ASSIGNMENT 1

Name:- Shumaila Shakeel *Subject:- Python*
Class:- MCA-A
Git-hub link:- https://github.com/Shumaila1617/python-assignment-1.git

Part 1: Operators:

Exercise 1: Arithmetic Operators

Objective:

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 first number:"))
num2=float(input("Enter second number:"))
# Perform calculations
addition = num1 + num2
subtraction = num1 - num2
multiplication = num1 * num2
division = num1 / num2
modulus = num1 % num2
exponentiation = num1 ** num2
floor_division = num1 // num2
# Display results
print(f"Addition: {addition}")
print(f"Subtraction: {subtraction}")
print(f"Multiplication: {multiplication}")
print(f"Division: {division:.2f}")
print(f"Modulus: {modulus}")
print(f"Exponentiation: {exponentiation}")
print(f"Floor Division: {floor_division}")
```

Output:

```
Enter first number:10
Enter second number:3
Addition: 13.0
Subtraction: 7.0
Multiplication: 30.0
Division: 3.33
Modulus: 1.0
Exponentiation: 1000.0
Floor Division: 3.0
```

Explanation:

I have taken two numbers from the user and used Python's built-in operators (+, -, *, /, %, **, and //)
to perform the required calculations. The results are then displayed to the user.

Key Learnings:

This exercise helped me practice how to handle user inputs and perform basic arithmetic operations
in Python. I also learned about floor division and exponentiation.

Exercise 2: Comparison Operators

Objective:

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

Code:

```
# Comparison Operations
# Get input from the user
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
# Perform comparisons
greater_than = num1 > num2
equal_to = num1 == num2
less_than_or_equal = num1 <= num2
# Display results
print(f"Is the first number greater than the second?
{greater_than}")
print(f"Is the first number equal to the second? {equal_to}")
print(f"Is the first number less than or equal to the second?
{less_than_or_equal}")
```

Output:

```
Enter first number: 23
Enter second number: 34
Is the first number greater than the second? False
Is the first number equal to the second? False
Is the first number less than or equal to the second? True
```

Explanation:

I have used the comparison operators (>, ==, <=) to compare the two user-input numbers and have
printed the results.

Key Learnings:

This task familiarized me with conditional comparisons and Python interprets them when
comparison are done between two values.

Exercise 3: Logical Operators

Objective:

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

Code:

```
# Logical Operations
# Get input from the user
a = input("Enter first boolean value (True/False): ") == 'True'
b = input("Enter second boolean value (True/False): ") == 'True'
c = input("Enter third boolean value (True/False): ") == 'True'
# Combine boolean values using logical operators
and_result = a and b and c
or_result = a or b or c
not_result1 = not a
not_result2 = not b
not_result3 = not c
# Display results
print(f"AND result: {and_result}")
print(f"OR result: {or_result}")
print(f"NOT first boolean: {not_result1}")
print(f"NOT second boolean: {not_result2}")
print(f"NOT third boolean: {not_result3}")
```

Output:

```
Enter first boolean value (True/False): True
Enter second boolean value (True/False): False
Enter third boolean value (True/False): True
AND result: False
OR result: True
NOT first boolean: False
NOT second boolean: True
NOT third boolean: False
```

Explanation:

I have used boolean values as input and applied logical operators to compute the combined result.

This program prints the results of and, or, and not operations.

Key Learnings:

This exercise is useful as it made me understand who logical operations does work in Python and
how to use them to evaluate conditions.

<u>Part 2: Strings</u>

Exercise 4: String Manipulation

Objective:

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

Code:

```
 # String Manipulation
# Get string input from the user
user_string = input("Enter a string: ")
# Perform string manipulations
string_length = len(user_string)
first_character = user_string[0]
last_character = user_string[-1]
reversed_string = user_string[::-1]
uppercase_string = user_string.upper()
lowercase_string = user_string.lower()
# Display results
print(f"Length of string: {string_length}")
print(f"First character: {first_character}")
print(f"Last character: {last_character}")
print(f"Reversed string: {reversed_string}")
print(f"Uppercase string: {uppercase_string}")
print(f"Lowercase string: {lowercase_string}")
```

Output:

```
Enter a string: Shumaila
Length of string: 8
First character: S
Last character: a
Reversed string: aliamuhS
Uppercase string: SHUMAILA
Lowercase string: shumaila
```

Explanation:

This program allows the user to input a string and performs several operations on it. First, it calculates the length of the string using len(), which counts the number of characters. It then extracts the first and last characters using indexing—[0] for the first and [-1] for the last. Next, it reverses the string by using slicing with [::-1], which starts from the end and steps backward. It converts the string to uppercase using the upper() method and to lowercase using lower(). Finally, the results are printed, allowing the user to see the string's length, the first and last characters, the reversed string, and how it looks in uppercase and lowercase.

Exercise 5: String Formatting

Objective:

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

Code:

```
# String Formatting
# Get user input for name and age
name = input("Enter your name: ")
age = input("Enter your age: ")
# Display message
print(f"Hello {name}, you are {age} years old.")
```

Output:

```
Enter your name: Shumaila
Enter your age: 23
Hello Shumaila, you are 23 years old.
```

Explanation:

I have used the Python’s f-string formatting to display the name and age in the required format.

Key Learnings:

This exercise introduced me to string formatting techniques, which are useful for creating formatted
output.

Exercise 6: Substring Search

Objective:

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: I am a student of MCA
Enter a word to search for: am
The word 'am' exists in the sentence at index 2.
```

Explanation:

I have used Python’s in operator to check for the presence of the word and index() method to find its
position.

