

### Assignment -3

#### Python Programming

Assignment Date	7 October 2022
Student Name	Harish Kumar V
Student Roll Number	921319205038
Maximum Marks	2 Marks

Challenge:

To run the python program on google colab.

The screenshot shows a Google Colab notebook with the following content:

```
921319205038_Harish_kumar.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text
RAM 1 GB Disk 10 GB Editing

Exercises
Answer the questions or complete the tasks outlined in bold below, use the specific method described if applicable.

** What is 7 to the power of 4? **

[1] print(pow(7,4))
2401

** Split this string: **
s = 'Hi there Sam!' *into a list. *

[2] s = "Hi there Sam!"

[3] s.split()
['Hi', 'there', 'Sam!']

** Given the variables: **
```

The notebook is running on a virtual machine with 0s of CPU time and completed at 12:58 PM.

The screenshot shows a Google Colab notebook with the following content:

```
921319205038_Harish_kumar.ipynb
File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text
RAM 1 GB Disk 10 GB Editing

** Given the variables: **
planet = "Earth" diameter = 12742 ** Use .format() to print the following string: **
The diameter of Earth is 12742 kilometers.

[5] print("The diameter of {} is {} kilometers".format(planet,diameter))
The diameter of Earth is 12742 kilometers

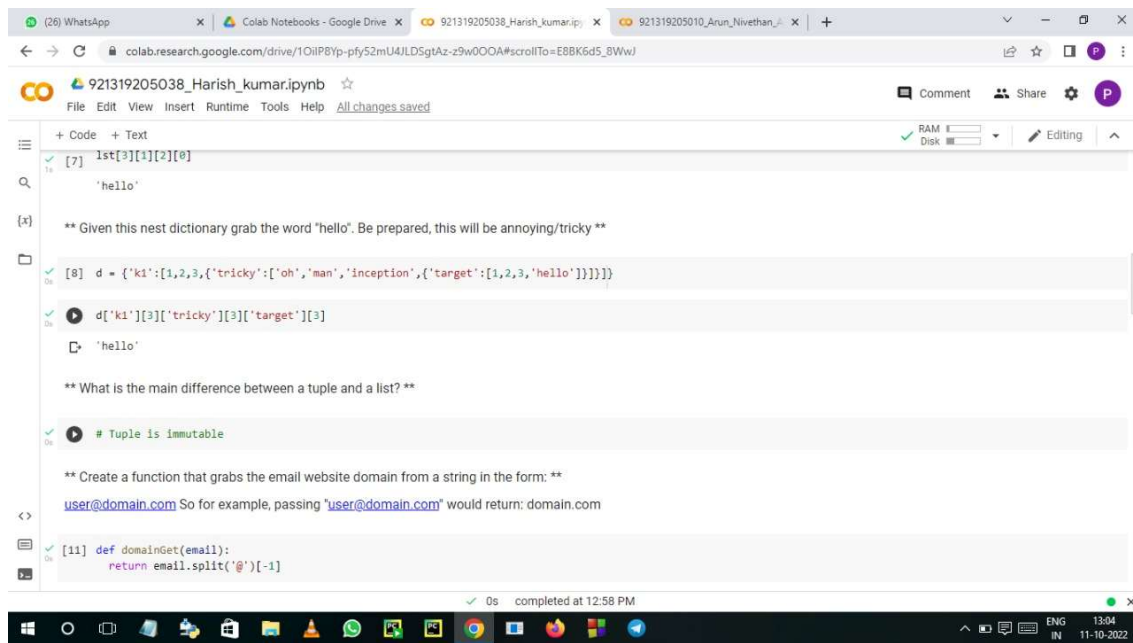
** Given this nested list, use indexing to grab the word "hello" **

[6] lst = [1,2,[3,4],[5,[100,200,['hello']],23,11],1,7]

[7] lst[3][1][2][0]
'hello'

** Given this nest dictionary grab the word "hello". Be prepared, this will be annoying/tricky **
```

The notebook is running on a virtual machine with 0s of CPU time and completed at 12:58 PM.



