UNIVERSITY OF PETROLEUM AND ENERGY STUDIES

PYTHON PROGRAMMING

1st Year

Semester 2

Session 2023-24

Submitted By: Jiya Tyagi

Submitted To: Ms. Sugandha Sharma

Batch:43

SAP ID: 500119743

ROLL NO. : R2142231454

Experiment 1: Python Installation And Starting With Python

Experiment 2: Use of input statements, operators

1. Declare these variables (x, y and z) as integers. Assign a value of 9 to x, Assign a value of 7 to y, perform addition, multiplication, division and subtraction on these two variables and Print out the result

CODE:

```
#1

x = int(9)
y = int(7)

z = int(x+y)
print('The sum is :')
print(x + y)

z = int (x-y)
print('The difference is :')
print(z)

z = int (x*y)
print("The value of multiplication is :")
print(z)

z = int (x/y)
print('The value of division is :')
print(z)
```

OUTPUT :

```
In [7]: runfile('C:/Users/DELL/.spyder-py3/Lab_2_Python.py', wdir='C:/
Users/DELL/.spyder-py3')
The sum is :
16
The difference is :
2
The value of multiplication is :
63
The value of division is :
1
```

2. Write a Program where the radius is taken as input to compute the area of a circle.

CODE:

```python

#2
radius = int( input('Enter the radius of circle : '))
pi = float (3.14)
area = pi*radius*radius
print('The area of the circle is :')
print(area)
```

OUTPUT:

```
In [11]: runfile('C:/Users/DELL/.spyder-py3/Lab_2_Python.py', wdir='C:/
Users/DELL/.spyder-py3')
Enter the radius of circle : 10
The area of the circle is :
314.0

In [12]: runfile('C:/Users/DELL/.spyder-py3/Lab_2_Python.py', wdir='C:/
Users/DELL/.spyder-py3')
Enter the radius of circle : 21
The area of the circle is :
1384.74
```

3. Write a Python program to solve (x+y)*(x+y)

1. CODE :

```python


#3
x = int (input ('Enter value of x'))
y = int (input('Enter value of y'))
solution = float ((x+y)*(x+y))
print('the solution is :')
print(solution)
```

OUTPUT:

```
In [13]: runfile('C:/Users/DELL/.spyder-py3/Lab_2_Python.py', wdir='C:/
Users/DELL/.spyder-py3')
Enter value of x 6
Enter value of y 7
the solution is :
169.0
```

4. Write a program to compute the length of the hypotenuse (c) of a right triangle using Pythagoras theorem.

```python
#4
from math import sqrt
B = int(input('enter the length of base'))
R = int(input('Enter the length of perpendicular'))
H =  sqrt((B*B)+(R*R))
print("the value of hypotenuse is : ")
print(H)
```

```
In [6]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
enter the length of base 3
Enter the length of perpendicular 4
the value of hypotenuse is :
5.0
```

5. Write a program to find simple interest.

```python

#5
'''
#program to find simple interest
P = int(input('Enter the Principal Amount'))
R = int(input('Enter the Rate'))
T = int(input('Enter the Time Period'))
SI = P*R*T/100
print("The Simple interest is :")
print(SI)
'''
```

```
In [7]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
Enter the Principal Amount 10000
Enter the Rate 2
Enter the Time Period 3
The Simple interest is :
600.0
```

6. Write a program to find area of triangle when length of sides are given.

```
#6
'''
# Area of triangle
Height = 6
Base = 3
Area = 1/2*Base*Height
print("The Area of triangle is :",Area)
'''
```

```
In [8]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
The Area of triangle is : 9.0
```

7. Write a program to convert given seconds into hours, minutes and remaining seconds.

```
#7
'''
#convert seconds into hours,minutes and seconds.
Sec = int(input('enter the seconds : '))
Hours = Sec // 3600
Min = (Sec%3600)//60
Remaining_sec=Sec%60
print("The time is:",Hours,"hours",Min,"minutes and",Remaining_sec,"seconds")
'''
```

```
In [9]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
enter the seconds : 4567
The time is: 1 hours 16 minutes and 7 seconds
```

8. Write a program to swap two numbers without taking additional variable.

```python
#8

#swapping two numbers
a=int(input("enter first number:"))
b=int(input("enter second number:"))
a=a+b
b=a-b
a=a-b
print("after swapping the first number is:",a)
print("after swapping the second number is:",b)
'''

```

```
In [11]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
enter first number: 6
enter second number:7
after swapping the first number is: 7
after swapping the second number is: 6
```

