University of Engineering and Technology Taxila
Computer Engineering Department

Data Structure and Algorithm

Lab Manual # 01

Lab Instructor: Engr. Shahryar Khan

Lab Title: Python Basics and Foundations

Lab Overview

This lab focuses on introducing Python to beginners, providing a strong foundation in the language's core constructs. It is designed to cover essential topics like variables, data types, strings, Boolean logic, control structures, and Python collections. The lab consists of guided tasks to help students gain hands-on experience, along with exercises to test their understanding. At the end of this lab, students will be equipped with the basic knowledge required to advance into more complex programming concepts.

Lab Objectives

1. Familiarize students with Python syntax and programming basics.
2. Understand and work with variables, data types, and strings.
3. Learn the concepts of Boolean logic, operators, and control structures.
4. Explore Python collections such as lists, tuples, sets, and dictionaries.
5. Introduce the use of functions and arrays.

Lab Requirements

- Install Python: Download and install Python 3.10+ from python.org.
- Install VSCode: Download and install Visual Studio Code from code.visualstudio.com.
- Add Python Extension in VSCode: Open VSCode → Go to Extensions (Ctrl+Shift+X) → Search for "Python" → Install.
- Install Code Runner Extension: In VSCode Extensions, search for "Code Runner" and install it to run Python programs easily.
- Verify Python Installation: Open a terminal or command prompt and type python --version to confirm Python is installed correctly.
- Set Up Workspace: Create a folder for lab exercises (e.g., Python_Labs) and open it in VSCode as your workspace.
- Reference Official Documentation: Use the Python official documentation (https://docs.python.org/3/) and sample programs to enhance learning.

Guided Tasks

Task 1: Python Variables

Objective: Learn how to define and use variables in Python.

Steps:

1. Open a Python file in VSCode (e.g., lab1_task1.py).
2. Write the following program

```
# Define variables
name = "Alice"
age = 25
is_student = True
# Print variables
print("Name:", name)
print("Age:", age)
print("Is Student:", is_student)
```

What You Learn: How to declare variables and store data in Python.

Expected Output

```
Name: Alice
Age: 25
Is Student: True
```

Task 2: Python Data Types

Objective: Understand different Python data types and their use.

Create a file *task2.py* in VSCode and type the following program.

```
# Data types
integer_value = 10
```

```
float_value = 20.5
string_value = "Hello Python"
boolean_value = True
# Print data types
print(type(integer_value))
print(type(float_value))
print(type(string_value))
print(type(boolean_value))
```

What You Learn: The common data types in Python and how to check them using `type()`.

Expected Output:

```
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
```

Task 3: Python Numbers and Type Conversions

Objective: Learn how to perform operations on numbers and convert data types.

Steps:

Write the following code:

```
# Numbers and conversions
num = 10
decimal = 5.7
text = "123"
# Arithmetic operations
print("Sum:", num + decimal)
print("Product:", num * decimal)
```

```
# Type conversions

print("Integer to Float:", float(num))

print("String to Integer:", int(text))
```

What You Learn: Arithmetic operations and type conversion methods like `int()` and `float()`.

Expected Output:

```
Sum: 15.7

Product: 57.0

Integer to Float: 10.0

String to Integer: 123
```

Task 4: Python Strings

Objective: Explore string slicing, concatenation, and commonly used methods.

Steps

Write the following program

```
text = "Python Programming"

# Slicing

print("First 6 characters:", text[:6])

print("Last 6 characters:", text[-6:])

# Concatenation

greeting = "Hello, "

print("Greeting:", greeting + text)

# String methods

print("Uppercase:", text.upper())

print("Replace 'Python' with 'Java':", text.replace("Python", "Java"))
```

What You Learn: String manipulation and using methods like `upper()` and `replace()`.

Expected Output:

```
First 6 characters: Python
Last 6 characters: mming
Greeting: Hello, Python Programming
Uppercase: PYTHON PROGRAMMING
Replace 'Python' with 'Java': Java Programming
```

Task 5: Python Boolean

Objective: Understand how Boolean values work and how to use them.

