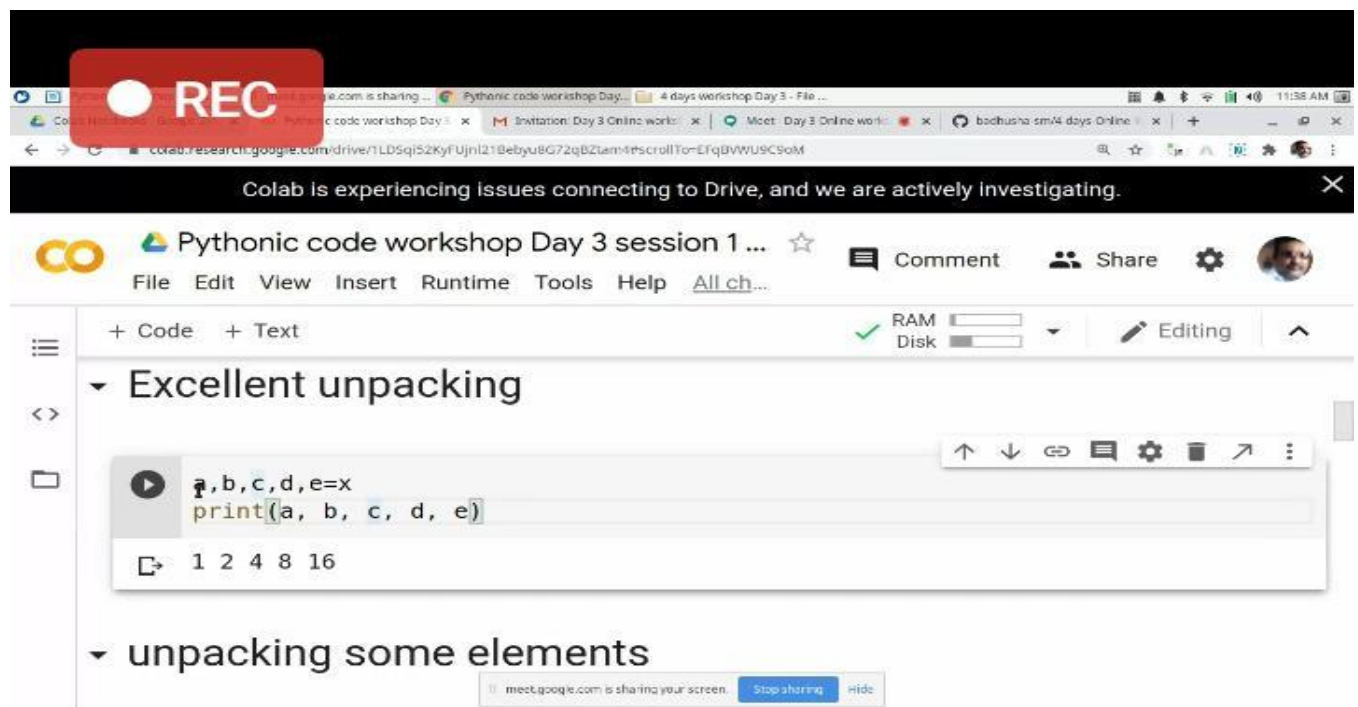
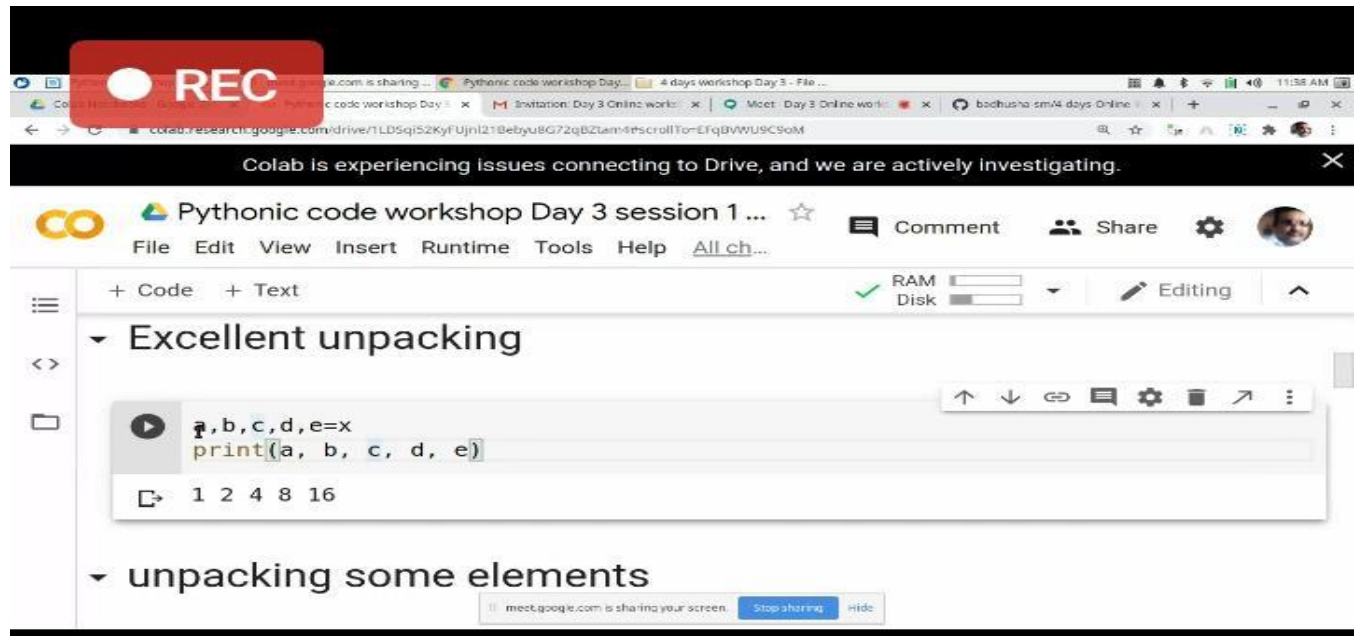


