

# MACHINE LEARNING ASSIGNMENT 2

NAME: SUPRIYA SAMA  
700744510

Video link:

[https://drive.google.com/file/d/1s2eDRS\\_BCWJ4KqAFXlwlKeo5BpI0dvtr/view?usp=share\\_link](https://drive.google.com/file/d/1s2eDRS_BCWJ4KqAFXlwlKeo5BpI0dvtr/view?usp=share_link)

1. To print given star pattern we use 5 rows and we use nested loop for each column in the range 5. The star symbol is printed in 5 columns then we use end. Next, we use another nested loop to print another star columns.

The screenshot shows a Jupyter Notebook interface in a web browser. The browser address bar displays the URL: `localhost:8888/notebooks/ML%20class%20ass%201.ipynb`. The Jupyter Notebook title is "ML class ass 1". The code cell contains the following Python code:

```
In [8]: #number of rows
rows = 5
#using nested loop for each columns
for i in range(0,rows):
    for j in range(0,i+1):
        print("*",end=' ')
    #printing results
    print("\n")

#using nested loop for each columns
for i in range(rows, 0, -1):
    for j in range(0, i - 1):
        print("*", end=' ')
    #printing results
    print("\n")

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

The output of the code is a pattern of asterisks arranged in a 5x5 grid:

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

- For the given list of elements we can print the odd index elements by looping through the array by incrementing the value of i by 2 starting from the index 1. Then print the results.

The screenshot shows a Jupyter Notebook titled "ML class ass 2" with a last checkpoint 13 minutes ago. The notebook is running on Python 3 (ipykernel). The code in the cell is as follows:

```
In [4]: #given list
my_list = [10,20,30,40,50,60,70,80,90,100]
#Loop through the array by incrementing the value of i by 2
for i in range(1, len(my_list), 2):
#printing results
    print(my_list[i])
```

The output of the code is:

```
20
40
60
80
100
```

3. For the given list we have to create `type_list` to store the elements and we can append the data type of the elements by using `append(type(item))` method. Then print the elements and data types.

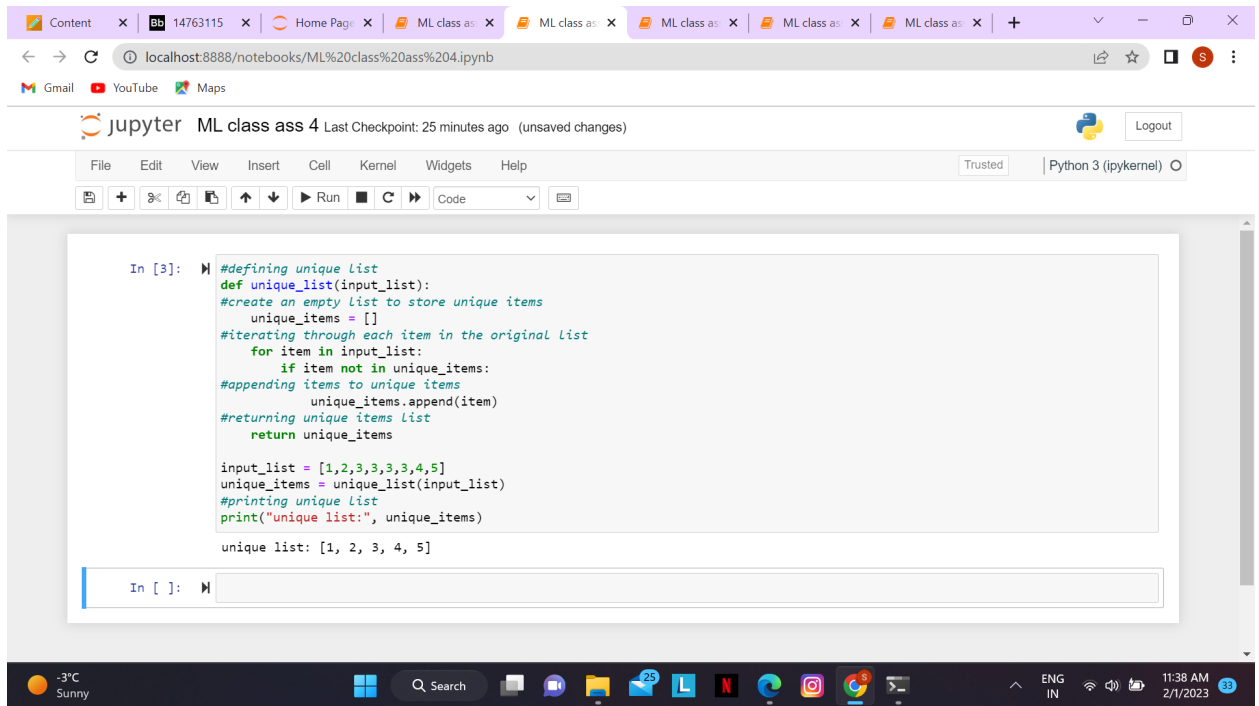
The screenshot shows a Jupyter Notebook titled "ML class ass 3" with a last checkpoint 39 minutes ago. The notebook is running on Python 3 (ipykernel). The code in the cell is as follows:

```
In [2]: #given List
x = [23, 'python', 23.98]
type_list = []
#using for loop to append the type of elements
for item in x:
    type_list.append(type(item))
#printing elements
print("Elements:", x)
#printing data type
print("Types:", type_list)
```

The output of the code is:

```
Elements: [23, 'python', 23.98]
Types: [<class 'int'>, <class 'str'>, <class 'float'>]
```

4. Create an empty list to store unique items then iterate through each item in original list by using for loop. We can append items to unique items by using `append( )` method. Then return the unique items list and print the results.



The screenshot shows a Jupyter Notebook interface in a web browser. The browser's address bar displays `localhost:8888/notebooks/ML%20class%20ass%204.ipynb`. The Jupyter interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for file operations and execution. The notebook title is "ML class ass 4" with a timestamp "Last Checkpoint: 25 minutes ago (unsaved changes)".

The code cell, labeled "In [3]:", contains the following Python code:

```
#defining unique list
def unique_list(input_list):
    #create an empty list to store unique items
    unique_items = []
    #iterating through each item in the original list
    for item in input_list:
        if item not in unique_items:
            #appending items to unique items
            unique_items.append(item)
    #returning unique items list
    return unique_items

input_list = [1,2,3,3,3,3,4,5]
unique_items = unique_list(input_list)
#printing unique list
print("unique list:", unique_items)
```

The output of the code is displayed below the cell: `unique list: [1, 2, 3, 4, 5]`. The bottom of the image shows a Windows taskbar with various application icons and a system tray indicating the temperature is -3°C and the date is 2/1/2023.

5. Given string 'The quick Brow Fox'. Define case count and iterate through each character in the string using for loop. Check if character is uppercase or lowercase and return the values. Then print the results.

Content x 14763115 x Home Page x ML class as x ML class as x ML class as x ML class as x ML class as x +

localhost:8888/notebooks/ML%20class%20ass%205.ipynb

Gmail YouTube Maps

jupyter ML class ass 5 Last Checkpoint: 19 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

In [6]:

```
#defining case count
def case_count(string):
    upper_count = 0
    lower_count = 0
    #iterating through each character in the string
    for char in string:
        #check if character is upper case
        if char.isupper():
            upper_count += 1
        #check if character is lower case
        elif char.islower():
            lower_count += 1
    #return the values
    return (upper_count, lower_count)

#given string
string = 'The quick Brown Fox'
upper_count, lower_count = case_count(string)
#printing 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

-3°C Sunny Search 25 L N 11:39 AM 2/1/2023