9. Write a program to find sum of first n natural numbers

```python

#9
'''
# find sum of first n natural numbers.
n=int(input("enter the number:"))
sum=n*(n+1)//2
print("the sum of first",n,"natural numbers is:",sum)
'''
```

```
In [12]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_2_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
enter the number: 65
the sum of first 65 natural numbers is: 2145

In [13]:
```

Experiment 3 : Conditional Statements

 # 1. Check whether given number is divisible by 3 and 5 both.

```
could make it even more secure.

In [32]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_3_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
Enter the Number 225
divisible by both 3 & 5

In [33]: runfile('E:/1st YEAR/2nd SEMESTER/python/
LABS/Lab_3_Python.py', wdir='E:/1st YEAR/2nd
SEMESTER/python/LABS')
Enter the Number 55
Number not divisible by both
```

'''num = int(input("Enter the number")) if(num%3==0.0 and

num%5==0.0):

print("number is divisible") else: print("number is not divisible")

'''

 # 2. Check whether a given number is multiple of five or not.

'''num = int(input("enter any number")) if(num%5==0):

print("the number is s multiple of 5")

else: print("the number is not a multiple

of 5")

'''

 # 3. find the greatest among two numbers

'''num1=int(input("enter the first number"))

num2=int(input("enter the second number")) if(num1>num2):

print(num1,'is greater') elif(num2>num1):

print(num2,'is greater') else:

print("both numbers are equal")'''

 # 4. Find the greatest among three numbers assuming no two values are same.

'''num1 = int(input("Enter the first number "))

num2 = int(input("Enter the second number"))

num3 = int(input("Enter the third number "))

if(num1 > num2 and num1 > num3):

print("first number is greater among all")

elif(num2 > num1 and num1 > num3):

print("second number is greater among all")
else: print("third number is greater

among all")

'''

 # 5. 4.Find the greatest among three numbers assuming no two values are same.

'''num1 = int(input("Enter the first number "))

num2 = int(input("Enter the second number"))

num3 = int(input("Enter the third number "))

if(num1 > num2 and num1 > num3):

print("first number is greater among all")

elif(num2 > num1 and num1 > num3):

```
print("second number is greater among all")
else:  print("third number is greater among
all")
'''
```

 # 6. Find whether a given year is a leap year or not.

```
'''year = int(input("Enter the year")) if(year % 4 == 0 and
year % 100 != 0) or (year % 400 == 0):  print("year is leap
year") else:  print("year is not a leap year")
'''
```

 # 7. Write a program which takes any date as input and display next date of the calendar e.g. I/P: day=20 month=9 year=2005 O/P: day=21 month=9 year 2005

```
'''day = int (input ("day :"))
month = int (input ("Month : "))
year = int (input ("Year : "))
print ("\n\nThis date : ") print
("\n\n") print ("Day : ",day)
print ("Month : ",month) print
("Year : ",year) if (month in
(1,3,5,7,8,10,12)) :  if (day <31
):  nday = day +1 ;
nmonth = month ;
 nyear = year ;
else :

 if (month ==12):  nday =
1 ;  nmonth = 1 ;  nyear =
year + 1 ;  else :  nday =
1;  nmonth = month +1 ;
nyear =year ; elif (month
in (4,6,9,11)):  if (day <
30):  nday = day + 1 ;
nmonth = month ;  nyear
= year ;  else :  nday =1 ;
nmonth = month +1 ;
```

```
nyear = year ; elif (month
==2):  if (year %4 == 0 ):
if (day < 29):  nday = day
+1 ;  nmonth = month ;
nyear = year ;  else :
nday = 1 ;  nmonth =
month +1 ;  nyear = year
;  else :  if (day <28):
nday = day +1 ;  nmonth
= month ; nyaer = year ;
 else :  nday = 1 ;
nmonth = month + 1 ;
nyear = year ; else :
 print ("Invalid Input"); print ("\n\n\n\n::::::THE
NEXT DATE IS ::::::\n") print ("Day : ",nday);
print ("Month : ",nmonth); print ("Year : ",nyear);
'''
# 8.
'''name = (input("Enter your name :-")) roll_no
= (input("Enter your Roll number :-")) sem =
(input("Semester :-")) pd = int(input("Enter the
marks of PDS")) ph = int(input("Enter the
marks of Physics")) ch = int(input("Enter the
marks of Chemistry")) py = int(input("Enter
the marks of Python")) en = int(input("Enter
the marks of English")) tm = pd+ph+ch+py+en
#tm = total marks percentage = tm/5
print("Your Percentage are",percentage) cgpa =
percentage/10 print("Your CGPA are",cgpa)
if(cgpa<=3.4):  print("F grade")
elif(cgpa<=5.0):  print("C+ grade")
elif(cgpa<=6.0):  print("B grade")
```

```
elif(cgpa<=7.0):  print("B+ grade")

elif(cgpa<=8.0): print("A grade")

elif(cgpa<=9.0):

print("A+ grade")

elif(cgpa<=10.0):

print("Outstanding")

else:  print(".")

'''
```