DAILY ASSESSMENT FORMAT

Date:	23/07/2020	Name:	Akshatha M Deshpande
Course:	Python	USN:	4AL17EC006
Topic:	Day 3 pythonic workshop	Semester & Section:	6th Sem A sec
Github Repository:	AkshathaDeshpande		

FORENOON SESSION DETAILS

Image of session



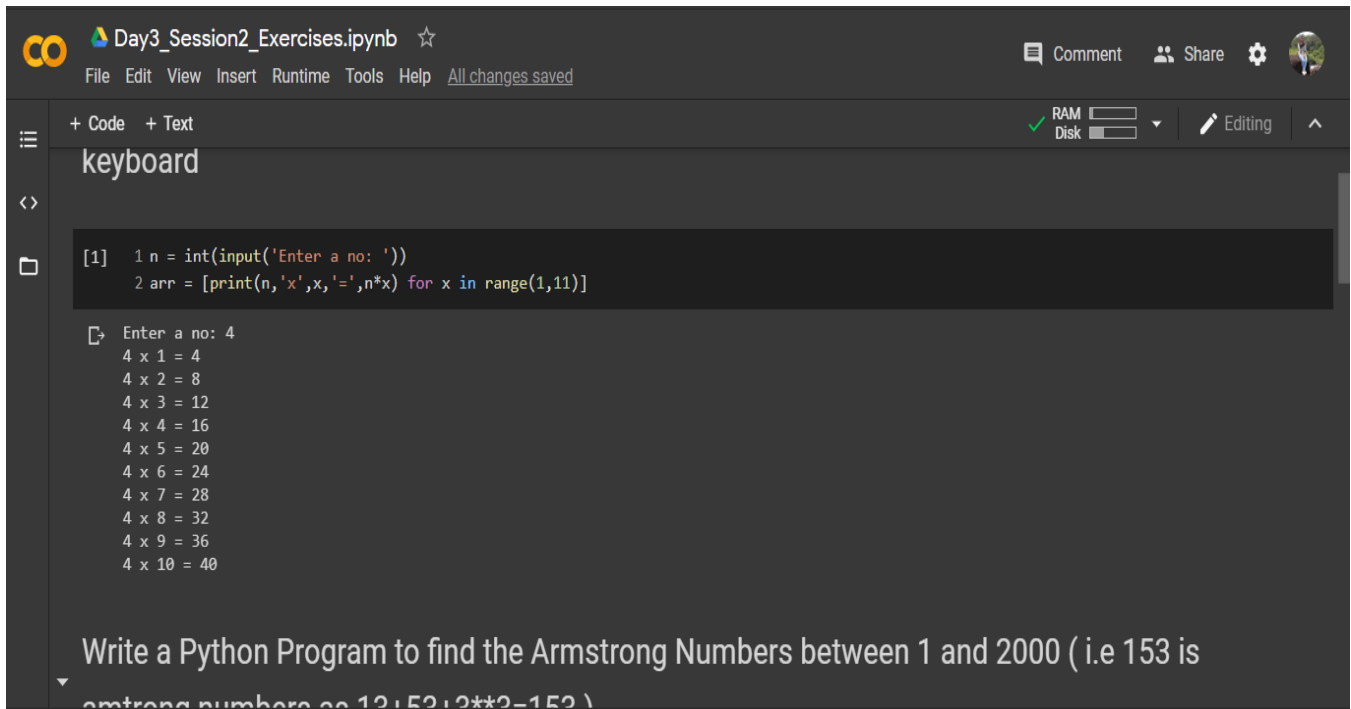
Report – Report can be typed or hand written for up to two pages.

