Python Programming Lab 2 Overview

# 1. Iterative Structures

We studied iterative functions in Lab 2, exploring essential concepts such as the **while loop** and **for in loop**. The while loop executes a block of code repeatedly as long as a specified condition remains true. It's particularly useful when the number of iterations isn't predetermined, allowing dynamic execution based on runtime conditions.

# 2. Loop Control Statements

During our studies, we also learn loop control statements that enhance the flexibility and control of iterative processes:

**The break statement** interrupts the loop execution immediately upon meeting a specific condition, allowing for premature termination based on predefined criteria. It's useful for escaping loops early when further iterations are unnecessary.

**The continue statement** skips the current iteration of a loop and proceeds directly to the next iteration. This feature proves beneficial for skipping over specific elements or conditions within the loop without exiting the loop entirely.

# 3. Python Functions

In Lab 2, we also explored the fundamentals of Python functions, which are pivotal for structuring modular and reusable code:

**Defining Functions**: Functions are defined using the def keyword followed by a function name and parameters enclosed in parentheses. The function body, containing executable statements, is indented to denote its scope.

**Function Parameters:** Parameters are variables within a function that hold values passed to the function when it is called. They facilitate dynamic behavior by accepting arguments either by position or by keyword, enhancing the flexibility of function invocation.

**Return Statement:** The return statement concludes the execution of a function and optionally returns a value or values to the calling code. It enables functions to produce output based on processed input or internal computations, promoting efficient data processing and retrieval

# 4. Python Classes and Objects

We also covered Python classes and objects, which are fundamental in object-oriented programming (OOP):

**Defining Classes:** Classes in Python serve as templates for creating objects. They group together data (attributes) and actions (methods) that describe what an object created from the class can do. To define a class, you use the keyword class, followed by the class's name and a colon. Inside the class, you define its attributes (variables) and methods (functions).

**Creating Objects:** Objects are instances of a class, which means they are specific examples created based on the blueprint provided by the class. You create an object by calling the class name followed by parentheses. If needed, you can pass arguments to initialize the object's attributes using a special method called \_\_init\_\_, which is known as the initializer or constructor method.

Conclusion:

In Lab 2 We learn about Basics of Programming in Python. That will help us about in advance programming