Experiment 4: Loops

```
# python lab 4
```

 # 1.Find a factorial of given number.

```
'''num = int(input("Enter a number :- "))

fact = 1 for i in range(1,num+1):  fact =

i*fact print("Factorial :- ",int(fact))

'''
```

 # 2. Find whether the given number is Armstrong number.

```
'''n = int (input ("Enter a no : ")) r

= n;

p = int (input ("Enter no of digit of this no : "))

s=0 ; for i in range (p,0,-1):  tmp = (n%10 ) ;

# print(tmp);  s = s + tmp**3 ;  n = n//10 ; if (r

== s):  print ("This no is armstrog no"); else :

 print ("This is not armstrong no")

'''
```

 # 3. Print Fibonacci series up to given term.

```
'''num = int(input("Enter range :- ")) a

= 0 #ft = first term

b = 1 #ft = second term

for i in range(2,num):

c = a + b  print(c,end

="")  a = b  b = c

'''
```

 # 4. Write a program to find if given number is prime number or not.

```
'''num = int(input("Enter a number :- "))
count = 0 for i in range(2,num//2):
if(num%i ==0):  count += 1  break
if(count == 1):
 print(num,"is a composite number")
elif(count == 0):  print(num,"is a
prime number")
'''
```

 # 5.Check whether given number is palindrome or not.

```
'''n = int (input ("Enter a no : ")) q
= n ;
p = int (input ("Enter no of digits of this no : "))
s = 0 ; for i in range (p,0,-1):
 tmp = n%10 ;  print (tmp, sep
= "" , end = "")  s = s*10 +
tmp ;  n = n//10 ;


print ("\n\n") if (q == s):  print
("This is palindrom no"); else :
 print ("This is not Paildrom no");
'''
```

 # 6. Write a program to print sum of digits.

```
'''n = int (input ("Enter a no : ")); s=0
;
r = int (input ("Enter no of digits : ")) for
i in range (r , 0 , -1):
 tmp = n%10;  print (tmp , end = "")  s = s
+ tmp ;  n = n//10 ; print ("Sum of the
digits of this no is : ",s);
'''
```

 # 7. Count and print all numbers divisible by 5 or 7 between 1 to 100.

```
'''print ("The number divsible by 5 : ")

for i in range (1,100,1):  if (i%5 ==

0):

 print (i ,sep = " " , end = " " );  else :

pass ; print ("\n\nThe number divisible

by 7 :") for j in range (1,100,1):  if (j%7

== 0):

 print (j , sep = " " , end = " ");

else :  pass ;

'''
```

 # 8. Convert all lower cases to upper case in a string.

```
'''str = input ("Enter a string : ");

for i in str :  t = ord (i) -32 ;  t =

chr (t)  print (t , end = " ") ;

'''
```

 # 9. Print all prime numbers between 1 and 100.

```
'''count = 0 ;
print ("Prime number between 1 and 100 : ")

for i in range (1,100,1):  for j in range

(2,100,1):  if (i%j == 0):

 break;

else :

pass ;  if (i

== j):

 print (j , end = " ");

else :  pass

'''
```

 # 10. Print the table for a given number: 5 * 1 = 5 5 * 2 = 10……..

```
'''n = int (input ("Enter the no : ")) p

= int (input ("Enter the length: "))

for i in range (1,p+1,+1) :  print (n

,"*",i ," :",n*i);

'''
```

Experiment 5

python lab 5

 # question 1 Write a program to count and display the number of capital letters in a given string.

```
'''def count_capital_letters(input_string):
count = 0  for char in input_string:  if
char.isupper():
 count += 1  return count def main():  input_string
= input("Enter a string: ")  capital_count =
count_capital_letters(input_string)  print("Number
of capital letters:", capital_count)
if __name__ == "__main__":
main()
'''
```

#question 2 Count total number of vowels in a given string.

```
'''def count_vowels(input_string):
count = 0  vowels =
"aeiouAEIOU"
 for char in input_string:
if char in vowels:  count
+= 1  return count
input_string = input("Enter a string: ") vowel_count
= count_vowels(input_string) print("Number of
vowels:", vowel_count)
'''
```

