ASSIGNMENT - PYTHON

Name: Sohan Raikar

Class: MCA ‘A’

Repository Link: https://github.com/sohanraikar/Python-Assignment

Exercise 1: Arithmetic Operators

- **Objective**: 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 n umber.
    4. Perform floor division on the two numbers.
- **Approach**: The program will use input() to get two numbers from the user, then perform and display the results of each operation.
- **Key Learnings**: Understanding Python's arithmetic operators, the importance of type convers ion using float(), and gaining familiarity with various arithmetic operations in Python.

Exercise 2: Comparison Operators

- **Objective**: 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.
- **Approach**: The program will use input() to get two numbers from the user, then use if, elif, a nd else statements to compare the numbers and display the results.
- **Key Learnings**: Gained insight into using comparison operators and conditional statements to control program flow based on logical conditions.

Exercise 3: Boolean Logic

- **Objective**: 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.
- **Approach**: The program will use input() to take three boolean values, then apply and, or, and not operators to these values and display the results.
- **Key Learnings**: Learned to manipulate boolean values using logical operators, enabling control of program flow and decision-making.

Exercise 4: String Manipulation

- **Objective**: Write a Python program that:
    1. Takes a string input from the user.
    2. Displays the following:
        - The length of the string.
        - The first and last character.
        - The string in reverse order.
        - The string in uppercase and lowercase.
- **Approach**: The program will use input() to take a string from the user, then apply various string methods and indexing to extract and display the required information.
- **Key Learnings**: Gained practical knowledge of string methods, indexing, and slicing in Python for effective string manipulation.

Exercise 5: String Formatting

- **Objective**: 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.
- **Approach**: The program will use input() to get the user's name and age, then use an f-string to format and display the message.
- **Key Learnings**: Gained practical knowledge of using f-strings for convenient and readable string formatting in Python.

Exercise 6: Substring Search

- **Objective**: 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).
- **Approach**: The program will use input() to get a sentence and a word from the user, then check if the word is in the sentence and display the position if it exists.
- **Key Learnings**: Developed skills in string manipulation, searching for substrings, and using string methods to find the index of a word.

Exercise 7: List Operations

- **Objective**: 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.
- **Approach**: The program will use input() to get five numbers from the user, store them in a list, then calculate and display the sum, largest, and smallest numbers in the list.
- **Key Learnings**: Developed skills in list creation and manipulation, as well as using list methods to perform common operations like finding the sum, maximum, and minimum values.

Exercise 8: List Manipulation

- **Objective**: Write a Python program that:
    1. Creates a list of 5 of your favorite fruits.
    2. Performs the following:
        - Adds one more fruit to the list.
        - Removes the second fruit from the list.
        - Prints the updated list.
- **Approach**: The program will start by creating a list of favorite fruits, then use list methods to add a new fruit and remove the second one, finally printing the updated list.
- **Key Learnings**: Gained practical knowledge of list operations such as adding and removing elements, and updating the list.

Exercise 9: Sorting a List

- **Objective**: 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.
- **Approach**: The program will use input() to get five numbers from the user, store them in a list, then sort and display the list in both ascending and descending order.
- **Key Learnings**: Developed skills in list manipulation, sorting, and displaying lists in different orders using built-in methods.

Exercise 10: List Slicing

- **Objective**: 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.
- **Approach**: The program will use list slicing to extract and display the specified elements.
- **Key Learnings**: Gained practical knowledge of list slicing techniques to access specific parts of a list.

Exercise 11: Nested List

- **Objective**: 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.
- **Approach**: The program will use input() to get the names and scores, store this information in a nested list, and then calculate and display each student's average score.
- **Key Learnings**: Gained practical knowledge of handling nested lists and performing calculations on data stored within them.