Name: Anuska Ghosh

Course: BCA (Sec A)

Registration Number: PROV/BCA/7/24/012

Github Repository Link: https://github.com/anuskaghosh17/Python-Assignment-1

Part 1: Operators

Exercise 1: Arithmetic Operators

Write a Python program to perform the following operations:

1. Add, subtract, multiply, and divide two numbers (input by the user).

2. Use the modulus operator to find the remainder of their division.

3. Use the exponentiation operator to raise the first number to the power of the second number.

4. Perform floor division on the two numbers.

Expected Input:

Enter first number: 10

Enter second number: 3

Expected Output:

Addition: 13

Subtraction: 7

Multiplication: 30

Division: 3.33

Modulus: 1

Exponentiation: 1000

Floor Division: 3

SOLUTION:

```
# Input of two numbers by the user
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))

# Perform the following operations
print("Addition:",a+b)
print("Subtraction:",a-b)
print("Multiplication:",a*b)
print('division:',round(a/b,2))
print("Modulus:",a%b)
print("Exponentiation:",a**b)
print("Floor Division:",a//b)
```

OUTPUT:

```
Enter first number: 10
Enter second number: 3
Addition: 13
Subtraction: 7
Multiplication: 30
division: 3.33
Modulus: 1
Exponentiation: 1000
Floor Division: 3
```

EXPLANATION:

1. **Input of two numbers by the user**

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

b=int(input("Enter second number:"))

INPUT: The code initiate the user to enter two numbers.

PROCESSING:

input(): A function that records user input as a string.

int(): A function that converts that string to an integer.

STORAGE:

a and b: Variables that stores the two numbers.

2. **Perform Operations**

The code performs several mathematical operations using the variables a and b, and then prints the results.

FOR ADDITION: print("Addition:",a+b)

Calculates the sum of a and b.

FOR SUBTRACTION: print("Subtraction:",a-b)

Calculates the difference between a and b.

FOR MULTIPLICATION: print("Multiplication:",a*b)

Calculates the product of a and b.

FOR FLOAT DIVISION: print('division:',round(a/b,2))

Calculates the float division of a by b i.e., the result will be in decimals.

round(): A function which rounds off the result to two decimal places.

FOR MODULUS: print("Modulus:",a%b)

Calculates the remainder when a is divided by b.

FOR EXPONENTIATION: print("Exponentiation:",a**b)

Calculates a raise to the power of b.

FOR FLOOR/INTEGER DIVISION: print("Floor Division:",a//b)

Performs floor division, which can be used to divide two numbers and get the result in which the quotient is rounded down to the nearest integer value.

print(): A function used to display result on the screen.

SUMMARY:

The above code takes two integers from the user and performs a series of basic arithmetic operations, displaying the results for addition, subtraction, multiplication, division (rounded), modulus, exponentiation, and floor division.

Exercise 2: Comparison Operators

Write a Python program that asks for two numbers and checks:

1. If the first number is greater than the second.

2. If the first number is equal to the second.

3. If the first number is less than or equal to the second.

Print the results.

SOLUTION:

```
# Input of two numbers by the user
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))

# Comparison between the two numbers using Comparison operators
print(a>b)
print(a==b)
print(a<=b)
```

OUTPUT:

```
Enter first number: 1
Enter second number: 2
False
False
True
```

EXPLANATION:

1. **Input of two numbers by the user**

```
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))
```

INPUT: The code initiate the user to enter two numbers.

PROCESSING:

input(): A function that records user input as a string.

int(): A function that converts that string to an integer.

STORAGE:

a and b: Variables that stores the two numbers.

2. Comparison Operations

The code performs a few comparison operations between the two numbers and prints the results.

Greater Than: print(a > b)

Operation: Checks if a is greater than b.

Output: This will print True if a is greater than b, otherwise it will print False.

Equality: print(a == b)

Operation: Checks if a is equal to b.

Output: This will print True if the two numbers are equal, otherwise it will print False.

