LAB # 01

INTRODUCTION TO PYTHON , OPERATOR ,STRING AND FUNCTION , LOOP AND CONDITIONAL STATEMENTS

# OBJECTIVE

Familiarization with Python language using operator and string and using function, loop and conditional statement.

# THEORY

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is widely used for Artificial Intelligence, with packages for a number of applications including Machine Learning and NLP. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
* **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**Python Features:**

Python's features include:

* **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
* **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable:** Python provides a better structure and support for large programs than shell scripting.

**Modules in Python:**

* A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a file consisting of Python code. A module can define functions, classes and variables.
* In Python, modules are accessed by using the ’import’ statement. When you do this, you execute the code of the module, keeping the scopes of the definitions so that your current file(s) can make use of these.
* Many build-in modules of python, some of above

1. Math
2. Random
3. Fraction
4. Decimal
5. OS

**Operator in Python:**

* + Basic algebraic operations
    - Four arithmetic operations: a+b, a-b, a\*b, a/b
    - Exponentiation: a\*\*b
    - Other elementary functions are not part of standard Python, but included in packages like NumPy and SciPy
  + Comparison operators
    - Greater than, less than, etc.: a < b, a > b, a <= b, a >= b
    - Identity tests: a == b, a != b
  + Bitwise operators
    - Bitwise or: a | b
    - Bitwise exclusive or: a ^ b # Don't confuse this with exponentiation
    - Bitwise and: a & b
    - Shift a left or right by b bits: a << b, a >> b

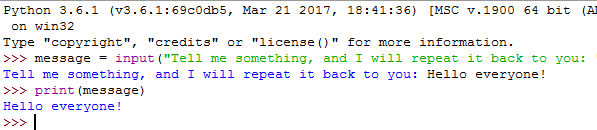
**Input( ) Function:**

The input() function pauses your program and waits for the user to enter some text. Once Python receives the user’s input, it stores it in a variable to make it convenient for you to work with.

The purpose of an input statement is to get some information from the user of a program and store it into a Variable.

* Syntax: <variable> = input (<prompt>)

**Example:**



**Strings:**

A *string* is simply a series of characters. Anything inside quotes is considered a string in Python, and you can use single or double quotes around your strings like this:

"This is a string." 'This is also a string.'

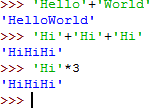
**Changing Case in a String with Methods**

One of the simplest tasks you can do with strings is change the case of the words in a string. Look at the following code, and try to determine what’s happening:

Text  Description automatically generated

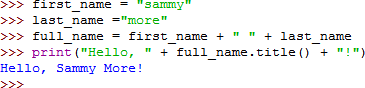
**Combining or Concatenating Strings:**

When applied to strings, the + operation is called concatenation. It produces a new string that is a copy of the two original strings pasted together end-to-end. Notice that concatenation doesn’t do anything clever like insert a space between the words. The Python interpreter has no way of knowing that you want a space; it does exactly what it is told.



Strings can be concatenated with the ‘+’ operator and repeated with ‘\*’

**Example:**



**Indexing of string:**

Python starts indexing at 0. A string s will have indexes running from 0 to len(s)-1 (where len(s) is the length of s) in integer quantities.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S** | **A** | **m** | **m** | **Y** |  | **S** | **h** | **A** | **r** | **k** | **!** |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |

**Example:**

* + s[i] : fetches the ith element in s
  + s[i:j] : fetches elements i (inclusive) through j (not inclusive)
  + s[:j] fetches all elements up to, but not including j
  + s[i:] fetches all elements from i onward (inclusive)
  + s[i:j:k] extracts every kth element starting with index i (inlcusive) and ending with index j (not inclusive)
  + Python also supports negative indexes. For example, s[-1] means extract the first element of s from the end (same as s[len(s)-1])

ss=”Sammy Shark! “ print(ss[2])

print(ss[6:11])

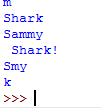
print(ss[:5])

print(ss[5:])

print(ss[0:5:2])

print(ss[-2])

**Output:**



**Functions:**

Basically, we can divide functions into the following two types:

1. [Built-in functions](https://www.programiz.com/python-programming/built-in-function) - Functions that are built into Python.
2. [User-defined functions](https://www.programiz.com/python-programming/user-defined-function) - Functions defined by the users themselves.