 # question 3 Input a sentence and print words in separate lines

```
'''def print_words_separate_lines(sentence):  words =
sentence.split()  for word in words:
 print(word)
input_sentence = input("Enter a sentence: ") print("Words
in separate lines:")
print_words_separate_lines(input_sentence)
```

'''

 # question 4 WAP to enter a string and a substring.

'''

```
string=(input("enter the string:"))

#count1 variable is for counting upercase

#count2 variable is for counting lowercase

count1=0 count2=0 for i in string:

if(i.isupper()):

 count1=count1+1

 else:
 count2=count2+1

print("the number of uppercase charaters in the string:",count1) print("the

number of lowercase charaters in the string:",count2)
```

'''

 # questino 5

'''

```
string=(input("enter the string:"))

#count1 variable is for counting upercase

#count2 variable is for counting lowercase

count1=0 count2=0 for i in string:

if(i.isupper()):

 count1=count1+1

 else:

 count2=count2+1

print("the number of uppercase charaters in the string:",count1) print("the

number of lowercase charaters in the string:",count2)
```

'''

 # question 6

```
'''sentence = "this is me @ Rishika"

count=0; upper=0 lower=0

unique=0 for i in sentence:

if(i.isupper()):

 upper=upper+1

 elif(i.islower()):
```

```
 lower=lower+1

 else:

 unique=unique+1 print("the number of unique charaters in the

sentence:",unique)'''
```

#question 7

```
'''n = int(input("Enter the number of fruits for each set: "))
# Input for the first set of fruits print("\nEnter

fruits for set s1:") s1 = set() for i in range(n):

fruit = input(f"Enter fruit {i+1}: ").strip().lower()

s1.add(fruit)

# Input for the second set of fruits print("\nEnter

fruits for set s2:") s2 = set() for i in range(n):

fruit = input(f"Enter fruit {i+1}: ").strip().lower()

s2.add(fruit)

# Fruits in both sets (Intersection) common =

s1.intersection(s2) print("\nFruits in both sets s1

and s2:", common) # Fruits only in s1 but not in

s2 (Difference) onlyfruits_s1 = s1.difference(s2)

print("Fruits only in set s1 but not in set s2:",onlyfruits_s1)

# Count of total no of items in both the stes totalcount =

len(s1.union(s2))

print("Total count of fruits from sets s1 and s2:", totalcount)'''
```

#question 8

```
'''s1 ={'red', 'yellow', 'orange', 'blue'} s2

={'voilet', 'blue', 'purple'}

union=s1.union(s2)

intersection=s1.intersection(s2)

difference1=s1.difference(s2)

difference2=s2.difference(s2)

sysmetric=s1.symmetric_difference(s2)

subset=s2.issubset(s1)

superset=s1.issuperset(s2)

print("union:",union)

print("intersection:",intersection)
```

```
print("difference of set1:",difference1)
print("difference in set2:",difference2)
print("sysmetric difference:",sysmetric)
print("subsets:",subset)
print("superset:",superset)
'''
```

Experiment 6

#1

```
'''n=(int(input("enter the number of items in list:")))
values=[] if n<0:
 print("the number is invalid")
else:  for i in range(n):
 value = int(input(f"enter the value[i+1](between 0 and 3):"))
if 0 <= value <=3:  values.append(value)
 else:
 print("value out of range 0 and 3")

occurance= [0]*4 for num in values:  occurance[num]
+= 1 for i in range(4):  print(f"number of occurance
of {i} : {occurance[i]}")
'''
```

#2

```
'''n = (int(input("enter the items you want in the tuple:"))) if
n<= 0:
 print("the number of items is invalid") else:
my_tuple=tuple(int(input(f"Enter value {i+1}:"))for i in range(n))
average=0
average=sum(my_tuple)/len(my_tuple) print("the
average of the tuple you entered is:",average)
'''
```

#3

```
'''
```

```
N =(int(input("enter the number of students:"))) #inputting
the scores of the students
scores = list(int(input(f"enter the scores of student i:"))for i in range(N))
if len(scores) !=N:  print("number of scores does not matchs the
number of students") else:
 max_scores =max(scores)
scores.remove(max_scores)
runnerup_scores=max(scores) print("the
runner-up score is:",runnerup_scores)
'''
```

#4

```
'''n = (int(input("enter the number of persons:")))
persons = {} for i in range(n):
 name = input(f"enter name of person{i+1}:")
city = input(f"enter the city of {name}:")
persons[name] = city

#displayin the names of the persons
print("\nNames of all the persons:")
for name in persons:  print(name)
#displaying the names of all the cities:
print("\ncity names of all the persons:")
for city in persons.values():  print(city)
#the number of students in the each city:
my_city = {} for city in
persons.values():  my_city[city]=
my_city.get(city, 0)+1 #displaying the
city and persons together print("number
of students in each city:") for city, count
in my_city.items():
print(f"{city}:{count}")
'''
```

