In-Class Programming Assignment-2

Name: Sammeta Jaya Sai Charan

Student ID: 700739775

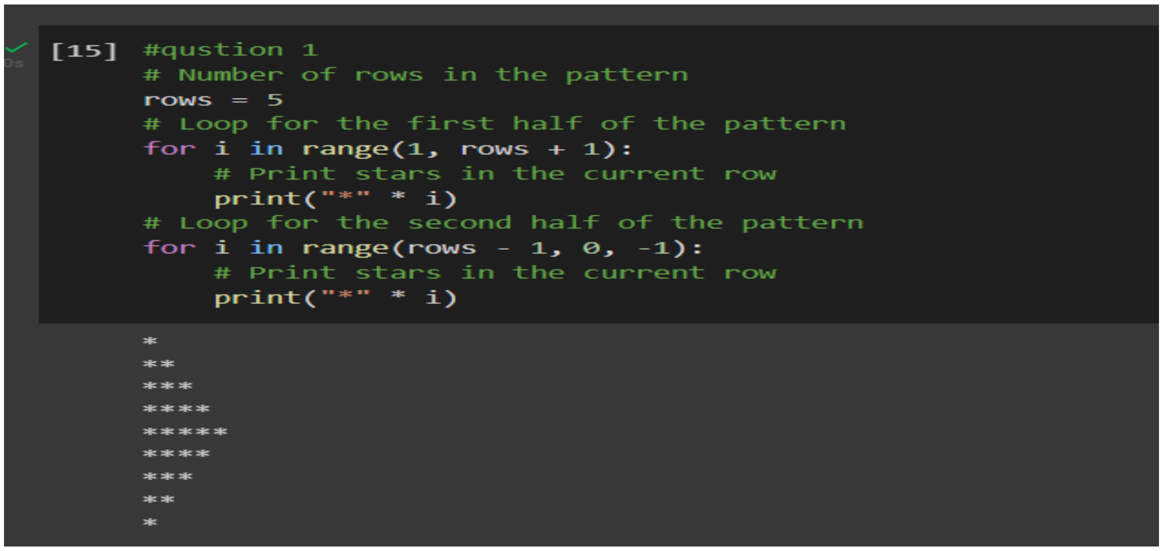
Github link: https://github.com/saicharan255/ICP_ML_2.git

Videolink:
https://drive.google.com/file/d/1yGi0tgD0VQofMFPpqMGKBN79OrhSvqVV/view?usp=sharing

Q1. Use a python code to display the star pattern using the for loop

Code:
```
# Number of rows in the pattern
rows = 5
# Loop for the first half of the pattern
for i in range(1, rows + 1):
    # Print stars in the current row
    print("*" * i)
# Loop for the second half of the pattern
for i in range(rows - 1, 0, -1):
    # Print stars in the current row
    print("*" * i)
```

Description:
From the above we have given the rows =5 then we written the for loop for first half pattern to print * by incrementing row + 1. In second half we have written another for loop to print the * in decrement form by row – 1.

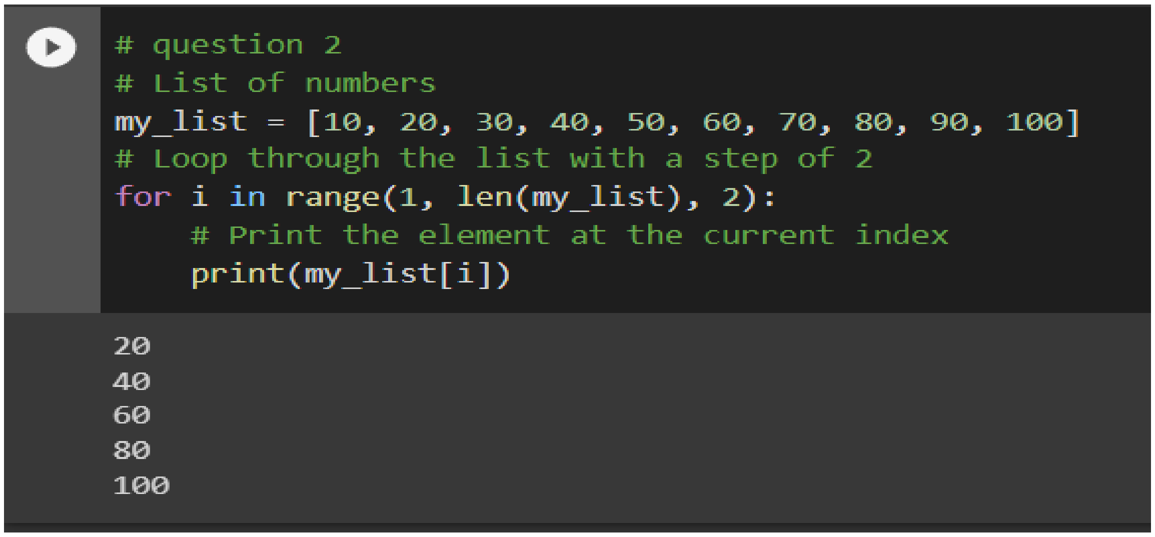

```
[15] #qustion 1
     # Number of rows in the pattern
     rows = 5
     # Loop for the first half of the pattern
     for i in range(1, rows + 1):
         # Print stars in the current row
         print("*" * i)
     # Loop for the second half of the pattern
     for i in range(rows - 1, 0, -1):
         # Print stars in the current row
         print("*" * i)

     *
     **
     ***
     ****
     *****
     ****
     ***
     **
     *
```