A function is a block of organized, reusable code that is used to perform a single, related action. Functions provide better modularity for your application and a high degree of code reusing.

As you already know, Python gives you many built-in functions like print(), etc. but you can also create your own functions. These functions are called user-defined functions*.*

Defining a Function:

You can define functions to provide the required functionality. Here are simple rules to define a function in Python.

* + Function blocks begin with the keyword **def** followed by the function name and parentheses ( ( ) ).
  + Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.
  + The code block within every function starts with a colon (:) and is indented.

**Syntax:**

def functioname ( PARAMETERS ): Statement

**Example:**

Text  Description automatically generated

**Arguments:**

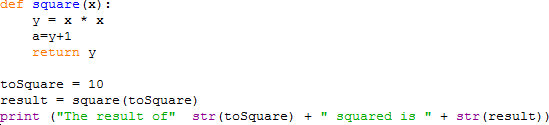
An argument is a piece of information that is passed from a function call to a function. When we call the function, we place the value we want the function to work with in parentheses.

You can call a function by using the following types of formal arguments:

* + Required arguments
  + Keyword arguments
  + Default arguments

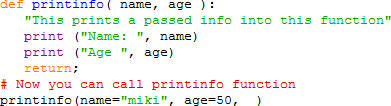
**Required Arguments:**

Required arguments are the arguments passed to a function in correct positional order. Here, the number of arguments in the function call should match exactly with the function definition.



**Keyword Arguments:**

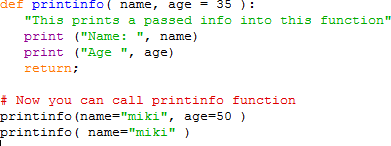
Keyword arguments are related to the function calls. When you use keyword arguments in a function call, the caller identifies the arguments by the parameter name.

This allows you to skip arguments or place them out of order because the Python interpreter is able to use the keywords provided to match the values with parameters.

When the above code is executed, it produces the following result –



Default Arguments:

A default argument is an argument that assumes a default value if a value is not provided in the function call for that argument. The following example gives an idea on default arguments, it prints default age if it is not passed –

When the above code is executed, it produces the following result –



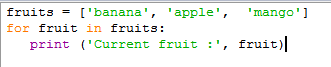
# For loop Statement:

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable. It has the ability to iterate over the items of any sequence, such as a list or a string.

**Syntax:**

for iterating variable in sequence: Statements(s)

**Example:**



**Output:**

Text  Description automatically generated

**While Loop:**

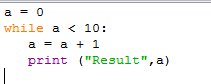
The for loop takes a collection of items and executes a block of code once for each item in the collection. In contrast, while loop statement in Python programming language repeatedly executes a target statement as long as a given condition is true.

**Syntax:**

The syntax of a while loop in Python programming language is − while expression:

statement(s)

**Example:**



**Loop Control Statements:**

Loop control statements change execution from its normal sequence. You might face a situation in which you need to exit a loop completely when an external condition is triggered or there may also be a situation when you want to skip a part of the loop and start next execution.

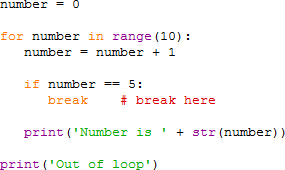
Python provides break and continue statements to handle such situations and to have good control on your loop.

**The break Statement:**

The break statement in Python terminates the current loop and resumes execution at the next statement, just like the traditional break found in C.

The most common use for break is when some external condition is triggered requiring a hasty exit from a loop. The break statement can be used in both while and for loops.

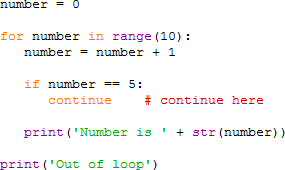
**Example**



**The continue Statement:**

The continue statement in Python returns the control to the beginning of the while loop. The continue statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.

The continue statement can be used in both while and for loops.