Python Day-3:

- Python scripts can put the system into different states, set configurations, and test all sorts of real-world use cases.
- Python can also be used to receive embedded system data that can be stored for analysis.
- Programmers can then use Python to develop parameters and other methods of analyzing that data.
- A tuple is a collection of objects which ordered and immutable. Tuples are sequences, just like lists.
- The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.
- Tuples are immutable which means you cannot update or change the values of tuple elements.
- You are able to take portions of existing tuples to create new tuples
- The most basic data structure in Python is the sequence. Each element of a sequence is assigned a number - its position or index.
- The first index is zero, the second index is one, and so forth. Python has six built-in types of sequences, but the most common ones are lists and tuples, which we would see in this tutorial.
- There are certain things you can do with all sequence types.
- These operations include indexing, slicing, adding, multiplying, and checking for membership. In addition, Python has built-in functions for finding the length of a sequence and for finding its largest and smallest elements.
- You're saving loads of time writing humongous piles of coddung code, so you're obviously becoming a smarter and more productive programmer.
- Python is a pretty slow language, and when you're trying to do something in Python, which is acquired from another language like Java or C++, you're going to worsen things.
- With idiomatic, Pythonic code, you're improving the speed of your programs. Moreover, idiomatic code is far easier to comprehend and understand for other developers who are working on the same code.
- It helps a great deal when you're trying to refactor someone else's code.
- Rather, quite a few developers and organisations have begun discriminating on the basis of whether someone can or cannot write Pythonic code.
- This is wrong, because, at the end of the day, though the PEP 8 exists, the idea of

the term Pythonic is different for different people.

- To some it might mean picking up a new style guide and improving the way you code. To others, it might mean being succinct and not repeating themselves.

