**DAY3-Programming fundamentals using python**

Assignment 22:

Write a Python program to generate the next 15 leap years starting from a given year. Populate the leap years into a list and display the list.  
Also write the pytest test cases to test the program.

#PF-Assgn-22

def find\_leap\_years(given\_year):

# Write your logic here

count=0

list\_of\_leap\_years=[]

while(count<15):

if(given\_year%4==0 or given\_year%400==0 and given\_year%100==0):

list\_of\_leap\_years.append(given\_year)

count=count+1

given\_year=given\_year+1

return list\_of\_leap\_years

list\_of\_leap\_years=find\_leap\_years(2000)

print(list\_of\_leap\_years)

Assignment 23:

ARS Gems Store sells different varieties of gems to its customers.  
  
Write a Python program to calculate the bill amount to be paid by a customer based on the list of gems and quantity purchased. Any purchase with a total bill amount above Rs.30000 is entitled for 5% discount. If any gem required by the customer is not available in the store, then consider total bill amount to be -1.  
  
Assume that quantity required by the customer for any gem will always be greater than 0.  
  
Perform case-sensitive comparison wherever applicable.

#PF-Assgn-23

def calculate\_bill\_amount(gems\_list, price\_list, reqd\_gems,reqd\_quantity):

bill\_amount=0

#Write your logic here

for i in range(len(reqd\_gems)):

if reqd\_gems[i] in gems\_list:

for j in range(len(gems\_list)):

if reqd\_gems[i] == gems\_list[j]:

bill\_amount+=price\_list[j]\* reqd\_quantity[i]

continue

else :

bill\_amount = -1

break

if bill\_amount >30000:

bill\_amount= bill\_amount \* 95/100

return bill\_amount

#List of gems available in the store

gems\_list=["Emerald","Ivory","Jasper","Ruby","Garnet"]

#Price of gems available in the store. gems\_list and price\_list have one-to-one correspondence

price\_list=[1760,2119,1599,3920,3999]

#List of gems required by the customer

reqd\_gems=["Ivory","Emerald","Garnet"]

#Quantity of gems required by the customer. reqd\_gems and reqd\_quantity have one-to-one correspondence

reqd\_quantity=[3,10,12]

bill\_amount=calculate\_bill\_amount(gems\_list, price\_list, reqd\_gems, reqd\_quantity)

print(bill\_amount)

Assignment 24:

Write a python function to check whether three given numbers can form the sides of a triangle.  
**Hint**  
: Three numbers can be the sides of a triangle if none of the numbers are greater than or equal to the sum of the other two numbers.

#PF-Assgn-24

def form\_triangle(num1,num2,num3):

#Do not change the messages provided below

success="Triangle can be formed"

failure="Triangle can't be formed"

#Write your logic here

if num1 < num2 + num3 and num2 < num1 + num3 and num3< num1 + num2:

print( success)

else:

print(failure)

#Use the following messages to return the result wherever necessary

#Provide different values for the variables, num1, num2, num3 and test your program

num1=3

num2=3

num3=5

form\_triangle(num1, num2, num3)

Assignment 25:

The program provided in the starter code tab is written to display “\*” as per the expected output given below. But the code is having logical errors, debug the program using Eclipse Debugger and correct it.

**Estimated Time: 15 minutes**

**Expected Output:**

\*\*\*\*\*

\*\*\*\*

\*\*\*

\*\*

\*

#PF-Tryout  
  
#debug the below code  
counter1=0  
counter2=5  
while(counter1 < 5):  
 star=""  
 while(counter2>counter1):  
 star=star+ "\*"  
 counter2-=1  
 print(star)  
 counter1+=1

Assignment 26:

Write a python program to solve a classic ancient Chinese puzzle.

We count 35 heads and 94 legs among the chickens and rabbits in a farm. How many rabbits and how many chickens do we have?

#PF-Assgn-26

def solve(heads,legs):

error\_msg="No solution"

chicken\_count=0

rabbit\_count=0

#Start writing your code here

#Populate the variables: chicken\_count and rabbit\_count

for chicken\_count in range(heads+1):

rabbit\_count = heads - chicken\_count

if (2\*chicken\_count)+(4\*rabbit\_count)==legs:

print(chicken\_count,rabbit\_count)

if chicken\_count == 0 and rabbit\_count == 0 or heads > legs:

print(error\_msg)

# Use the below given print statements to display the output

# Also, do not modify them for verification to work

#print(chicken\_count,rabbit\_count)