#5

```
'''n = (int(input("enter the number of movies:"))) movie={} for i in range(n):  print(f"\nEnter
details for movie{i+1}:")  movie_name = input("enter the name of the movie:")  year =
int(input("enter the release year of the movie:"))  director_name = input("enter the name of the
director of the movie:")  production_cost = float(input("enter the production cost of the
movie:"))  collection = float(input("enter the collection cost of the movie:"))
movie[movie_name]={'year': year, 'director': director_name, 'production_cost': production_cost,
'collection':collection}



#displaying the all details of the movie
print("\nAll movie details:") for
movie_name, details in movie.items():
print(f"MOVIE NAME: {movie_name}")
 print("details:")  for key, value
in details.items():
 print(f"{key}: {value}")
print()

#displayig names of the movie released before 2015
print("\nmovie released before 2015") for
movie_name, details in movie.items():  if
details['year'] < 2015:  print(movie_name)
#displaying the names of the movies which made profit
print("\nmovies realeased that made profit:") for
movie_name, details in movie.items():  if
details['collection'] > details['production_cost']:
 print(movie_name)
#displying the names of the movie for the asked director:
director__name=(input("enter the name of the director:"))
print(f"movie by director:{director__name}:") for
movie_name, details in movie.items():  if
details['director'] == director__name:
 print(movie_name)
'''
```

Experiment 7

#1

```
'''def max(sequence):


max_num = sequence[0]

min_num = sequence[0]

for num in sequence[1:]:

if num > max_num:

max_num=num

 elif num < min_num:

min_num=num return max_num,

min_num sequence =

[5,7,44,8,2,9,70,66] max_num,

min_num=max(sequence)

print("maximum number:",max_num)

print("minimun number:",min_num)

'''
```

#2

```
'''

def sum_of_cube(n):

sum_of_cube = 0

 for i in range(1,n):

sum_of_cube +=(i*i*i)


 return sum_of_cube

num = int(input("enter a positive integer:")) result

= sum_of_cube(num)

print("sum of the cube of the entered positive integer is:", num, "is:", result) '''
```

#3

```
'''#recurrsion means call itself in the function only

def print_number(n):  if n ==0:  return  else:

 print_number(n-1)

print(n)
```

```
num = int(input("enter the number:"))
print("number from 1 to") print_number(num)'''
#4
'''#a=term1 and b=term2
def fibonacci(n, a=0, b=1):
if n<=0:  return  else:
 print(a, end=" ")  fibonacci(n-
1,b,a+b)
num = int(input("enetr the number ofr fibonacci series:"))
print("printing the fibonacci series:") fibonacci(num)
'''
#7
'''#keyword argument:
def argument(name,message):  print(f"\nhello,
{name}! {message}") name = (input("\nenter the
name:")) message = (input("\nenter the message:"))
argument(name,message) #default argument
print("\nfor default argument;") def
argument2(name,message="what are you doing?"):
print(f"\nhello! {name} {message}")
name = (input("\nenter the name:")) message
= (input("\nenter the message:"))
argument2(name,message)
#variable length argument'''
```

Experiment 8

#1. Add few names, one name in each row, in "name.txt file".

```
# a. Count no of names
# b. Count all names starting with vowel
# c. Find longest name def
count_names(filename):  with
open(filename, 'r') as file:
```

```
 names = file.readlines()  return len(names) def
count_names_starting_with_vowel(filename):
vowels = "aeiouAEIOU"
 with open(filename, 'r') as file:
 names = file.readlines()  count = sum(1 for name in names if
name.strip()[0] in vowels)  return count def
find_longest_name(filename):  with open(filename, 'r') as file:
 names = file.readlines()  longest_name = max(names, key=lambda x:
len(x.strip()))  return longest_name.strip() def main():  filename = "myfile.txt"
total_names = count_names(filename)  print("Total number of names:",
total_names)  names_starting_with_vowel =
count_names_starting_with_vowel(filename)  print("Number of names
starting with a vowel:", names_starting_with_vowel)  longest_name =
find_longest_name(filename)  print("Longest name:", longest_name) if name
== "main":  main()
```

