

# CS 5710 –Machine Learning

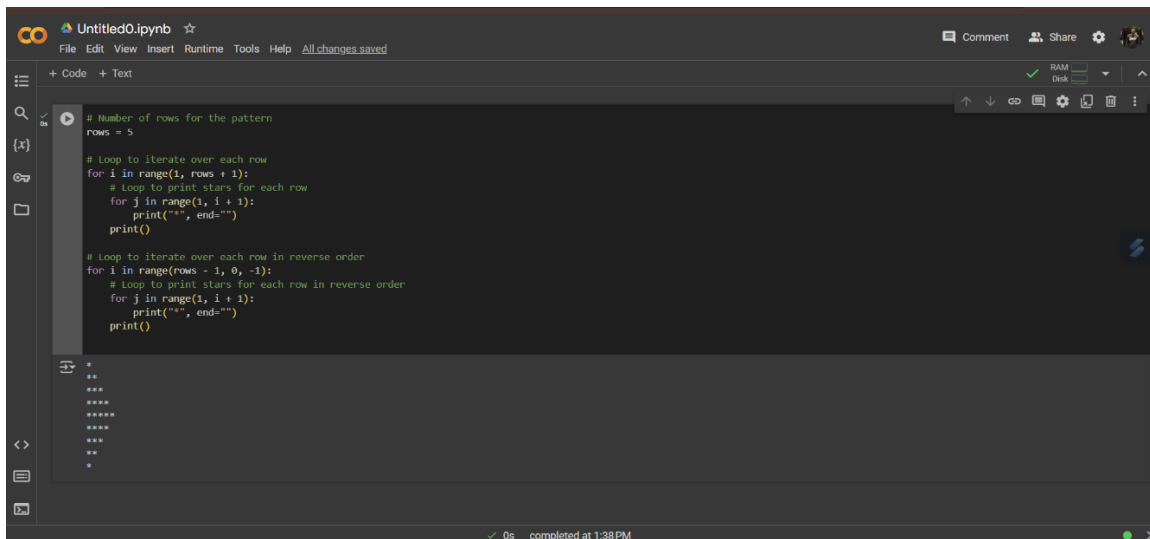
## Programming Assignment-2

GITHUB LINK: <https://github.com/vijender6/vijender>

RECORDING VIDEO LINK: [https://drive.google.com/file/d/1CG4iRetHS9xskLyUxv95y2eZo-agkRI9/view?usp=drive\\_link](https://drive.google.com/file/d/1CG4iRetHS9xskLyUxv95y2eZo-agkRI9/view?usp=drive_link)

NAME: VIJENDER REDDY KOOTURU  
700765220

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



The screenshot shows a Jupyter Notebook interface with a dark theme. The code cell contains the following Python code:

```
# Number of rows for the pattern
rows = 5

# Loop to iterate over each row
for i in range(1, rows + 1):
    # Loop to print stars for each row
    for j in range(1, i + 1):
        print("*", end=" ")
    print()

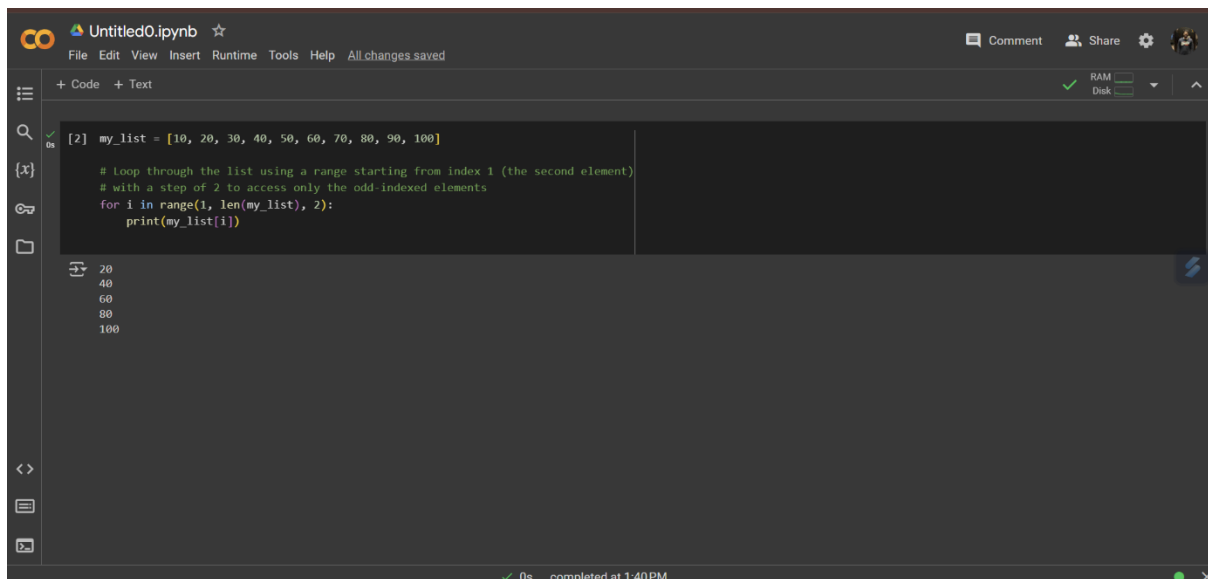
# Loop to iterate over each row in reverse order
for i in range(rows - 1, 0, -1):
    # Loop to print stars for each row in reverse order
    for j in range(1, i + 1):
        print("*", end=" ")
    print()
```