Q2. Use looping to output the elements from a provided list present at odd indexes.
my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

Code:

```
# List of numbers
my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
# Loop through the list with a step of 2
for i in range(1, len(my_list), 2):
    # Print the element at the current index
    print(my_list[i])
```

Description:
from the above code we have created a list with values, in for loop it starts iterating from index 1 with step of 2 and thus prints the odd indexes values.



```
# question 2
# List of numbers
my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
# Loop through the list with a step of 2
for i in range(1, len(my_list), 2):
    # Print the element at the current index
    print(my_list[i])
```

```
20
40
60
80
100
```

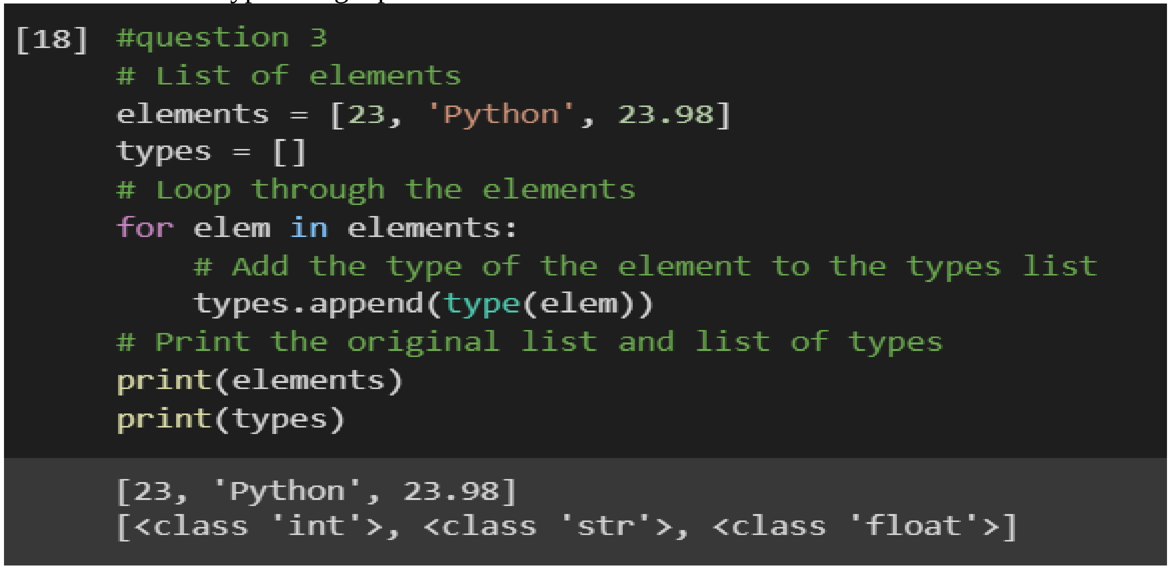
Q3. Write a code that appends the type of elements from a given list.
Input x = [23, ‘Python’, 23.98]
Expected output [23, 'Python', 23.98] [, ,]

Code:

```
# List of elements
elements = [23, 'Python', 23.98]
types = []
# Loop through the elements
for elem in elements:
    # Add the type of the element to the types list
    types.append(type(elem))
# Print the original list and list of types
print(elements)
print(types)
```