Steps:

Write the following code

```
# Boolean values
a = 10
b = 20
print("Is a equal to b?", a == b)
print("Is a less than b?", a < b)
print("Logical AND:", a < b and b > 15)
print("Logical OR:", a > b or b > 15)
```

What You Learn: Boolean operations and logical expressions.

Expected Output:

```
Is a equal to b? False
Is a less than b? True
Logical AND: True
Logical OR: True
```

Task 6: Python Operators

Objective: Learn and use Python's various operators.

Steps

Write the following program

```
x = 15
y = 4
# Arithmetic operators
print("Addition:", x + y)
print("Division:", x / y)
# Comparison operators
print("Is x greater than y?", x > y)
# Logical operators
print("Logical AND:", x > 10 and y < 5)
```

What You Learn: Arithmetic, comparison, and logical operators.
Expected Output:

```
Addition: 19
Division: 3.75
Is x greater than y? True
Logical AND: True
```

Task 7: Python Lists

Objective: Learn to work with Python lists and perform operations.
Steps

Write the following code

```
fruits = ["apple", "banana", "cherry"]

# Accessing list items
```

```
print("First fruit:", fruits[0])

# Modifying list

fruits.append("orange")

print("List after adding orange:", fruits)

# Looping through list

for fruit in fruits:

  print("Fruit:", fruit)
```

What You Learn: Accessing, modifying, and looping through lists.

Expected Output

```
First fruit: apple

List after adding orange: ['apple', 'banana', 'cherry', 'orange']

Fruit: apple

Fruit: banana

Fruit: cherry

Fruit: orange
```

Task 8: Python Tuples

Objective: Explore tuple properties and operations.

Steps

Write the following code

```
# Tuples

colors = ("red", "green", "blue")

print("First color:", colors[0])

# Unpacking tuple

(a, b, c) = colors

print("Unpacked values:", a, b, c)
```

What You Learn: Accessing and unpacking tuple elements.

Expected Output

```
First color: red

Unpacked values: red green blue
```

Task 9: Python Sets

Objective: Use sets to handle unique items and perform operations.
Steps:

Write the following code

```
# Sets

numbers = {1, 2, 3, 4}

numbers.add(5)

numbers.remove(2)

print("Updated set:", numbers)

# Set operations

even_numbers = {2, 4, 6}

print("Union:", numbers.union(even_numbers))

print("Intersection:", numbers.intersection(even_numbers))
```

What You Learn: Adding/removing items and performing set operations.

Expected Output

```
Updated set: {1, 3, 4, 5}

Union: {1, 2, 3, 4, 5, 6}

Intersection: {4}
```

Task 10: Python Dictionaries

Objective: Work with key-value pairs in Python dictionaries.
Steps

Write the following code

```
# Dictionaries

student = {"name": "Alice", "age": 21, "grade": "A"}

student["age"] = 22

print("Updated dictionary:", student)

# Loop through dictionary

for key, value in student.items():

   print(key, ":", value)
```

What You Learn: Accessing, updating, and iterating through dictionaries.
Expected Output

```
Updated dictionary: {'name': 'Alice', 'age': 22, 'grade': 'A'}

name : Alice

age : 22

grade : A
```

Task 11: Python If-Else

Objective: Understand decision-making in Python using if-else conditions.

Steps:

Write the following code

```
# Input from user

number = int(input("Enter a number: "))
```

```
# Check if the number is positive, negative, or zero
if number > 0:
   print("The number is positive.")
elif number < 0:
   print("The number is negative.")
else:
   print("The number is zero.")
```

What You Learn: Implementing conditional logic with if-else and handling user input.

Expected Output

For input = 5

```
Enter a number: 5
The number is positive.
```

Task 12: Python While Loop

Objective: **Understand how to use** a while loop to perform repeated actions.

Steps

Write the following code

```
# Print numbers from 1 to 5 using a while loop
i = 1
while i <= 5:
   print("Number:", i)
   i += 1
```

What You Learn: Using `while` loops for repetitive tasks with a known exit condition.