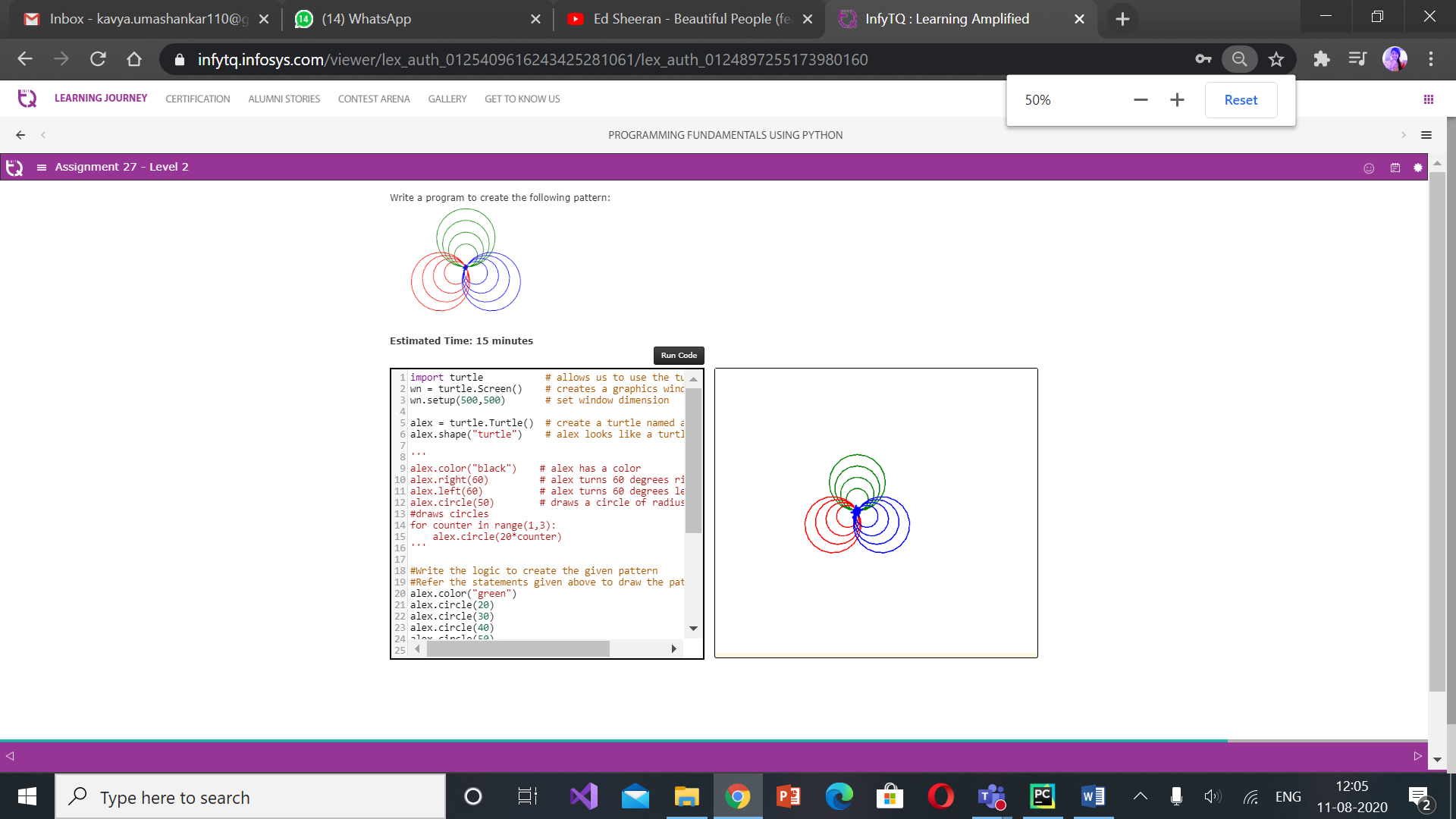
#print(error\_msg)

#Provide different values for heads and legs and test your program

solve(38,131)

Assignment 27:

Write a program to create the following pattern:



Assignment 28:

Write a python program which finds the maximum number from num1 to num2 (num2 inclusive) based on the following rules.

1. Always num1 should be less than num2
2. Consider each number from num1 to num2 (num2 inclusive). Populate the number into a list, if the below conditions are satisfied
   1. Sum of the digits of the number is a multiple of 3
   2. Number has only two digits
   3. Number is a multiple of 5
3. Display the maximum element from the list

In case of any invalid data or if the list is empty, display -1.

#PF-Assgn-28

def find\_max(num1, num2):

max\_num=-1

# Write your logic here

list1=[]

if num1 <= 99 and num1 >=10 and num2<=99 and num2 >=10 and num1 <num2:

for i in range(num1,num2+1):

if i >=10 and i<=99:

if i%5 == 0:

sum\_of\_digits = 0

for digit in str(i):

sum\_of\_digits += int(digit)

if sum\_of\_digits %3 == 0:

list1+=[sum\_of\_digits]

for i in list1:

if i > max\_num :

max\_num = i

return max\_num

#Provide different values for num1 and num2 and test your program.

max\_num=find\_max(10,15)

print(max\_num)