921319205038\_Harish\_kumar.ipynb

```
[7] lst[3][1][2][0]
'hello'
```

\*\* Given this nest dictionary grab the word "hello". Be prepared, this will be annoying/tricky \*\*

```
[8] d = {'k1':[1,2,3,{'tricky':['oh','man','inception',{'target':[1,2,3,'hello']}]}]}
```

```
[9] d['k1'][3][{'tricky'}[3][{'target'}[3]]
'hello'
```

\*\* What is the main difference between a tuple and a list? \*\*

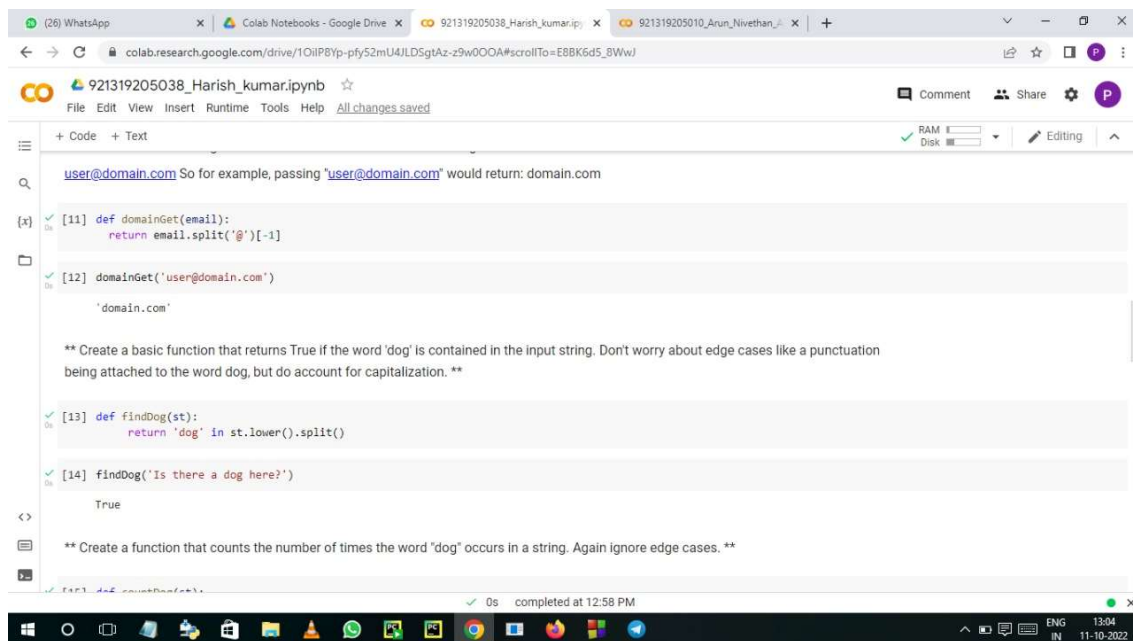
```
[10] # Tuple is immutable
```

\*\* Create a function that grabs the email website domain from a string in the form: \*\*

[user@domain.com](#) So for example, passing "[user@domain.com](#)" would return: domain.com

```
[11] def domainGet(email):
    return email.split('@')[-1]
```

0s completed at 12:58 PM



921319205038\_Harish\_kumar.ipynb

[user@domain.com](#) So for example, passing "[user@domain.com](#)" would return: domain.com

```
[11] def domainGet(email):
    return email.split('@')[-1]
```

```
[12] domainGet('user@domain.com')
'domain.com'
```

\*\* Create a basic function that returns True if the word 'dog' is contained in the input string. Don't worry about edge cases like a punctuation being attached to the word dog, but do account for capitalization. \*\*

```
[13] def findDog(st):
    return 'dog' in st.lower().split()
```

```
[14] findDog('Is there a dog here?')
True
```

\*\* Create a function that counts the number of times the word "dog" occurs in a string. Again ignore edge cases. \*\*

```
[15] def countDog(st):
```

0s completed at 12:58 PM

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include (26) WhatsApp, Colab Notebooks - Google Drive, and two Colab notebooks. The active notebook is titled "921319205038\_Harish\_kumar.ipynb". The menu bar includes File, Edit, View, Insert, Runtime, Tools, and Help. On the right, there are buttons for Comment, Share, and a user profile icon, along with RAM and Disk usage indicators and an "Editing" mode button.

The notebook content includes a code cell with the following Python code:

```
True

** Create a function that counts the number of times the word "dog" occurs in a string. Again ignore edge cases. **

[15] def countDog(st):
    count = 0
    for word in st.lower().split():
        if word == 'dog':
            count += 1
    return count

countDog('This dog runs faster than the other dog dude!')
```

The output of the code cell is the number 2.