#2. Store integers in a file.

```
# a. Find the max number
# b. Find average of all numbers
# c. Count number of numbers greater than 100
def read_integers(filename):  with
open(filename, 'r') as file:
 integers = [int(line.strip()) for line in file]
return integers def
find_max_number(integers):
 return max(integers) def
calculate_average(integers):  return
sum(integers) / len(integers) def
count_numbers_greater_than_100(integers):
return sum(1 for num in integers if num > 100)
def main():  filename = 'myfile.txt'   integers =
read_integers(filename)

 if integers:
```

print("Max number:", find_max_number(integers)) print("Average of all numbers:",

calculate_average(integers)) print("Count of numbers greater than 100:",

count_numbers_greater_than_100(integers))

else:

print("No integers found in the file.")

if name == "main": main()

#3. Assume a file city.txt with details of 5 cities in given format (cityname population(in lakhs) area(in sq KM)):

Example:

Dehradun 5.78 308.20

Delhi 190 1484

……

Open file city.txt and read to:

a. Display details of all cities

b. Display city names with population more than 10Lakhs
c. Display sum of areas of all cities def

details(filename):

cities = [] with

open(filename, 'r') as file: for

line in file:

city_details = line.strip().split() city_name =

city_details[0] population = float(city_details[1]) area =

float(city_details[2]) cities.append((city_name,

population, area)) return cities def

display_all_cities(cities): print("City Details:") for city in

cities: print("City:", city[0]) print("Population (in

lakhs):", city[1]) print("Area (in sq KM):", city[2]) print()

def display_cities_population_more_than_10_lakhs(cities):

print("Cities with population more than 10 Lakhs:") for

city in cities: if city[1] > 10: print(city[0]) def

calculate_sum_of_areas(cities): total_area = sum(city[2]

for city in cities) print("Sum of areas of all cities:",

total_area) def main(): filename = 'city.txt' # Change this

to the name of your file cities = details(filename) if cities:

```
 display_all_cities(cities)

display_cities_population_more_than_10_lakhs(cities)

calculate_sum_of_areas(cities)  else:

 print("No city details found in the file.")
if name == "main":

main()
```

#4. Input two values from user where the first line contains N, the number of test cases. The next N lines contain the

 # space separated values of a and b. Perform integer division and print a/b. Handle exception in case of

 # ZeroDivisionError or ValueError.

```
def perform_division(a, b):

 try:

 result = int(a) // int(b)

print(result)  except ValueError

as ve:  print("Error Code:", ve)

except ZeroDivisionError as zde:

 print("Error Code:", zde) if

name == "main":

 N = int(input("Enter the number of test cases: "))

for _ in range(N):  a, b = input().split()

perform_division(a, b)
```

#5. Create multiple suitable exceptions for a file handling program.

```
def read(filename):

 try:

 with open(filename, 'r') as file:

 content = file.read()

print(content)  except

FileNotFoundError:

 print(f"Error: File '{filename}' not found.")

except PermissionError:

 print(f"Error: Permission denied to open '{filename}'.")

except IsADirectoryError:

 print(f"Error: '{filename}' is a directory, not a file.")

except UnicodeDecodeError:
```

```
 print(f"Error: Unable to decode file '{filename}'. It may not be a text file.")
except Exception as e:
 print(f"An unexpected error occurred: {e}")
except ValueError as ve:  print(f"Error
code:",ve) if name == "main":
 filename = input("Enter the name of the file to read: ")
read(filename)
```

Experiment 9

#1. Create a class of student (name, sap id, marks[phy,chem,maths]). Create 3 objects by taking inputs from the user and display details of all students.

#2. Add constructor in the above class to initialize student details of n students and implement following methods:

#a) Display() student details

#b) Find Marks_percentage() of each student

#c) Display result() [Note: if marks in each subject >40% than Pass else Fail]

```
class Student:  def init(self, name, sap_id, marks):
 self.name = name  self.sap_id = sap_id  self.marks =
marks  def display_details(self):  print("Name:",
self.name)  print("SAP ID:", self.sap_id)  print("Physics
Marks:", self.marks['physics'])  print("Chemistry
Marks:", self.marks['chemistry'])  print("Mathematics
Marks:", self.marks['mathematics'])  print() if name ==
"main":  students = []  for i in range(3):
 name = input("Enter student name: ")  sap_id = input("Enter
SAP ID: ")  physics_marks = float(input("Enter Physics marks:
"))  chemistry_marks = float(input("Enter Chemistry marks: "))
mathematics_marks = float(input("Enter Mathematics marks: "))
marks = {'physics': physics_marks,'chemistry':
chemistry_marks,'mathematics': mathematics_marks }  student =
Student(name, sap_id, marks)  students.append(student)
print("\nDetails of all students:")  for student in students:
student.display_details()
```