Less Than or Equal To: print(a <= b)

Operation: Checks if a is less than or equal to b.

Output: This will print True if a is less than or equal to b, otherwise it will print False.

print(): A function used to display result on the screen

SUMMARY:

The above code allows the user to input two integers and then compares these two numbers using comparison operators (>, ==, and <=). It prints the results of these comparisons, which will be either True or False.

Exercise 3: Logical Operators

Write a Python program that:

1. Takes three boolean values (True or False) as input.

2. Uses and, or, and not operators to return the result of combining them.

SOLUTION:

```
# Taking three boolean values as input
a = input("Enter first boolean (True or False): ").strip().lower() == 'true'
b = input("Enter second boolean (True or False): ").strip().lower() == 'true'
c = input("Enter third boolean (True or False): ").strip().lower() == 'true'

# Using logical operators
print("AND operation result:", a and b and c)
print("OR operation result:", a or b or c)
print("Negation results:", not a, not b, not c)
```

OUTPUT:

```
Enter first boolean (True or False):  true
Enter second boolean (True or False):  false
Enter third boolean (True or False):  false
AND operation result: False
OR operation result: True
Negation results: False True True
```

EXPLANATION:

1. Input of Three Boolean Values

a = input("Enter first boolean (True or False): ").strip().lower() == 'true'

b = input("Enter second boolean (True or False): ").strip().lower() == 'true'

c = input("Enter third boolean (True or False): ").strip().lower() == 'true'

INPUT: The code initiates the user to enter three boolean values (either True or False).

PROCESSING:

input(): A function that records user input as a string.

strip(): A function that removes any leading or trailing whitespace from the input.

lower(): A function that converts the input string to lowercase.

The comparison == 'true': converts the input to a boolean value i.e., if the input is 'true', the variable is set to True; otherwise, it is set to False.

STORAGE:

a and b: Variables that stores the three Boolean values.

2. Using Logical Operators

The code performs logical operations on the three boolean values and prints the results.

AND Operation: print("AND operation result:", a and b and c)

> **Operation**: The AND operator checks if all three boolean values are True.
>
> **Output**: It will print True if a, b, and c are all True; otherwise, it will print False.

OR Operation: print("OR operation result:", a or b or c)

> **Operation**: The OR operator checks if at least one of the boolean values is True.
>
> **Output**: It will print True if at least one of a, b, or c is True; otherwise, it will print False.

Negation: print("Negation results:", not a, not b, not c)

> **Operation**: The NOT operator negates each boolean value.
>
> **Output**: It will print the negated values of a, b, and c. If a value is True, it will print False, and vice versa.

print(): A function used to display result on the screen

SUMMARY:

The above code takes three boolean inputs from the user and converts them into actual boolean values. It then performs and prints the results of logical operations: AND, OR, and the negation of each value.

Part 2: Strings

Exercise 4: String Manipulation

1. Take a string input from the user.

2. Display the following:

o The length of the string.

o The first and last character.

o The string in reverse order.

o The string in uppercase and lowercase.

SOLUTION:

```
#Take a string input from the user

str=input("Enter a string:")


# Display the following

print("The length of the string is:",len(str))

print("The first character is:",str[0],",","The last character is:",str[-1])

print("The string in reverse order is:",str[::-1])

print("The string in uppercase is:",str.upper(),",","The string in lowercase is:",str.lower())
```

OUTPUT:

```
Enter a string: ANUSKA

The length of the string is: 6

The first character is: A , The last character is: A

The string in reverse order is: AKSUNA

The string in uppercase is: ANUSKA , The string in lowercase is: anuska
```

EXPLANATION:

1. **Taking string input from the user**

str = input("Enter a string:")

INPUT: The code initiates the user to enter a string.

PROCESSING:

input(): A function that records user input as a string.

STORAGE:

str: Variable that stores the entered string.