Description:

from the above code we have given random elements list with different data types. In for loop all elements are iterated, by using type() we determine the data types and then appended using append(). The elements and data types are get printed.



```
[18] #question 3
     # List of elements
     elements = [23, 'Python', 23.98]
     types = []
     # Loop through the elements
     for elem in elements:
         # Add the type of the element to the types list
         types.append(type(elem))
     # Print the original list and list of types
     print(elements)
     print(types)
```

```
[23, 'Python', 23.98]
[<class 'int'>, <class 'str'>, <class 'float'>]
```

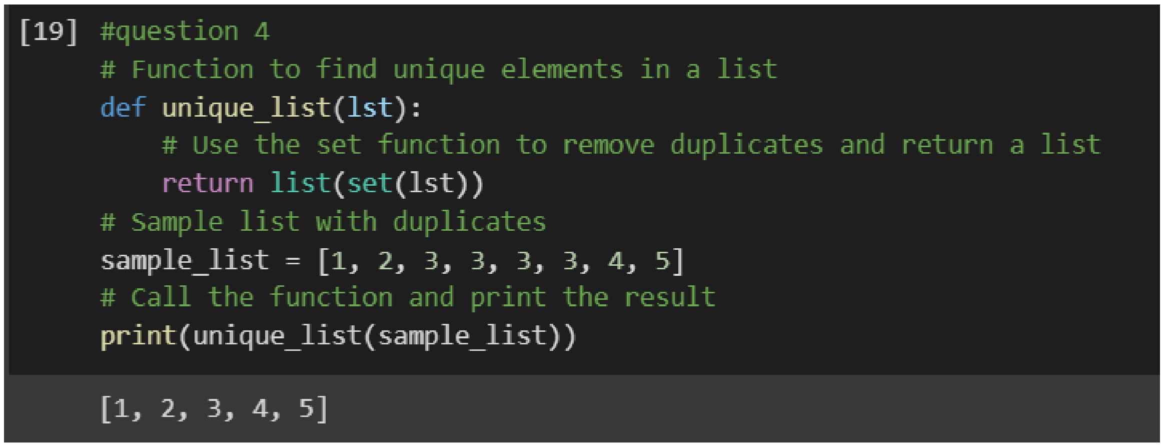
Q4. Write a function that takes a list and returns a new list with unique items of the first list. Sample List: [1,2,3,3,3,3,4,5] Unique List: [1, 2, 3, 4, 5]

Code:

```
#question 4
# Function to find unique elements in a list
def unique_list(lst):
   # Use the set function to remove duplicates and return a list
   return list(set(lst))
# Sample list with duplicates
sample_list = [1, 2, 3, 3, 3, 3, 4, 5]
# Call the function and print the result
print(unique_list(sample_list))
```

Description:

from the above code we used set() to remove the duplicates in the list the, return list will be the unique values and then it is called and printed