 # 3.Create programs to implement different types of inheritances.

Single Inheritance

class Parent: def

parent_method(self):

print("Parent's method")

class Child(Parent): def

child_method(self):

print("Child's method") #

Multiple Inheritance

class Parent1: def

method1(self):

 print("Parent 1's method")

class Parent2: def

method2(self):

 print("Parent 2's method") class

ChildMultiple(Parent1, Parent2): def

child_method(self): print("Child's

method") # Multilevel Inheritance class

Grandparent: def

grandparent_method(self):

print("Grandparent's method") class

ParentMultilevel(Grandparent): def

parent_method(self): print("Parent's

method") class

ChildMultilevel(ParentMultilevel): def

child_method(self): print("Child's

method")

Hierarchical Inheritance
class ParentHierarchical: def

parent_method(self): print("Parent's method")

class Child1Hierarchical(ParentHierarchical):

def child1_method(self): print("Child1's

method") class

Child2Hierarchical(ParentHierarchical): def

child2_method(self): print("Child2's method")

Single Inheritance print("Single

Inheritance:") child_obj = Child()

child_obj.parent_method()

child_obj.child_method() # Multiple

Inheritance print("\nMultiple Inheritance:")

child_multiple_obj = ChildMultiple()

child_multiple_obj.method1()

child_multiple_obj.method2()

child_multiple_obj.child_method() #

Multilevel Inheritance print("\nMultilevel

Inheritance:") child_multilevel_obj =

ChildMultilevel()

child_multilevel_obj.grandparent_method()

child_multilevel_obj.parent_method()

child_multilevel_obj.child_method() #

Hierarchical Inheritance print("\nHierarchical

Inheritance:") child1_hierarchical_obj =

Child1Hierarchical() child2_hierarchical_obj =

Child2Hierarchical()

child1_hierarchical_obj.parent_method()

child1_hierarchical_obj.child1_method()

child2_hierarchical_obj.parent_method()

child2_hierarchical_obj.child2_method()

4.Create a class to implement method Overriding. class Parent: def method(self):

print("Parent's method") class

Child(Parent): def method(self):

print("Child's overridden method")

par_obj = Parent() child_obj =

Child() par_obj.method()

child_obj.method()

 # 5.Create a class for operator overloading which adds two Point Objects where Point has x & y

values e.g. if

P1(x=10,y=20)

P2(x=12,y=15)

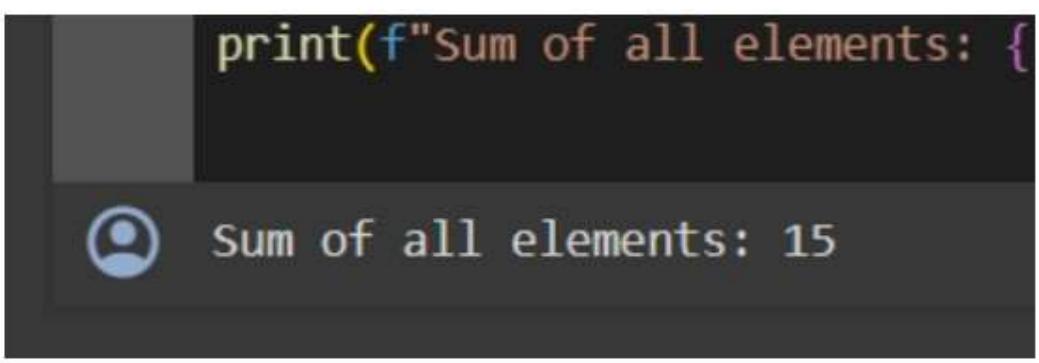
P3=P1+P2 => P3(x=22,y=35)

```
class Point:  def init(self, x,
y):
 self.x = x  self.y =
y  def add(self,
other):
 if isinstance(other, Point): #subclass
new_x = self.x + other.x  new_y =
self.y + other.y  return Point(new_x,
new_y)
 else:
 raise TypeError("Unsupported operand type(s) for +: 'Point' and '{}'".format(type(other)))
def str(self):
 return "Point(x={}, y={})".format(self.x, self.y)
P1 = Point(25, 60)
P2 = Point(92, 10) P3
= P1 + P2
print("P1:", P1)
print("P2:", P2)
print("P3:",P3)
```

Experiment 10

#1. Create numpy array to find sum of all elements in an array.

```
import numpy as np arr = np.array([1, 2, 3, 4, 5]) sum_of_elements
= np.sum(arr)
print(f"Sum of all elements: {sum_of_elements}")
```



