National University of Computer and Emerging Sciences

Lab Manual 01

Table of Contents

Department of Computer Science FAST-NU,
Lahore, Pakistan

• Objectives

After performing this lab, students shall be able to understand:

- Python data types.
- Python operators (math, comparison, boolean)
- Python condition and loops
- Python functions

• Task Distribution

Total Time 170 Minutes Python data types 15 Minutes
Python operators 10 Minutes
Python if-else conditions 10 Minutes
Python loops 15 Minutes
Python functions 20 Minutes
Exercise 90 Minutes

Online Submission 10 Minutes

• Online Python Interpreter

We will be working on Google Colab to run Python programs.

Colaboratory, or "Colab" for short, allows you to write and execute Python in your browser without any configuration. To do that we need to create a Colab notebook.

Colab notebooks allow you to combine executable code and rich text in a single document, along with images, HTML, LaTeX and more. When you create your own Colab notebooks, they are stored in your Google Drive account. You can easily share your Colab notebooks with co-workers or friends, allowing them to

comment on your notebooks or even edit them.

Colab notebooks are Jupyter notebooks that are hosted by Colab. The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. Learn more about Jupyter here.

Follow the instructions given below:

- Visit the following link: https://colab.research.google.com/ •
  Sign in using your nu email id.
- You will be directed to the 'Welcome to Colaboratory' page.
- Go to File->New Notebook.
- A new notebook is created by the name 'Untitled.ipynb'.
- Rename the notebook to your roll number.
- Write your first Python program by typing the following statement in the first cell `print('Hello World')`
- Execute this cell by clicking the play button on the left of the cell or by pressing Ctrl+Enter.
- You will notice that the notebook will connect to a runtime. RAM and Disk resources are allocated. (Refer to the screenshot below)

For offline usage, PyCharm IDE is recommended. Installation details for PyCharm will be shared later.

• Data Types

The following section describes the standard types that are built into the Python interpreter. These data types are divided into different categories like numeric, sequences, mapping etc. Typecasting is also discussed below.

• Built-in Types

The following chart summarizes the standard data types that are built into the Python interpreter.

Sr#	Categories	Data Type Examples
		int -2, -1, 0, 1, 2, 3, 4, 5, int(20) float -1.25, -1.0, --0.5, 0.0, 0.5, 1.0, 1.25,

1	Numeric Types	float(20.5)
2		complex 1j, complex(1j)
3		

4	Text Sequence Type	str 'a', 'Hello!', str("Hello World")
5	Boolean Type	bool True, False, bool(5)
6	Sequence Types	list ["apple", "banana", "cherry"], list(("apple", "banana", "cherry"))
7		tuple ("apple", "banana", "cherry"), tuple(("apple", "banana", "cherry"))
8		range range(6)
9	Mapping Type	dict {"name" : "John", "age" : 36}, dict(name="John", age=36)
10	Set Types	set {"apple", "banana", "cherry"}, set(("apple", "banana", "cherry"))
11		frozenset frozenset({"apple", "banana", "cherry"})
12	Binary Sequence Types	bytes b"Hello", bytes(5)
13		bytearray bytearray(5)

14 memoryview memoryview(bytes(5))

Python has no command for declaring a variable for any datatype. A variable is created the moment you first assign a value to it. Variable names are case-sensitive. Just like in other languages, Python allows you to assign values to multiple variables in one line.

- Typecasting

The process of explicitly converting the value of one data type (int, str, float, etc.) to another data type is called type casting. In Type Casting, loss of data may occur as we enforce the object to a specific data type.

Notice the`type()` function used in the above example. Find out what it does. Execute the given example in the Jupyter notebook to observe the result of type casting.

• Operators

This section contains the details of different Python operators i.e. Math operators, comparison operators and Boolean operators.

