# PYTHON

## *Assignment 1*

| | |
|---|---|
| Name:- Ankit Kumar Shukla | Class:- MCA(A) |
| Git-hub link:-https://github.com/shuklaankit17/python-assignment-1.git | |

## PART 1: OPERATORS:

**Exercise 1: Arithmetic Operators**

Write a Python program to perform arithmetic operations (addition, subtraction, multiplication,
division, modulus, exponentiation, and floor division) on two user-input numbers.

**Code :**

```
# Get input from the user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

# Perform arithmetic operations
addition = num1 + num2
subtraction = num1 - num2
multiplication = num1 * num2
division = num1 / num2
modulus = num1 % num2
exponentiation = num1 ** num2
floor_division = num1 // num2
```

```
# Print the results
print("Addition:", addition)
print("Subtraction:", subtraction)
print("Multiplication:", multiplication)
print("Division:", division)
print("Modulus:", modulus)
print("Exponentiation:", exponentiation)
print("Floor Division:", floor_division)
```

**output:**

```
Enter the first number: 10
Enter the second number: 20
Addition: 30.0
Subtraction: -10.0
Multiplication: 200.0
Division: 0.5
Modulus: 10.0
Exponentiation: 1e+20
Floor Division: 0.0
```

**Approach:**

The code takes two floating-point numbers from the user,
performs a series of arithmetic operations
(addition, subtraction, multiplication, division, modulus,
exponentiation, and floor division), and
prints the results of each operation.

**Exercise 2: Comparison Operators**

Check if the first number is greater than, equal to, or less than or equal to the second number.

**Code:**

```
# Get input from the user
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))

# Check if the first number is greater than the second
greater_than = num1 > num2
print("Is the first number greater than the second?",
greater_than)

# Check if the first number is equal to the second
equal_to = num1 == num2
print("Is the first number equal to the second?", equal_to)

# Check if the first number is less than or equal to the second
less_than_or_equal_to = num1 <= num2
print("Is the first number less than or equal to the second?",
less_than_or_equal_to)2
```

**Output:**

```
Enter the first number: 10
Enter the second number: 0
Is the first number greater than the second? True
Is the first number equal to the second? False
Is the first number less than or equal to the second? False
```

**Approach:**

The code takes two floating-point numbers from the user and checks three conditions: whether the

first number is greater than, equal to, or less than or equal to the second number. It then prints the
results of these comparisons.

**Exercise 3: Logical Operators**

Take three boolean values and combine them using and, or, and not operators.

**Code:**

```
# Get input from the user
bool1 = input("Enter the first boolean value (True or False):
").lower() == "true"
bool2 = input("Enter the second boolean value (True or False):
").lower() == "true"
bool3 = input("Enter the third boolean value (True or False):
").lower() == "true"

# Use logical operators to combine the boolean values
and_result = bool1 and bool2 and bool3
or_result = bool1 or bool2 or bool3
not_result = not bool1

# Print the results
print("Result of AND operation:", and_result)
print("Result of OR operation:", or_result)
print("Result of NOT operation on the first value:", not_result)
```

**output:**

```
Enter the first boolean value (True or False): 10
Enter the second boolean value (True or False): 20
Enter the third boolean value (True or False): 30
```

```
Result of AND operation: False
Result of OR operation: False
Result of NOT operation on the first value: True
```

**Approach:**

The code prompts the user for three boolean values, processes them using logical operations (AND,
OR, NOT), and prints the results of these operations. The use of .lower() allows for case-insensitive
input, ensuring that both "True" and "true" are interpreted correctly.

**Part 2: Strings**

**Exercise 4: String Manipulation**

Perform various operations on a string, such as finding its length, reversing it, and changing the case.

**Code:**

```
# Get input from the user
user_string = input("Enter a string: ")

# Calculate the length of the string
string_length = len(user_string)

# Get the first and last characters
first_char = user_string[0]
last_char = user_string[-1]

# Reverse the string
reversed_string = user_string[::-1]
```

```
# Convert the string to uppercase and lowercase
uppercase_string = user_string.upper()
lowercase_string = user_string.lower()

# Print the results
print("Length of the string:", string_length)
print("First character:", first_char)
print("Last character:", last_char)
print("Reversed string:", reversed_string)
print("Uppercase string:", uppercase_string)
print("Lowercase string:", lowercase_string)
```

**output:**

```
Enter a string: ankit shukla
Length of the string: 12
First character: a
Last character: a
Reversed string: alkuhs tikna
Uppercase string: ANKIT SHUKLA
Lowercase string: ankit shukla
```

**Approach:**

The code takes a string input from the user and performs several operations: it calculates the length
of the string, retrieves the first and last characters, reverses the string, and converts it to both
uppercase and lowercase. Finally, it prints all these results for the user.

**Exercise 5: String Formatting**

Take the user's name and age and format it into a sentence.

**Code:**

```
# Get input from the user
name = input("Enter your name: ")
age = input("Enter your age: ")

# Display the formatted message
print(f"Hello {name}, you are {age} years old.")
```

**Output:**

```
Enter your name: ankit shukla
Enter your age: 21
Hello ankit shukla, you are 21 years old.
```

**Approach:**

The code prompts the user for their name and age, then prints a personalized message that includes
both pieces of information. The use of an f-string makes it easy to format the output in a readable
way.

**Exercise 6: Substring Search**

Find whether a word exists in a sentence and, if it does, print its index.