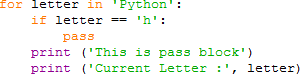


**The pass Statement:**

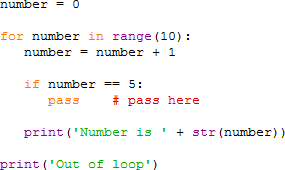
The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.

The pass statement is a null operation; nothing happens when it executes.

**Example 1:**



**Example 2:**

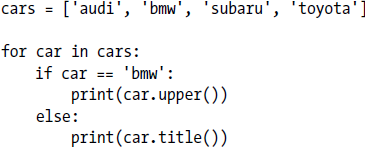


**CONDITIONAL STATEMENTS:**

***If*-*Else* Statement:**

An *if-else* block is similar to a simple if statement, but the else statement allows you to define an action or set of actions that are executed when the conditional test fails.\

**Example:**



**The *if-elif-else* Statement:**

The elif statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE.

**Syntax:**

if expression1: statement(s)

elif expression2:

statement(s)

elif expression3: statement(s)

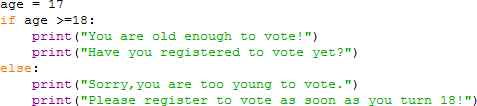
else:

statement(s)

**Example: (Simple if statements)**

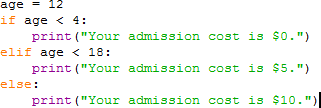
Logo  Description automatically generated with medium confidence

**Example: (if/else statements)**



**Example (*if-elif-else statements*)**

For example, consider an amusement park that charges different rates for different age groups:

* Admission for anyone under age 4 is free.
* Admission for anyone between the ages of 4 and 18 is $5.
* Admission for anyone age 18 or older is $10.

You can use as many elif blocks in your code as you like.

# Lab Task:

1. Write a script that take user input for a number then adds 3 to that number. Then multiplies the result by 2, subtract 4, then again adds 3, then print the result.
2. Write a script that takes input as Celsius and then convert Celsius to Fahrenheit. (hint: Fahrenheit = (Celsius \* 1.8) + 32)
3. Write a script that takes input as radius then calculate area of circle. (hint: A = πr²)
4. Write a Python script that asks users for their favourite color. Create the following output (assuming blue is the chosen color) (hint: use ‘+’ and ‘\*’)

blueblueblueblueblueblueblueblueblueblue blue blue

blueblueblueblueblueblueblueblueblueblue

1. Store a person’s name, and include some whitespace characters at the beginning and end of the name. Make sure you use each character combination, "\t" and "\n", at least once. Print the name once, so the whitespace around the name is displayed. Then print the name using each of the three stripping functions, lstrip( ), rstrip(), and strip().

6 .Write a python script that take a user input and to create the multiplication table (from 1 to 10) of that number.

1. Write a function called describe\_city( ) that accepts the name of a city and its country. The function should print a simple sentence, such as **Islamabad is in Pakistan**. Give the parameter for the country a default value. Call your function for three different cities, at least one of which is not in the default country.
2. Write a function called absolute\_num() that accepts one parameter, num. The function should return only positive value, and apply condition on it. This function returns the absolute value of the entered number.
3. Write Python Program to check whether an alphabet is a vowel or consonant? (use if, else conditional statement).

10 .Write a Python program to check whether a number is prime or not? (use if, else conditional statement).

11.Write a Python program that prints all the numbers from 0 to 6 except 3 and 6. (Hint: Use 'continue' statement).

12 .Write a Python program to construct the following pattern. (using nested loop)

**1**

**2 2**

**3 3 3**

**4 4 4 4**

**5 5 5 5 5**

1. Stages of Life: Write an if-elif-else chain that determines a person’s stage of life. Set a value for the variable age, and then:
   * If the person is less than 2 years old, print a message that the person is a baby.
   * If the person is at least 4 years old but less than 13, print a message that the person is a kid.
   * If the person is at least 13 years old but less than 20, print a message that the person is a teenager.
   * If the person is at least 20 years old but less than 65, print a message that the person is an adult.
   * If the person is age 65 or older, print a message that the person is an elder.