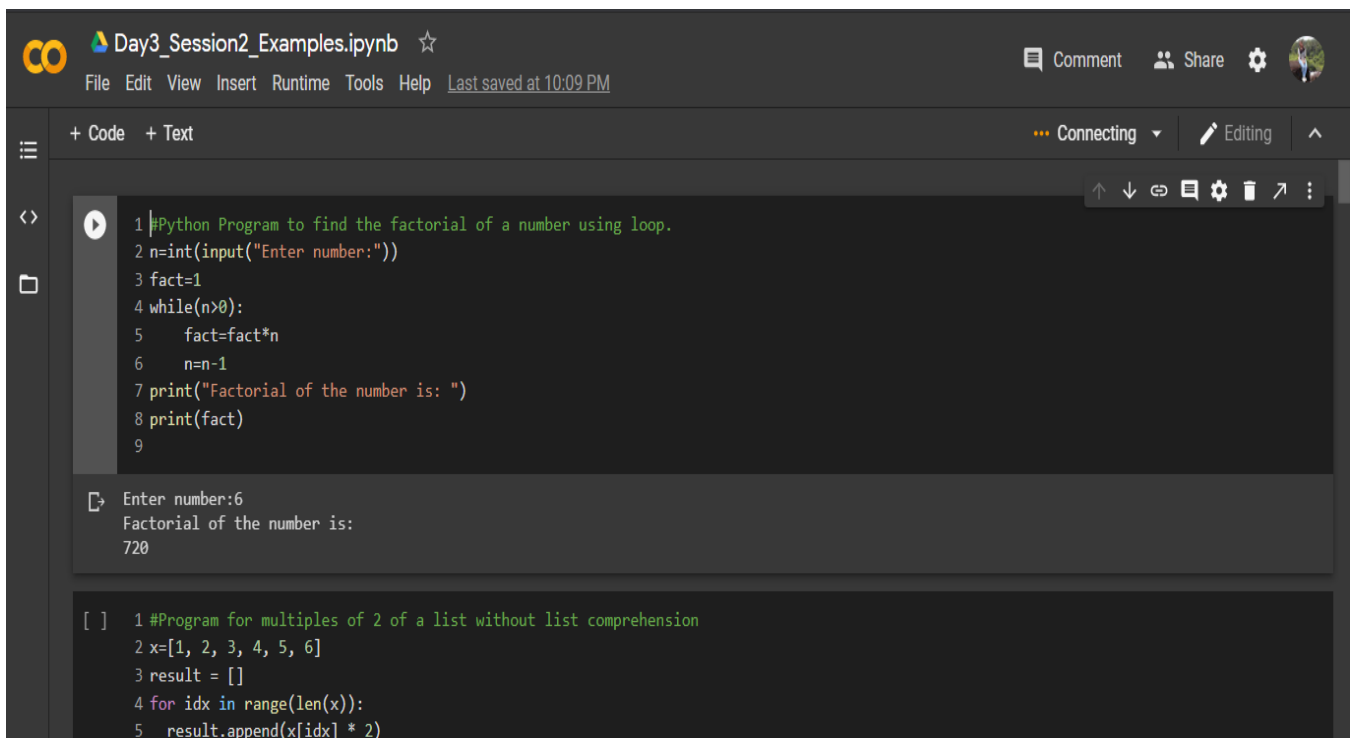


The screenshot shows a Jupyter Notebook interface with the title 'Day3_Session2_Exercises.ipynb'. The code cell contains a program that takes an input 'n' and prints a multiplication table for 'n' from 1 to 10. The output shows the input '4' and a 4x10 multiplication table. Below the code cell, there is a text prompt: 'Write a Python Program to find the Armstrong Numbers between 1 and 2000 (i.e 153 is an armstrong number as 1³+5³+3³=153)'.

```
[1] 1 n = int(input('Enter a no: '))
    2 arr = [print(n,'x',x,'=',n*x) for x in range(1,11)]

Enter a no: 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

Write a Python Program to find the Armstrong Numbers between 1 and 2000 (i.e 153 is an armstrong number as 1³+5³+3³=153)



The screenshot shows a Jupyter Notebook interface with the title 'Day3_Session2_Examples.ipynb'. The first code cell contains a program to calculate the factorial of a number using a while loop. The output shows the input '6' and the factorial '720'. The second code cell contains a program to find multiples of 2 in a list without using list comprehension. The output shows the list [1, 2, 3, 4, 5, 6] and the resulting list of multiples [2, 4, 6].

```
1 #Python Program to find the factorial of a number using loop.
2 n=int(input("Enter number:"))
3 fact=1
4 while(n>0):
5     fact=fact*n
6     n=n-1
7 print("Factorial of the number is: ")
8 print(fact)
9

Enter number:6
Factorial of the number is:
720

[ ] 1 #Program for multiples of 2 of a list without list comprehension
    2 x=[1, 2, 3, 4, 5, 6]
    3 result = []
    4 for idx in range(len(x)):
    5     result.append(x[idx] * 2)
```