The output of the code is a star pattern that looks like this:

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

The status bar at the bottom indicates the code was completed at 1:38 PM.

2. 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]



Untitled0.ipynb

```
File Edit View Insert Runtime Tools Help All changes saved
```

+ Code + Text

```
[2] my_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]

# loop through the list using a range starting from index 1 (the second element)
# with a step of 2 to access only the odd-indexed elements
for i in range(1, len(my_list), 2):
    print(my_list[i])
```

```
20
40
60
80
100
```

0s completed at 1:40 PM

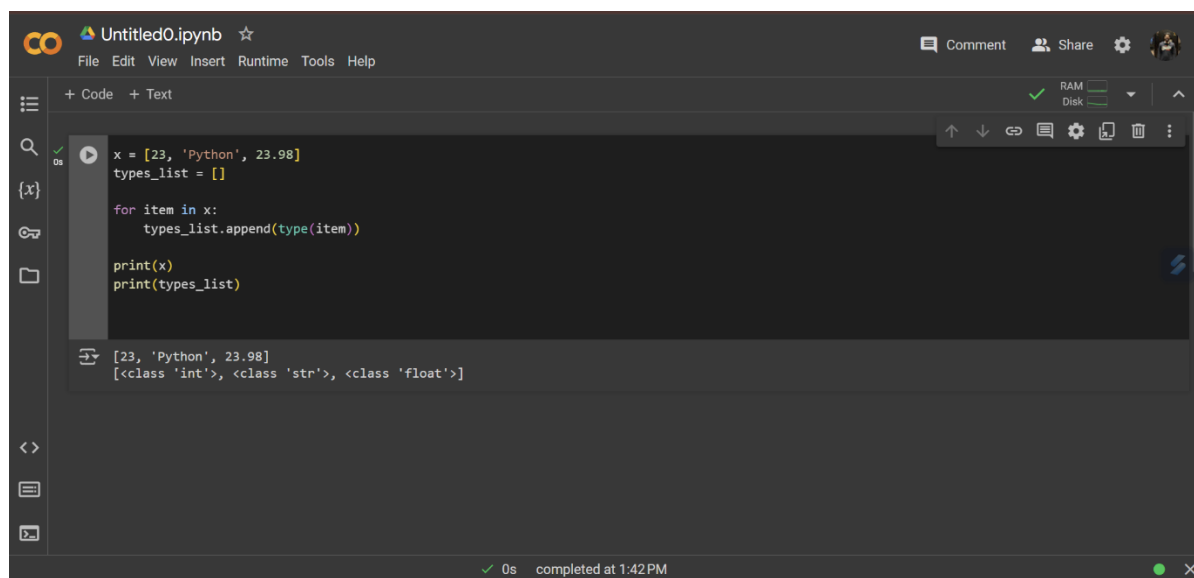
3. Write a code that appends the type of elements from a given list.