• 1 Math Operators

From Highest to Lowest precedence:

Operators **	Operation Example Exponent `2 ** 3 = 8`
%	Modulus/Remainder `22 % 8 = 6`

Operators //	Operation on Example Integer division `22 // 8 = 2`
/	Division `22 / 8 = 2.75`
*	Multiplication `3 * 3 = 9`
-	Subtraction `5 - 2 = 3`
+	Addition `2 + 2 = 4`

• Comparison Operators

Operator ==	Meaning Equal to
!=	Not equal to

<	Less than
>	Greater Than
<=	Less than or Equal to
>=	Greater than or Equal to

- Boolean Operators

There are three Boolean operators: and,or and not

The And Operator'sTruth Table:

Expression Evaluates to True and True True
True and False False
False and True False
False and False False

The OR Operator'sTruth Table:

Expression Evaluates to True or True True
True or False True
False or True True
False or False False

The NOT Operator'sTruth Table:

Expression	Evaluates to
not True	False
not False	True

• If-else Conditions

Python supports conditional statements i.e.`if`,`elif`,`else`. Comparison operators and Boolean operators written in the previous section can be used in if-elif-else statements. Python uses indentation instead of curly-brackets to define the scope in the code.

• if Statement Example:

• if-else Statement Example:

• if-elif-else Statement Example:

Python conditional statements only require *if* statements to execute. Both`elif` and`else` are optional and used as per requirement.

• Loops

Python has two types of loops i.e.`while`,`for`. Use Jupyter notebook to execute all code snippets given in the examples below to observe their results.

• While Loop Example with break Statement

With the while loop we can execute a set of statements as long as a condition is true or the loop execution reaches a break statement.

`input()` in the above example is a built-in Python function which is discussed in the functions section below.

• While Loop Example with continue Statement

When the program reaches continue statement, the program execution immediately jumps back to the start of the loop.

• for Loop Example with range()

• for Loop Example with range() arguments

The`range()` function can also be called with three arguments. The first two arguments will be the start and stop values, and the third will be the step argument. The step is the amount that the variable is increased by after each iteration.

• Functions

This section contains the details of Python user defined or custom functions along with a few examples of Python built-in functions.

• Custom Functions

Programmers can define their own functions in Python. Functions can contain all types of Python statements like variables, conditions and loops etc.

• Simple Function Example

A function in Python starts with a def keyword followed by the function name with round brackets. Function parameters can be passed depending on the requirement.

• Function Example with Return Statement

A return statement consists of the following:

- The return keyword.
- The value or expression that the function should return.

• Built-in Functions

The Python interpreter has a number of functions built into it that are always available.

We have already covered a few built-in functions in the datatypes section above. Refer to this link for the complete list of Python built-in functions.

• Built-in Function Examples

Execute the code given below in your Jupyter notebook to find the results of built-in functions.

• Exercise (25 Marks)

Complete all tasks given below

• Reverse String (5 Marks)

Write a function that reverses a string. The input string is given as an array of characters. You can use a Python list to create the array of characters. Do not allocate extra space for another array, you must do this reversal by modifying the input array in- place with O(1) extra memory.

Example 1:
Input: ["h","e","l","l","o"]
Output: ["o","l","l","e","h"]

Example 2:
Input: ["H","a","n","n","a","h"]
Output: ["h","a","n","n","a","H"]

• Valid Palindrome (10 Marks)

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases. Refer to this link to learn about Python string functions which might come handy in this task.

Example 1:
Input: s = "A man, a plan, a canal: Panama"
Output: true
Explanation: "amanaplanacanalpanama" is a palindrome.

Example 2
Input: s = "race a car"
Output: false
Explanation: "raceacar" is not a palindrome.

• UniqueList (10 Marks)

Given a list of integers with duplicates, return a new unique list removing duplicates.

Example 1:
Input: x = [1,2,3,4,5,6,6,1,5]
Output: [1,2,3,4,5,6]

Example 2:

Input: x = [1,4,1,1,5]
Output: [1,4,5]

• Submission Instructions

Always read the submission instructions carefully.

- Rename your Jupyter notebook to your roll number and download the notebook as .ipynb extension.
- To download the required file, go to File->Download .ipynb • Only submit the .ipynb file. DO NOT zip or rar your submission file • Submit this file on Google Classroom under the relevant assignment. • Late submissions will not be accepted.