Key Learnings:

This exercise taught me how to search for substrings in a string and how to handle user inputs in a
search operation.

<u>Part 3: Lists</u>

Exercise 7: List Operations

Objective:

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: 12
Enter number 2: 23
Enter number 3: 34
Enter number 4: 45
Enter number 5: 56
List of numbers: [12.0, 23.0, 34.0, 45.0, 56.0]
Sum of the numbers: 170.0
Largest number: 56.0
Smallest number: 12.0
```

Explanation:

I used Python’s sum (), max (), and min () functions to perform these operations on a list of user-
input numbers.

Key Learnings:

I became familiar with manipulating lists and applying basic list operations in Python.

Exercise 8: List Manipulation

Objective:

Add and remove elements from a list of fruits.

Code:

```
# List Manipulation
# Create a list of favorite fruits
fruits = ["Apple", "Banana", "Cherry", "Mango", "Orange"]
# Add a new fruit
fruits.append("Pineapple")
# Remove the second fruit
fruits.pop(1)
# Display the updated list
print(f"Updated fruit list: {fruits}")
```

Output:

```
Updated fruit list: ['Apple', 'Cherry', 'Mango',
'Orange', 'Pineapple']
```

Explanation:

I used append () to add a fruit to the list and pop () to remove the second fruit.

Key Learnings:

This exercise introduced me to modifying lists by adding and removing elements dynamically.

Exercise 9: Sorting a List

Objective:

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

Code:

```
# Get input from the user for a list of 5 numbers
```

```
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)
print("List in ascending order:", numbers_ascending)

# Sort the list in descending order
numbers_descending = sorted(numbers, reverse=True)
print("List in descending order:", numbers_descending)
```

Output:

```
Enter number 1: 12
Enter number 2: 34
Enter number 3: 56
Enter number 4: 67
Enter number 5: 78
List in ascending order: [12.0, 34.0, 56.0, 67.0, 78.0]
List in descending order: [78.0, 67.0, 56.0, 34.0, 12.0]
```

Explanation:

I used Python’s sorted() function to sort the list in both orders.

Key Learnings:

I learned how to sort lists using the sorted() function and how to reverse the sorting order using the
reverse argument.

Exercise 10: List Slicing

Objective:

Slice a list to retrieve specific subsets of elements.

Given the list numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], perform the following: 1. Print the first 5 elements. 2. Print the last 5 elements. 3. Print the elements from index 2 to index 7.

Code:

```
# List Slicing
n = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
# Print the first 5 elements
print(f"First 5 elements: {n[:5]}")
# Print the last 5 elements
print(f"Last 5 elements: {n[-5:]}")
# Print elements from index 2 to index 7
print(f"Elements from index 2 to 7: {n[2:8]}")
```

Output:

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

Explanation:

I used Python’s list slicing ([:5], [-5:], and [2:8]) to extract specific portions of the list.

Key Learnings:

This task helped me understand the concept of slicing lists to retrieve specific elements or ranges.

<u>Bonus Challenge: Nested List</u>

Exercise 11: Nested List

Objective:

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

Code:

```
# Create an empty list to store student data
student_data = []

# Get input for 3 students
for i in range(3):
    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
{name}: "))
        scores.append(score)
    student_data.append([name, scores])

# Calculate and print average score for each student
for student in student_data:
    name = student[0]
    scores = student[1]
    average_score = sum(scores) / len(scores)
    print(f"{name}'s average score: {average_score:.2f}")
```
Output:
```
Enter the name of student 1: Ankit
Enter score for subject 1 for Ankit: 89
Enter score for subject 2 for Ankit: 90
Enter score for subject 3 for Ankit: 87
Enter the name of student 2: Shumaila
Enter score for subject 1 for Shumaila: 76
Enter score for subject 2 for Shumaila: 89
Enter score for subject 3 for Shumaila: 88
Enter the name of student 3: Shivam
Enter score for subject 1 for Shivam: 88
Enter score for subject 2 for Shivam: 77
Enter score for subject 3 for Shivam: 80
Ankit's average score: 88.67
Shumaila's average score: 84.33
Shivam's average score: 81.67
```

Explanation:

I created a nested list structure to store students' names and scores. Then, I used a loop to calculate
the average score for each student and display it.

Key Learnings:

This exercise taught me how to work with nested data structures and how to manipulate lists within
lists.

Conclusion

This assignment reinforced my understanding of Python operators, strings, and lists. It allowed me to
explore essential Python concepts, such as arithmetic and logical operations, string manipulations,
and list processing. Additionally, I gained practical experience in writing clean, readable code with
comments explaining each step. This will be beneficial for more complex projects and collaborative
work in the future.