2. Displaying Information About the String

Length of the String: print("The length of the string is:", len(str))

Operation: The len() function calculates the number of characters in the string.

Output: It prints the length of the string.

First and Last Characters: print("The first character is:", str[0], ",", "The last character is:", str[-1])

Operation:

- str[0] gives the first character of the string (indexing starts at 0).
- str[-1] gives the last character of the string (negative indexing allows access from the end).

Output: It prints the first and last characters of the string.

Reversed String: print("The string in reverse order is:", str[::-1])

Operation: str[::-1] uses slicing to reverse the string. The '::' indicates the entire string, and -1 means to step backwards.

Output: It prints the string in reverse order.

Uppercase and Lowercase Versions: print("The string in uppercase is:", str.upper(), ",", "The string in lowercase is:", str.lower())

Operation:

- str.upper() converts all characters in the string to uppercase.
- str.lower() converts all characters in the string to lowercase.

Output: It prints the uppercase and lowercase versions of the string.

print(): A function used to display result on the screen

SUMMARY:

The above code takes a string input from the user and displays various information about it: its length, the first and last characters, the string in reverse order, and its uppercase and lowercase forms.

Exercise 5: String Formatting

Write a program that asks for the user's name and age, and displays the message in this format:

Hello [Name], you are [Age] years old.

SOLUTION:

```
# Take user's name and age as input
Name=input("Enter your name:")
Age=int(input("Enter your age:"))

# Display the message
print("Hello",Name,",","you are",Age,"years old.")
```

OUTPUT:

```
Enter your name: Anuska
Enter your age: 22
Hello Anuska , you are 22 years old.
```

EXPLANATION:

1. Taking User's Name as Input

Name = input("Enter your name:")

INPUT: The code initiates the user to enter their name.

PROCESSING:

input(): A function that records user input as a string.

STORAGE:

Name: Variable that stores the entered name as a string.

2. Taking User's Age as Input

Age = int(input("Enter your age:"))

INPUT: The code initiates the user to enter their age.

PROCESSING:

input(): A function that records user input as a string.

int(): A function that converts that string to an integer.

STORAGE:

Age: Variable that stores the age.

3. Displaying a Message

print("Hello", Name, ",", "you are", Age, "years old.")

Output: This line prints a greeting message using the values stored in Name and Age.

Formatting: The print() function automatically separates its arguments with spaces, so the output will appear formatted nicely.

- It will greet the user with their name and inform them of their age.

print(): A function used to display result on the screen.

SUMMARY:

The above code collects a user's name and age and then displays a personalized greeting that includes both pieces of information i.e, name and age.

Exercise 6: Substring Search

Write a Python program that:

1. Asks for a sentence input from the user.

2. Asks for a word to search in the sentence.

3. Outputs whether the word exists in the sentence and, if it does, at which position (index).

SOLUTION:

```
# Take a sentence as input from the user
sentence=input("Enter a sentence:")


# Input a word to search in the sentence
word=input("Enter a word to search:")


# Checking whether the word exists in the sentence or not
if word in sentence:
   print(sentence.index(word))
else:
   print("Word not found")
```

OUTPUT:

```
Enter a sentence: My name is Anuska Ghosh
Enter a word to search: name
3
```

EXPLANATION:

1. Taking a Sentence as Input

sentence = input("Enter a sentence:")

INPUT: The code initiates the user to enter a sentence.

PROCESSING:

input(): A function that records user input as a string.

STORAGE:

sentence: Variable that stores the entered sentence as a string.

2. Inputting a Word to Search

word = input("Enter a word to search:")

INPUT: The code initiates the user to enter a word they want to search for in the sentence.

PROCESSING:

input(): A function that records user input as a string.

STORAGE:

word: Variable that stores the entered word.

3. Checking for the Word in the Sentence

if word **in** sentence:

Operation: This line checks whether the word exists in the sentence.

Condition: The expression 'word in sentence' evaluates to True if the word is found and False otherwise.