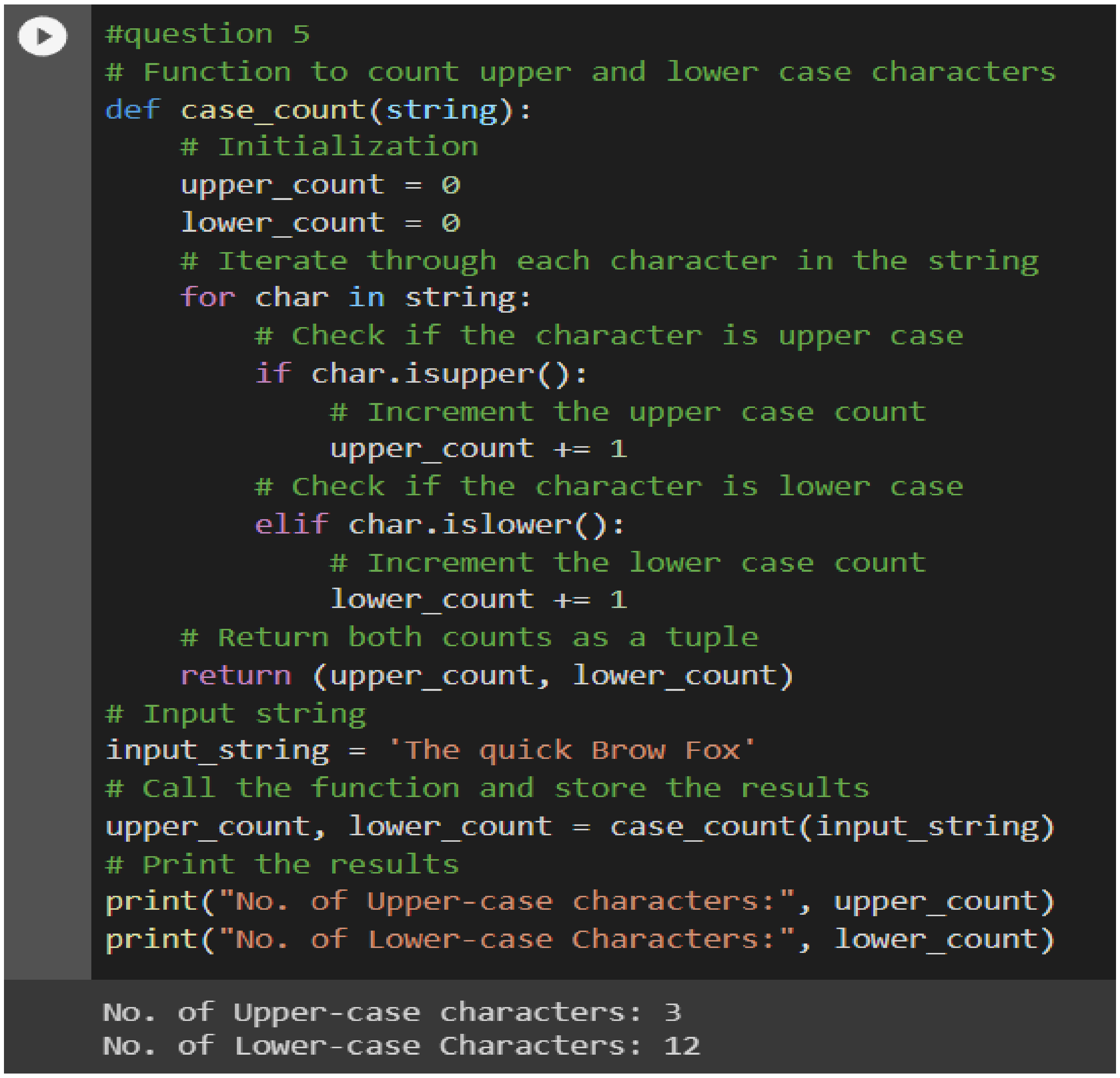
```
[19] #question 4
     # Function to find unique elements in a list
     def unique_list(lst):
         # Use the set function to remove duplicates and return a list
         return list(set(lst))
     # Sample list with duplicates
     sample_list = [1, 2, 3, 3, 3, 3, 4, 5]
     # Call the function and print the result
     print(unique_list(sample_list))

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

Q5. Write a function that accepts a string and calculate the number of upper-case letters and lower-case letters.
Input String: 'The quick Brow Fox'
Expected Output: No. of Upper-case characters: 3
 No. of Lower-case Characters: 12

Code:

```
#question 5
# Function to count upper and lower case characters
def case_count(string):
    # Initialization
    upper_count = 0
    lower_count = 0
    # Iterate through each character in the string
    for char in string:
        # Check if the character is upper case
        if char.isupper():
            # Increment the upper case count
            upper_count += 1
        # Check if the character is lower case
        elif char.islower():
            # Increment the lower case count
            lower_count += 1
    # Return both counts as a tuple
    return (upper_count, lower_count)
# Input string
input_string = 'The quick Brow Fox'
# Call the function and store the results
upper_count, lower_count = case_count(input_string)
# Print the results
print("No. of Upper-case characters:", upper_count)
print("No. of Lower-case Characters:", lower_count)
```

Description:
firlsty we have intitilazed the counts with 0 and by checking each character by upper or lower function we increment the respective counts then returning the values as tuple and print the count of the upper and lower case characters.



```
#question 5
# Function to count upper and lower case characters
def case_count(string):
    # Initialization
    upper_count = 0
    lower_count = 0
    # Iterate through each character in the string
    for char in string:
        # Check if the character is upper case
        if char.isupper():
            # Increment the upper case count
            upper_count += 1
        # Check if the character is lower case
        elif char.islower():
            # Increment the lower case count
            lower_count += 1
    # Return both counts as a tuple
    return (upper_count, lower_count)
# Input string
input_string = 'The quick Brow Fox'
# Call the function and store the results
upper_count, lower_count = case_count(input_string)
# Print the results
print("No. of Upper-case characters:", upper_count)
print("No. of Lower-case Characters:", lower_count)
```

```
No. of Upper-case characters: 3
No. of Lower-case Characters: 12
```