Expected Output

```
Number:  1

Number:  2

Number:  3

Number:  4

Number:  5
```

Task 13: Python For Loop

Objective: Explore Python's for loop to iterate over sequences.

Steps

1- Write the following code

```
# Print each character of a string

for char in "Python":

  print("Character:", char)
```

2- Write the following program and observe the output

```
for num in range(1, 6):

  print("Square of", num, "is", num ** 2)
```

Task 14: Python Functions

Objective: Learn how to define and call functions in Python.

Steps:

1- Write the following program to define a simple function:

```
# Define a function to greet the user

def greet(name):

  print(f"Hello, {name}! Welcome to Python programming.")
```

```
# Call the function

greet("Alice")

greet("Bob")
```

2- Write another program to calculate the factorial of a number:

```
def factorial(n):

  if n == 0 or n == 1:

    return 1

  else:

    return n * factorial(n - 1)

# Input from user

num = int(input("Enter a number: "))

print("Factorial of", num, "is", factorial(num))
```

What You Learn: Writing reusable functions and using recursion.

Task 15: Python Lambda Functions

Objective: Explore anonymous functions using the lambda keyword.

Steps

1- Write a program to demonstrate a simple lambda function

```
# Lambda to find square of a number

square = lambda x: x ** 2

print("Square of 5:", square(5))
```

2- Use a lambda function with `filter()`

```
# Filter even numbers from a list

numbers = [1, 2, 3, 4, 5, 6, 7, 8]
```

```
even_numbers = list(filter(lambda x: x % 2 == 0, numbers))

print("Even numbers:", even_numbers)
```

What You Learn: Defining and using lambda functions with built-in methods like `filter()`.

-------------------------------------- xxxxx ------- xxxx ---

Exercise Questions:

1. **Check Positive or Negative**
   Write a program that takes a number as input and checks whether it is positive, negative, or zero.
2. **Calculate the Sum of Two Numbers**
   Write a program to take two numbers as input and print their sum.
3. **Find the Maximum of Three Numbers**
   Write a program to input three numbers and print the largest one.
4. **Reverse a String**
   Write a program to reverse a string input by the user.
5. **Check for Even or Odd Number**
   Write a program to check if a number is even or odd.
6. **Print Multiplication Table**
   Write a program to print the multiplication table for a given number (from 1 to 10).
7. **Convert Celsius to Fahrenheit**

   Write a program to convert a temperature from Celsius to Fahrenheit using the formula:

   $$\text{Fahrenheit} = (\text{Celsius} \times 9 / 5) + 32$$

8. **Count the Vowels in a String**
   Write a program to count the number of vowels in a string the user provides.
9. **Calculate the Factorial of a Number**
   Write a program to calculate the factorial of a number using a loop.
10. **Find Prime Numbers in a Range**
    Write a program to print all prime numbers between 1 and 50.

Real-Life Scenario Questions (5 Questions)

1- Electricity Bill Calculation

Write a program to calculate the electricity bill based on the following tariff:

- Up to 100 units: Rs. 30/unit
- 101 to 300 units: Rs. 40/unit
- Above 300 units: Rs. 60/unit
- Meter Rent: 1500
- Additional Tax: 200

Ask the user to input the units consumed and print the total bill.

2- GPA Calculation

Write a program to calculate the GPA of a student.
Input the grades for 5 courses (on a scale of 4.0) and calculate the average GPA.

3- Monthly Budget Planner

Write a program to help a user plan their monthly budget.
Input the monthly income and expenses for categories like rent, food, transportation, and savings. Calculate the remaining balance or deficit.

4- Loan EMI Calculator

Write a program to calculate the EMI (Equated Monthly Installment) for a loan based on the following formula:

$$EMI = \frac{P \times R \times (1+R)^N}{(1+R)^N - 1}$$

Where:

- P = Principal loan amount
- R = Monthly interest rate (Annual interest rate / 12 / 100)
- N = Number of monthly installments