4. Printing the Result

If the Word Exists: print(sentence.index(word))

Operation: If the word is found in the sentence, the index() method is called on sentence.

Output: This method returns the starting index of the first occurrence of the word in the sentence and prints it.

If the Word Does Not Exist:

else:

print("Word not found")

Output: If the word is not found in the sentence, it prints "Word not found".

print(): A function used to display result on the screen.

SUMMARY: The above code takes a sentence and a word as input from the user. It checks if the word exists in the sentence and prints the starting index of the word if found; otherwise, it notifies the user that the word was not found.

Part 3: Lists

Exercise 7: List Operations

Write a Python program that:

1. Creates a list of 5 numbers (input from the user).

2. Displays the sum of all the numbers in the list.

3. Finds the largest and smallest number in the list.

SOLUTION:

```
# Input of 5 numbers from the user

num1=int(input())

num2=int(input())

num3=int(input())

num4=int(input())

num5=int(input())


# Create a list of 5 numbers

list=[num1,num2,num3,num4,num5]


# Printing the list

print(list)


# Display the sum of all the numbers in the list

print("The sum of all the numbers in the list:",sum(list))


# Find the largest and smallest number in the list

print("The largest number in the list is:",max(list))

print("The smallest number in the list is:",min(list))
```

OUTPUT:

```
1

2

3

4

5

[1, 2, 3, 4, 5]

The sum of all the numbers in the list: 15

The largest number in the list is: 5

The smallest number in the list is: 1
```

EXPLANATION:

1. Input of 5 Numbers

num1 = int(input())

num2 = int(input())

num3 = int(input())

num4 = int(input())

num5 = int(input())

INPUT: The code initiates the user to enter five numbers one by one.

PROCESSING:

input(): A function that records user input as a string.

int(): A function that converts that string to an integer.

STORAGE:

num1,num2,num3,num4,num5: Variables that stores the five numbers.

2. Creating a List of 5 Numbers

list = [num1, num2, num3, num4, num5]

Operation: A list named list is created containing the five numbers.

3. Printing the List: print(list)

Output: This line prints the contents of the list, showing the five numbers entered by the user.

4. Displaying the Sum of All Numbers

print("The sum of all the numbers in the list:", sum(list))

Operation: The sum() function calculates the total of all the numbers in the list.

Output: It prints the sum of the numbers.

5. Finding the Largest and Smallest Numbers

print("The largest number in the list is:", max(list))

print("The smallest number in the list is:", min(list))

- **Largest Number**:
    - max(list) finds the largest number in the list and prints it.
- **Smallest Number**:
    - min(list) finds the smallest number in the list and prints it.

print(): A function used to display result on the screen.

SUMMARY: The above code takes five numbers from the user, stores them in a list, and prints the list. It then calculates and displays the sum of the numbers, as well as the largest and smallest numbers in the list.

Exercise 8: List Manipulation

1. Create a list of 5 of your favorite fruits.

2. Perform the following:

o Add one more fruit to the list.

o Remove the second fruit from the list.

o Print the updated list.

SOLUTION:

```
# Create a list of 5 favorite fruits

fruits=['mango','watermelon','guava','apple','banana']


# Add one more fruit to the list

fruits.append('orange')


# Remove the second fruit from the list

fruits.remove('watermelon')


# Print the updated list

print("The updated list of 5 favorite fruits is:",fruits)
```

OUTPUT:

```
The updated list of 5 favorite fruits is: ['mango', 'guava', 'apple', 'banana',
'orange']
```

EXPLANATION:

1. Creating a List of Fruits

fruits = ['mango', 'watermelon', 'guava', 'apple', 'banana']

List Creation: This line creates a list called fruits containing five favorite fruits: 'mango', 'watermelon', 'guava', 'apple', and 'banana'.

Storage: The fruits are stored as strings within the list.

2. Adding a Fruit to the List

fruits.append('orange')

Operation: The append() method adds the string 'orange' to the end of the fruits list.

