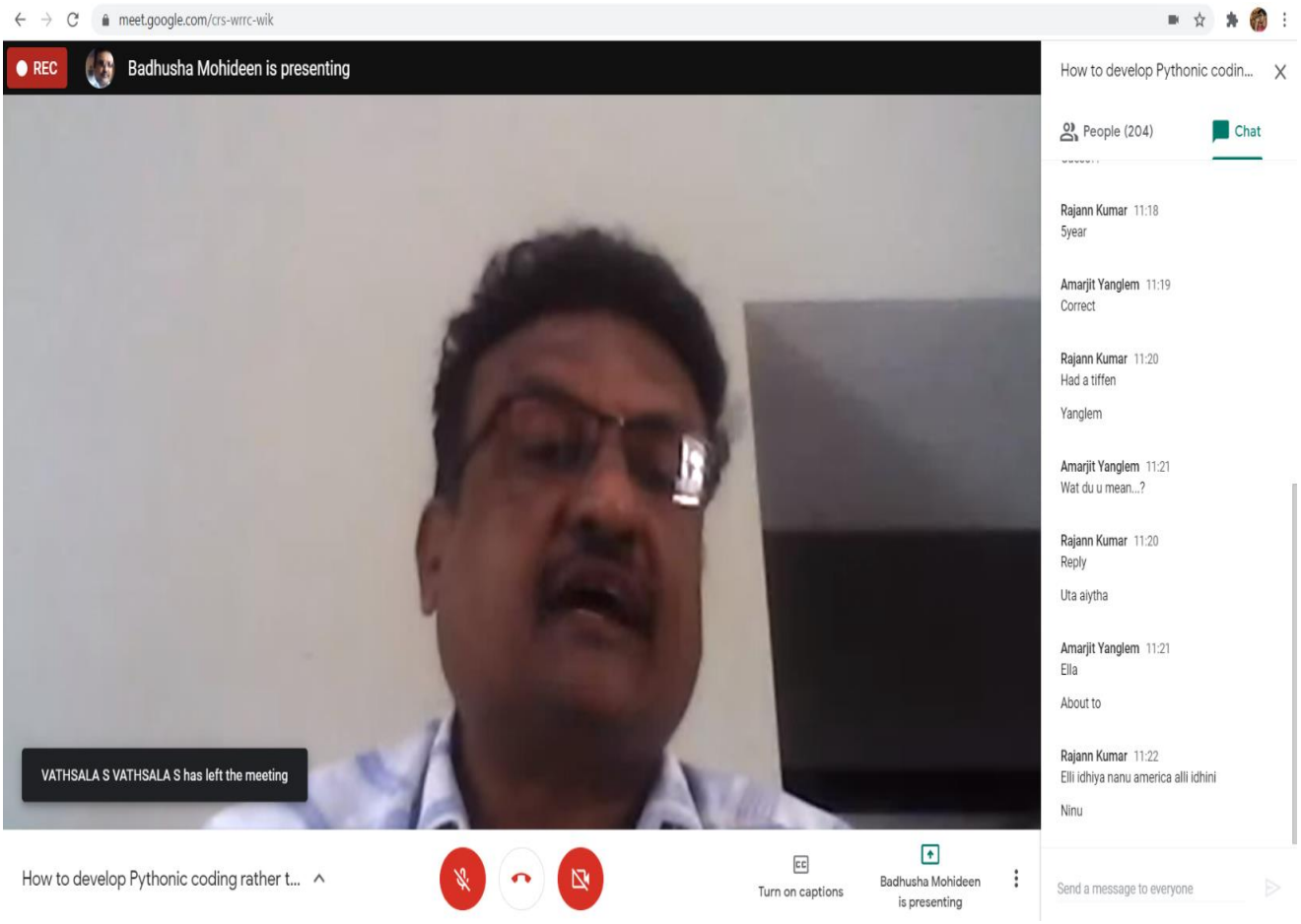
DAILY ASSESSMENT REPORT

Date:	**23 July 2020**	**Name:**	**Gagan M K**
Course:	**How to develop Pythonic coding – Logic Perspective**	**USN:**	**4AL17EC032**
Topic:	• **Day 3**	**Semester & Section:**	**6th sem & 'A' sec**
GitHub Repository:	**Alvas-education-foundation/Gagan-Git**		

FORENOON SESSION DETAILS

Image of session



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

Python:

Finding the factorial of a number

In [1]:
```
n=int(input("Enter number:"))
fact=1
while(n>0):
    fact=fact*n
    n=n-1
print("Factorial of the number is: ")
print(fact)
```

```
Enter number:5
Factorial of the number is:
120
```