Below the code cell is a "Problem" section with the following text:

\*You are driving a little too fast, and a police officer stops you. Write a function to return one of 3 possible results: "No ticket", "Small ticket", or "Big Ticket". If your speed is 60 or less, the result is "No Ticket". If speed is between 61 and 80 inclusive, the result is "Small Ticket". If speed is 81 or more, the result is "Big Ticket". Unless it is your birthday (encoded as a boolean value in the parameters of the function) – on your birthday, your speed can be 5 higher in all cases.\*

The bottom status bar shows "0s completed at 12:58 PM" and a Windows taskbar at the very bottom with various application icons and the system clock showing 13:04 on 11-10-2022.

This screenshot shows the same Google Colab notebook interface as the first one. The code cell now contains a function to determine if a driver should get a ticket based on their speed and whether it's their birthday.

```
def caught_speeding(speed, is_birthday):

    if is_birthday:
        speeding = speed - 5
    else:
        speeding = speed

    if speeding > 80:
        return 'Big Ticket'
    elif speeding > 60:
        return 'Small Ticket'
    else:
        return 'No Ticket'
```

Below the code cell, two test cases are shown:

```
[18] caught_speeding(81, False)
'Big Ticket'

[19] caught_speeding(81, True)
```

The bottom status bar shows "0s completed at 12:58 PM" and the same Windows taskbar at the bottom.

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include '26 WhatsApp', 'Colab Notebooks - Google Drive', and two Colab notebooks. The active notebook is titled '921319205038\_Harish\_kumar.ipynb'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. On the right, there are buttons for 'Comment', 'Share', and a settings icon, along with a RAM/Disk usage indicator and an 'Editing' mode button. The left sidebar contains icons for file management. The main code area has a 'Code' tab selected. The code includes a function call `caught_speeding(81, True)` which returns the string `'Small Ticket'`. Below this, a task instruction reads: 'Create an employee list with basic salary values(at least 5 values for 5 employees) and using a for loop retrieve each employee salary and calculate total salary expenditure.' The code then defines a list `employee=[400,500,550,600,250]` and uses a `for` loop to print individual salaries and calculate a total sum of 2300. The output of the code execution is displayed below the code cells, showing the same salary values and the total sum. At the bottom, a status bar indicates '0s completed at 12:58 PM'.

```
[19] caught_speeding(81, True)

'Small Ticket'

Create an employee list with basic salary values(at least 5 values for 5 employees) and using a for loop retrieve each employee salary and calculate total salary expenditure.

employee=[400,500,550,600,250]
sum=0
print ("salaryof 1st person is",employee[0])
print ("salaryof 2nd person is",employee[1])
print ("salaryof 3rd person is",employee[2])
print ("salaryof 4th person is",employee[3])
print ("salaryof 5th person is",employee[4])
for x in employee:
    sum=sum+x
print("The total salary is", sum)

salaryof 1st person is 400
salaryof 2nd person is 500
salaryof 3rd person is 550
salaryof 4th person is 600
salaryof 5th person is 250
The total salary is 2300

0s completed at 12:58 PM
```

The screenshot shows a Google Colab notebook interface. The browser tabs at the top include '26 WhatsApp', 'Colab Notebooks - Google Drive', and two Colab notebooks. The active notebook is titled '921319205038\_Harish\_kumar.ipynb'. The menu bar includes 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. On the right, there are buttons for 'Comment', 'Share', and a settings icon, along with a RAM/Disk usage indicator and an 'Editing' mode button. The left sidebar contains icons for file management. The main code area has a 'Code' tab selected. The code includes a task instruction: 'Create two dictionaries in Python: First one to contain fields as EmpId, Empname, Basicpay. Second dictionary to contain fields as DeptName, DeptId. Combine both dictionaries.' The code then defines two dictionaries, `d1` and `d2`, and prints their combined output. The output of the code execution is displayed below the code cells, showing the combined dictionary with fields for EmpId, Empname, Basicpay, deptname, and DEPTID. At the bottom, a status bar indicates '0s completed at 12:58 PM'.

```
[21] d1 = { "EmpId":9213,"Empname":"MaxAdam","Basicpay": 80000}
d2 = {"deptname":"Software Engineering", "DEPTID": '205'}
print(**d1, **d2)

{'EmpId': 9213, 'Empname': 'MaxAdam', 'Basicpay': 80000, 'deptname': 'Software Engineering', 'DEPTID': '205'}

0s completed at 12:58 PM
```