Output: The list now contains six fruits.

3. Removing the Second Fruit

fruits.remove('watermelon')

Operation: The remove() method removes the first occurrence of the string 'watermelon' from the list.

Output: After this operation, 'watermelon' is no longer in the list.

4. Printing the Updated List

print("The updated list of 5 favorite fruits is:", fruits)

Output: This line prints the message along with the current contents of the fruits list, showing the updated list of favorite fruits.

print(): A function used to display result on the screen.

SUMMARY:

The above code creates a list of five favorite fruits, adds another fruit ('orange'), removes one fruit ('watermelon'), and then prints the updated list of fruits.

Exercise 9: Sorting a List

Write a Python program that:

1. Asks the user to input a list of 5 numbers.

2. Sorts the list in ascending order and displays it.

3. Sorts the list in descending order and displays it.

SOLUTION:

```
# Input of 5 numbers from the user

num1=int(input())

num2=int(input())

num3=int(input())

num4=int(input())

num5=int(input())


# Create a list of 5 numbers

list=[num1,num2,num3,num4,num5]


# Printing the list

print(list)


# Sort the list in ascending order

list.sort()

print("The list in ascending order:",list)


# Sort the list in descending order

list.sort(reverse=True)

print("The list in descending order:",list)
```

OUTPUT:

```
1

 2

 3

 4

 5

[1, 2, 3, 4, 5]

The list in ascending order: [1, 2, 3, 4, 5]

The list in descending order: [5, 4, 3, 2, 1]
```

EXPLANATION:

1. Input of 5 Numbers

num1 = int(input())

num2 = int(input())

num3 = int(input())

num4 = int(input())

num5 = int(input())

INPUT: The code initiates the user to enter five numbers one by one.

PROCESSING:

input(): A function that records user input as a string.

int(): A function that converts that string to an integer.

STORAGE:

num1,num2,num3,num4,num5: Variables that stores the five numbers.

2. Creating a List of 5 Numbers

list = [num1, num2, num3, num4, num5]

Operation: A list named list is created containing the five numbers.

3. Printing the List: print(list)

Output: This line prints the contents of the list, showing the five numbers entered by the user.

4. Sorting the List in Ascending Order

list.sort()

print("The list in ascending order:", list)

Operation: The sort() method sorts the list in ascending order (from smallest to largest).

Output: The updated list in ascending order is printed.

5. Sorting the List in Descending Order

list.sort(reverse=True)

print("The list in descending order:", list)

Operation: The sort(reverse=True) method sorts the list in descending order (from largest to smallest).

Output: The updated list in descending order is printed.

print(): A function used to display result on the screen.

SUMMARY:

The above code takes five numbers from the user, stores them in a list, prints the original list, sorts it in ascending order, and prints the sorted list. It then sorts the same list in descending order and prints that as well.

Exercise 10: List Slicing

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.

SOLUTION:

```
# Enter the list of numbers given

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


# Print the first 5 elements

print(numbers[0:5])


# Print the last 5 elements

print(numbers[-5:])


# Print the elements from index 2 to index 7

print(numbers[2:8])
```

OUTPUT:

```
[1, 2, 3, 4, 5]

[6, 7, 8, 9, 10]

[3, 4, 5, 6, 7, 8]
```

EXPLANATION:

1. Entering the List of Numbers

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

List Creation: This line creates a list called numbers containing the integers from 1 to 10.

Storage: The list stores these integers in sequential order.

2. Printing the First 5 Elements

print(numbers[0:5])

Slicing: The slice numbers[0:5] gives elements from index 0 up to index 5(but not including index 5).

Output: This will print the first five elements of the list: [1, 2, 3, 4, 5].

3. Printing the Last 5 Elements

print(numbers[-5:])

Slicing with Negative Indexing: The slice numbers[-5:] gives elements starting from index -5 (which corresponds to the fifth element from the end) to the end of the list.