Input x = [23, 'Python',

23.98] Expected output

[23, 'Python', 23.98]

[<class 'int'>, <class 'str'>, <class 'float'>]



Untitled0.ipynb

```
File Edit View Insert Runtime Tools Help
```

+ Code + Text

```
x = [23, 'Python', 23.98]
types_list = []

for item in x:
    types_list.append(type(item))

print(x)
print(types_list)
```


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

0s completed at 1:42 PM

4. 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]**



A screenshot of a Jupyter Notebook titled 'Untitled0.ipynb'. The code defines a function `unique_list(input_list)` that iterates through the input list and appends items to a new list only if they are not already present. A sample list `[1, 2, 3, 3, 3, 3, 4, 5]` is used to demonstrate the function. The output shows the sample list and the resulting unique list `[1, 2, 3, 4, 5]`. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for code, text, and execution, and a status bar at the bottom indicating completion at 1:43 PM.

```
def unique_list(input_list):
    unique_items = []
    for item in input_list:
        if item not in unique_items:
            unique_items.append(item)
    return unique_items

# Sample List
sample_list = [1, 2, 3, 3, 3, 3, 4, 5]

# Get unique list
unique_result = unique_list(sample_list)

print("Sample List:", sample_list)
print("Unique List:", unique_result)
```

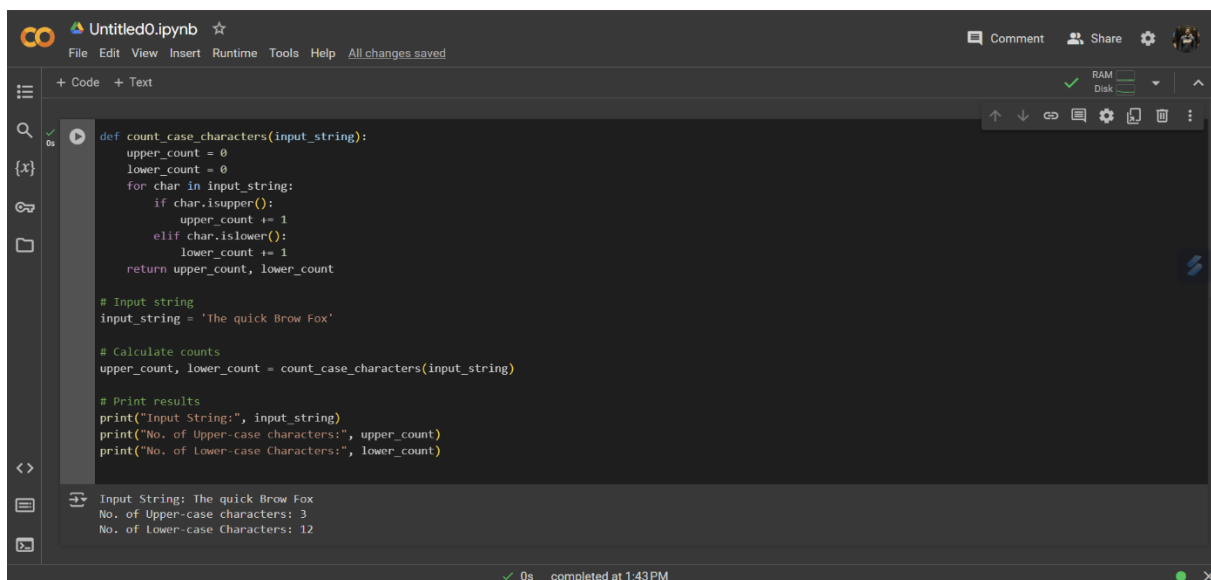
Sample List: [1, 2, 3, 3, 3, 3, 4, 5]  
Unique List: [1, 2, 3, 4, 5]

**5. 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**



A screenshot of a Jupyter Notebook titled 'Untitled0.ipynb'. The code defines a function `count_case_characters(input_string)` that iterates through each character in the input string and increments counters for upper and lower case letters. The input string 'The quick Brow Fox' is used. The output displays the input string and the counts: 3 upper-case characters and 12 lower-case characters. The notebook interface is similar to the first screenshot, with a status bar indicating completion at 1:43 PM.

```
def count_case_characters(input_string):
    upper_count = 0
    lower_count = 0
    for char in input_string:
        if char.isupper():
            upper_count += 1
        elif char.islower():
            lower_count += 1
    return upper_count, lower_count

# Input string
input_string = 'The quick Brow Fox'

# Calculate counts
upper_count, lower_count = count_case_characters(input_string)

# Print results
print("Input String:", input_string)
print("No. of Upper-case characters:", upper_count)
print("No. of Lower-case Characters:", lower_count)
```

Input String: The quick Brow Fox  
No. of Upper-case characters: 3  
No. of Lower-case Characters: 12