Program for multiples of 2 of a list without list comprehension

In [2]:
```
[2, 4, 6, 8, 10, 12]
X=[1, 2, 3, 4, 5, 6]
result = []
for idx in range(len(x)):
  result.append(x[idx] * 2)
print(result)
```

```
[2, 4, 6, 8, 10, 12]
```

In [3]:
```
#Single line program of Pythonic coding
print([i*2 for i in [1, 2, 3, 4, 5, 6] ])
```

```
[2, 4, 6, 8, 10, 12]
```

In [4]:
```
#More Pythonic program of 3 lines!
from functools import reduce
sequences = [x for x in range(1,int(input('Enter no'))+1)]
product = reduce(lambda x, y: x*y, sequences)
print(product)
```

```
Enter no5
120
```

Number and its square as Tuple for a range

In [5]:
```
l_range=int(input("Enter the lower range:"))
u_range=int(input("Enter the upper range:"))
a=[(x,x**2) for x in range(l_range,u_range+1)]
print(a)
```

```
Enter the lower range:2
Enter the upper range:8
[(2, 4), (3, 9), (4, 16), (5, 25), (6, 36), (7, 49), (8, 64)]
```

Finding the perfect squares

In [6]:
```
from math import *
sequences = [10,2,8,7,5,4,3,11,0,9,16,1]
result=[]
for i in sequences:
  if int(sqrt(i))**2==i:
    result.append(i)
print(result)
```

```
[4, 0, 9, 16, 1]
```

Program to create a list of tuples with the rst element as the number and the second

element as the square of the number.

In [10]:
```
l_range=int(input("Enter the lower range:"))
u_range=int(input("Enter the upper range:"))
a=[(x,x**2) for x in range(l_range,u_range+1)]
print(a)
```

```
Enter the lower range:2
Enter the upper range:5
[(2, 4), (3, 9), (4, 16), (5, 25)]
```

In [11]:
```
#The aforementioned program is already pythonic.
a=[(x,x**2) for x in range(int(input("Enter the lower range:")),\
                           int(input("Enter the upper range:"))+1)]
print(a)
```

```
Enter the lower range:2
Enter the upper range:5
[(2, 4), (3, 9), (4, 16), (5, 25)]
```

In [12]:
```
# We can write in the most pythonic way with one line ! as follows
print([(x,x**2) for x in range(int(input("Enter the lower range:")),\
                           int(input("Enter the upper range:"))+1)])
```

```
Enter the lower range:2
Enter the upper range:5
[(2, 4), (3, 9), (4, 16), (5, 25)]
```

Program to generate random numbers from 1 to 20 and append them to the list.

In [13]:
```
import random
a=[]
n=int(input("Enter number of elements:"))
for j in range(n):
    a.append(random.randint(1,20))
print('Randomised list is: ',a)
```

```
Enter number of elements:3
Randomised list is:  [8, 15, 12]
```

In [14]:
```
#Pythonic program of 2 lines!
import random
l=[random.randint(1,20) for _ in range(int(input("Enter how many elements")))]
print('Randomised list is: ',l)
```

```
Enter how many elements3
Randomised list is:  [6, 16, 4]
```

Program for printing list of values with indexing

In [15]:
```
names = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
index=0
for i in names:
    print(str(index)+":"+i)
    index=index+1
```

```
0:Jan
1:Feb
2:Mar
3:Apr
4:May
5:Jun
6:Jul
7:Aug
8:Sep
9:Oct
10:Nov
11:Dec
```

using Pythonic way using enumerate()!

In [16]:
```
names = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
for index, value in enumerate(names):
    print(f'{index+1}: {value}')
```

```
1: Jan
2: Feb
3: Mar
```

- **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.**

Program for prime number -bad code

In [1]:
```
i=25
for x in range(2, i//2+1):
    if i%x==0:
        print("The number {} is not prime".format(i))
        break
if x ==i//2:
   print ("{} is a prime number".format(i))
```

```
The number 25 is not prime
```

Good code using for... else

In [2]:
```
i=25
for x in range(2, i//2+1):
    if i%x==0:
        print("The number {} is not prime".format(i))
        break
else:
    print ("{} is a prime number".format(i))
```

```
The number 25 is not prime
```

Using Unpacking to Write Concise Code

In [3]:
```
a, b = 2, 'my-string'
print(a)
print(b)
```

```
2
my-string
```

Bad unpacking

In [4]:
```
x = (1, 2, 4, 8, 16)
a = x[0]
b = x[1]
c = x[2]
d = x[3]
e = x[4]
print(a, b, c, d, e)
```

```
1 2 4 8 16
```

Excellent unpacking

In [5]:
```
a,b,c,d,e=x
print(a, b, c, d, e)
```

```
1 2 4 8 16
```

unpacking some elements

In [ ]:
```
a, *y, c = x
print(a)
print(y)
print(c)
```

```
1
[2, 4, 8]
16
```