Output: This will print the last five elements of the list: [6, 7, 8, 9, 10].

4. Printing Elements from Index 2 to Index 7

print(numbers[2:8])

Slicing: The slice numbers[2:8] gives elements from index 2 up to index 8(but not including index 8).

Output: This will print the elements from the third element at index 2 to the eighth element at index 7 in the list: [3, 4, 5, 6, 7, 8].

print(): A function used to display result on the screen.

SUMMARY:

The above code initializes a list of numbers from 1 to 10 and demonstrates slicing to print:

- The first five elements,
- The last five elements, and
- The elements from index 2 to index 7.

Bonus Challenge

Exercise 11: Nested List

Write a Python program that:

1. Takes input of 3 students' names and their respective scores in 3 subjects.

2. Stores them in a nested list.

3. Prints each student's name and their average score.

SOLUTION:

```
# Creating an empty list for names of students
students = []

# Input of 3 students' names
for i in range(3):
    name = input(f"Enter the name of student {i + 1}:")

    # Creating an empty list for scores
    scores = []

    # Input their respective scores in 3 subjects
    for j in range(3):
        score = float(input(f"Enter the score for subject {j + 1} for {name}:"))
        scores.append(score)

    # Append the student name and scores to the students list
    students.append([name, scores])

# Print each student's name and their average score
print("\nStudent Average Scores:")
```

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

OUTPUT:

```
Enter the name of student 1: ANUSKA
Enter the score for subject 1 for ANUSKA: 19
Enter the score for subject 2 for ANUSKA: 18
Enter the score for subject 3 for ANUSKA: 20
Enter the name of student 2: ANANNYA
Enter the score for subject 1 for ANANNYA: 17.5
Enter the score for subject 2 for ANANNYA: 18.5
Enter the score for subject 3 for ANANNYA: 19.5
Enter the name of student 3: ANUSHKA
Enter the score for subject 1 for ANUSHKA: 17
Enter the score for subject 2 for ANUSHKA: 18.5
Enter the score for subject 3 for ANUSHKA: 16

Student Average Scores:
ANUSKA: 19.0
ANANNYA: 18.5
ANUSHKA: 17.166666666666668
```

EXPLANATION:

1. Creating an Empty List for Students

students = []

List Creation: This line initializes an empty list called students that will be used to store the names and scores of students.

2. Input of 3 Students' Names

for i in range(3):

name = input(f"Enter the name of student {i + 1}:")

Loop: This for loop iterates three times (for i values 0, 1, and 2).

Input: Inside the loop, it prompts the user to enter the name of each student, with a formatted message indicating the student number.

Storage: Each entered name is stored in the variable ‘name’.

3. Creating an Empty List for Scores

scores = []

List Creation: This initializes an empty list called scores for storing the scores of the current student in the inner loop.

4. Inputting Scores for 3 Subjects

for j in range(3):

score = float(input(f"Enter the score for subject {j + 1} for {name}:"))

scores.append(score)

Inner Loop: This for loop iterates three times to collect scores for three subjects.

Input: It prompts the user to enter the score for each subject, using the current student's name in the message.

Conversion and Storage: Each score is converted to a float and appended to the scores list.

5. Appending Student Name and Scores to the Students List

students.append([name, scores])

Operation: This line creates a list containing the student's name and their corresponding scores, and appends it to the students list.

6. Printing Each Student's Name and Average Score

```
print("\nStudent Average Scores:")

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

Output Header: It prints a header for the average scores.

Loop Through Students: The for loop iterates over each student's data in the students list.

- name retrieves the student's name.
- scores retrieves the student's scores.
- **Average Calculation**: The average score is calculated by summing the scores and dividing by the number of subjects (len(scores)).

Output: Finally, it prints each student's name along with their average score.

print(): A function used to display result on the screen.

SUMMARY:

This above code creates a program that collects the names and scores of three students in three subjects. It calculates and displays each student's average score after all input has been gathered.