**Code:**

```
# Get input from the user
sentence = input("Enter a sentence: ")
word_to_search = input("Enter a word to search for: ")

# Check if the word exists in the sentence
if word_to_search in sentence:
    # Find the index of the word
    index = sentence.find(word_to_search)
    print(f"The word '{word_to_search}' exists in the sentence at index {index}.")
else:
    print(f"The word '{word_to_search}' does not exist in the sentence.")
```

**Output:**

```
Enter a sentence: alliance university is good among all the universities in india because i am studying in it
Enter a word to search for: india
The word 'india' exists in the sentence at index 58.
```

**Approach:**

The code takes a sentence and a word as input, checks if the word exists in the sentence, finds its
index if it does, and provides feedback to the user based on whether the word was found or not.

## Part 3: Lists

**Exercise 7: List Operations**

Create a list of numbers, then find the sum, largest, and smallest values.

**Code:**

```
# Create an empty list to store the numbers
numbers = []

# Get input from the user for 5 numbers
for i in range(5):
    number = float(input(f"Enter number {i+1}: "))
    numbers.append(number)
```

```
# Calculate the sum of the numbers
sum_of_numbers = sum(numbers)

# Find the largest and smallest numbers
largest_number = max(numbers)
smallest_number = min(numbers)

# Print the results
print("List of numbers:", numbers)
print("Sum of the numbers:", sum_of_numbers)
print("Largest number:", largest_number)
print("Smallest number:", smallest_number)
```

**Output:**

```
Enter number 1: 5
Enter number 2: 6
Enter number 3: 7
Enter number 4: 8
Enter number 5: 9
List of numbers: [5.0, 6.0, 7.0, 8.0, 9.0]
Sum of the numbers: 35.0
Largest number: 9.0
Smallest number: 5.0
```

**Approach:**

This code collects five floating-point numbers from the user, stores them in a list, calculates their
sum, and identifies the largest and smallest numbers among them. It then outputs these results to

the user.

**Exercise 8: List Manipulation**

Add and remove elements from a list of fruits.

**Code:**

```
# Take input from the user
fruits = input("Enter your favorite fruits separated by commas:
").split(',')

# Strip any extra whitespace
fruits = [fruit.strip() for fruit in fruits]

# Add one more fruit
new_fruit = input("Enter one more fruit to add: ")
fruits.append(new_fruit.strip())

# Remove the second fruit (if it exists)
if len(fruits) > 1:
   fruits.pop(1)

# Print the updated list
print("Updated list of fruits:")
print(fruits)
```

**Output:**

```
Enter your favorite fruits separated by commas:
mango,banana,litchi
Enter one more fruit to add: apple
Updated list of fruits:
['mango', 'litchi', 'apple']
```

**Approach:**
This code collects a list of favorite fruits from the user, processes the input to clean it up, adds
another fruit, potentially removes the second fruit, and then displays the updated list of fruits.

**Exercise 9: Sorting a List**

Sort a list of numbers in both ascending and descending order.

**Code:**
```
# Get input from the user
numbers = []
for i in range(5):
    number = float(input(f"Enter number {i+1}: "))
    numbers.append(number)

# Sort the list in ascending order
numbers_ascending = sorted(numbers)

# Sort the list in descending order
numbers_descending = sorted(numbers, reverse=True)

# Print the results
print("List in ascending order:", numbers_ascending)
print("List in descending order:", numbers_descending)
```
**Output:**
```
Enter number 1: 5
Enter number 2: 10
Enter number 3: 15
Enter number 4: 20
Enter number 5: 25
List in ascending order: [5.0, 10.0, 15.0, 20.0, 25.0]
List in descending order: [25.0, 20.0, 15.0, 10.0, 5.0]
```

**Approach:**

The code collects 5 numbers from the user and stores them in a list.
It then sorts the list twice: once in ascending order and once in descending order, displaying both
results

**Exercise 10: List Slicing**

Slice a list to retrieve specific subsets of elements.

**Code:**

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# Print the first 5 elements
print("First 5 elements:", numbers[:5])

# Print the last 5 elements
print("Last 5 elements:", numbers[-5:])

# Print the elements from index 2 to index 7
print("Elements from index 2 to index 7:", numbers[2:8])
```

**Output:**

```
First 5 elements: [1, 2, 3, 4, 5]
Last 5 elements: [6, 7, 8, 9, 10]
Elements from index 2 to index 7: [3, 4, 5, 6, 7, 8]
```

**Approach:**

The code initializes a list of numbers from 1 to 10.

It uses slicing to print:

• The first 5 elements.

• The last 5 elements.

• Elements from index 2 to index 7

**Exercise 11: Nested List**

Store and process a nested list containing the names and scores of three students, and calculate their
average scores.

**Code:**

```
student_data = []

for i in range(3):
  student_name = input(f"Enter the name of student {i + 1}: ")
  scores = []
  for j in range(3):
    score = float(input(f"Enter score for subject {j + 1} for
{student_name}: "))
    scores.append(score)
  student_data.append([student_name, scores])

for student in student_data:
  name = student[0]
  scores = student[1]
  average_score = sum(scores) / len(scores)
  print(f"{name}: Average Score = {average_score}")
```

**Output:**

```
Enter the name of student 1: sarif
Enter score for subject 1 for sarif: 99
Enter score for subject 2 for sarif: 98
Enter score for subject 3 for sarif: 97
Enter the name of student 2: badmas
Enter score for subject 1 for badmas: 1
Enter score for subject 2 for badmas: 2
Enter score for subject 3 for badmas: 3
Enter the name of student 3: ankit
Enter score for subject 1 for ankit: 99
Enter score for subject 2 for ankit: 100
Enter score for subject 3 for ankit: 100
sarif: Average Score = 98.0
badmas: Average Score = 2.0
ankit: Average Score = 99.66666666666667
```

**Approach:**

The code allows users to enter names and scores for three students across three subjects, and then it
calculates and displays each student's average score.