EXPERIMENT NO. 1

INTRODUCTION TO PYTHON PROGRAMMING

AIM

To familiarize python platform of programming

DESCRIPTION

Python is a high-level, versatile, and widely-used programming language known for its simplicity and readability. Created by Guido van Rossum and first released in 1991, Python has gained immense popularity in various fields, from web development and scientific research to data analysis, machine learning, and automation. Unlike many similar languages, its core language is very small and easy to master, while allowing the addition of modules to perform a virtually limitless variety of tasks. Python is a true object-oriented language, and is available on a wide variety of platforms.

RUNNING CODE IN INTERACTIVE SHELL

Python is an interpreted language, and you can run simple Python expressions and statements in an interactive programming environment called the shell. A shell window contains an opening message followed by the special symbol >>>, called a shell prompt. The cursor at the shell prompt waits for you to enter a Python command. When you enter an expression or statement, Python evaluates it and display its result, if there is one, followed by a new prompt.

Python programs can also be written and saved in a file with extension .py. To execute a script, type the file name along with the path at the prompt. The output will appear on the shell.

BASIC PROGRAMMING CONSTRUCTS IN PYTHON

- **Print function** produces text output on the console.

  Syntax:

  print ("***Message***")

  print (***Expression)***

- **Input function** is used for getting data from user. It accepts all user input as string and assigns it to the variable on left-hand side of the assignment operator (=).

Syntax: <variable_name>=input("Display Message")

- **Variables**

Variables are containers for storing data value. Variable declaration is implicit in Python, i.e., variables are automatically declared and defined when they are assigned a value the first time.

Syntax: <variable_name>= value

- **Python Indentation**

  Indentation refers to the spaces at the beginning of a code line. Python uses indentation to indicate a block of code

- **Comment**

  A comment starts with # (hash sign). Everything following the # till the end of that line is treated as a comment

- **Data Types**

  Data type identifies the type of data values a variable can hold and the operations that can be performed on that data

  - **Numeric Type**- int, float, complex
  - **Boolean Type**-bool
  - **None Type-** None, used to signify the absence of a value
  - **Mapping Type**-Dictionary
  - **Sequence Type**-A Python sequence is an ordered collection of items, where each item is indexed **Mapping Type** by an integer. The three types of sequence data types available in Python are string, list, and tuple

    String- String is a group of characters. These characters may be alphabets, digits or special characters including spaces. String values are enclosed either in single quotation marks or in double quotation marks

    List- List is a sequence of items separated by commas and the items are enclosed in square brackets [ ].

    Syntax: <list_name>= ["value1","value2" ...." valuen"]

    Tuple- Tuple is a sequence of items separated by commas and items are enclosed in parenthesis ( ).It is a collection which is ordered and unchangeable

    Syntax: <tuple_name>= ("value1","value2" ...." valuen")

Set- Set is an unordered collection of items separated by commas and the items are enclosed in curly brackets { }. Set cannot have duplicate entries. Once created, elements of a set cannot be changed.

Syntax: <set_name>= {‘“value1”,”value2” ….” valuen”}

Dictionary- Dictionary holds data items in key-value pairs. Items in a dictionary are enclosed in curly brackets { }. Dictionaries permit faster access to data. Every key is separated from its value using a colon (:) sign. The key: value pairs of a dictionary can be accessed using the key.

- **Conditional Statements**

  Conditional statements facilitate decision-making within a program and perform an action depending on whether a defined condition is true.

```
if<condition1>:
    <sequence of statements1>
elif<condition2>:
    <sequence of statements2>
else:
    <default sequence of statements>
```

- **Loops**
  - Loops simplify the process of repeating an action for a specific number of steps or until a condition is met. Python presents two types of loops when code needs to be repeated: for and while.
  - Syntax while loop

```
while(condition):
        Body of the loop
```

  - Syntax for loop

```
for <var> in range (<start>,<stop>):
        Body of the loop
```

- **Functions**
  - In Python a function is defined using the **def** keyword:

- Syntax

```
def function_name():
        Function body
```