynamic language.
    - Python is case-sensitive.
    - In Python, a value must be given to create a variable.
- On indentation in Python:
    - Indentation is the whitespace characters added to the beginning of a line of code.
    - Python uses indentation to determine the scope of a statement.
    - Statements belonging to the same scope must have consistent indentation for the program to work properly.
    - In Python indentation is not just for code presentation, it’s part of the program logic.
- On Boolean expressions
    - There are six Relational operators: >, <, >=, <=, ==, !=
    - A relational operator combines two values (variables) to form a Boolean expression.
    - There are three Logical operators: and, or, not. There exists precedence of the three logical operators. The order of operations is “not” before “and” before “or”.
    - A logical operator combines two Boolean expressions to form a complex Boolean expression.
    - The Truth Table is used to evaluate the outcome of a complex Boolean expression (with two or more Boolean expressions combined).
    - A Boolean expression (or multiple Boolean expressions connected by logical operators) always evaluate to a Boolean value (True or False)
    - A Boolean expression should contain three components, with two operands and one relational operator between them.
- On iteration (loops):
    - A definite loop is one whose number of iterations is definite or known when entering the loop.
    - For an indefinite loop, its number of iterations is not known (or indefinite) when the loop starts.

    - An infinite loop (or endless loop) is a sequence of instructions in a computer program which loops endlessly, either due to the loop having no terminating condition, having one that can never be met. An infinite loop is a situation we normally should avoid when writing programs.
    - The continuation condition of a loop needs to eventually become false to stop the loop and avoid an infinite loop.
- On Python functions:
    - When you pass a value to invoke a function, this value is called an argument.
    - A function can be defined with no parameters. They are called parameter-less functions.
    - You can use the return statement to send a value to the caller of the function.
- On Python lists:
    - Integers are used to indicate the positions of list elements. They are called indices. The first element of a list has the index 0.
    - To retrieve an element of a list, all you need is the index of the element.
    - To retrieve an element of a list, all you need is the name of the list.
    - The elements of a list can be of different data type.