#2. Create numpy array of (3,3) dimension. Now find sum of all rows & columns individually. Also find 2nd maximum element in the array import numpy as np

matrix = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

sum_of_rows = np.sum(matrix, axis=1) sum_of_columns =

np.sum(matrix, axis=0) second_max_element =

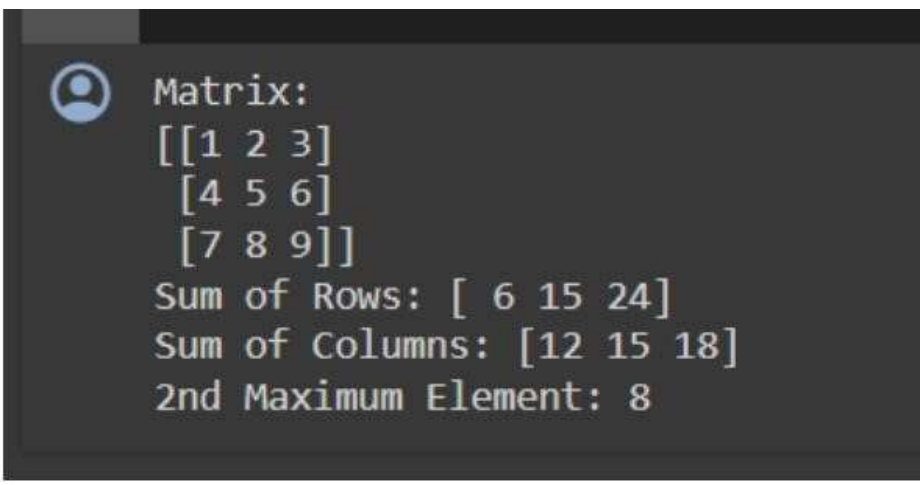
np.partition(matrix.flatten(), -2)[-2]

print(f"Matrix:\n{matrix}") print(f"Sum of Rows:

{sum_of_rows}") print(f"Sum of Columns:

{sum_of_columns}") print(f"2nd Maximum Element:

{second_max_element}")



#3. Perform Matrix multiplication of any 2 n*n matrices. import

numpy as np

matrix1 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]]) matrix2 =

np.array([[9, 8, 7], [6, 5, 4], [3, 2, 1]]) result_matrix =

np.dot(matrix1, matrix2) print(f"Matrix 1:\n{matrix1}")

print(f"Matrix 2:\n{matrix2}") print(f"Result Matrix (Matrix 1

* Matrix 2):\n{result_matrix}")

```
Matrix 1:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Matrix 2:
[[9 8 7]
 [6 5 4]
 [3 2 1]]
Result Matrix (Matrix 1 * Matrix 2):
[[ 30  24  18]
 [ 84  69  54]
 [138 114  90]]
```

#4. Write a Pandas program to get the powers of an array values element-wise. import

pandas as pd

data = {'X': [78, 85, 96, 80, 86], 'Y': [84, 94, 89, 83, 86], 'Z': [86, 97, 96, 72, 83]}

df = pd.DataFrame(data) powers_df = df.pow([1, 2, 3]) print("Expected

Output:") print(powers_df)

```
Expected Output:
    X     Y       Z
0  78  7056  636056
1  85  8836  912673
2  96  7921  884736
3  80  6889  373248
4  86  7396  571787
```

#5. Write a Pandas program to get the first 3 rows of a given DataFrame.

import pandas as pd import

numpy as np

exam_data = {'name': ['Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Kevin', 'Jonas'],

'score': [12.5, 9, 16.5, np.nan, 9, 20, 14.5, np.nan, 8, 19],

'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],

'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes']} labels

= ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

df = pd.DataFrame(exam_data, index=labels) first_three_rows

= df.head(3)

print("First three rows of the data frame:") print(first_three_rows)

```
First three rows of the data frame:
        name  score  attempts qualify
a  Anastasia   12.5         1     yes
b       Dima    9.0         3      no
c  Katherine   16.5         2     yes
```

#6. Write a Pandas program to find and replace the missing values in a given DataFrame which do not

have any valuable information.

```
import pandas as pd import

numpy as np

data = {'Name': ['John', 'Alice', 'Bob', np.nan, 'Eve'],

 'Age': [25, np.nan, 30, np.nan, 22],

 'Salary': [50000, 60000, np.nan, np.nan, 55000]}

df = pd.DataFrame(data) df_filled = df.fillna(-1)

print("DataFrame with Missing Values:")

print(df) print("\nDataFrame after

Replacement:") print(df_filled)
```

```
DataFrame with Missing Values:
    Name   Age   Salary
0   John  25.0  50000.0
1  Alice   NaN  60000.0
2    Bob  30.0      NaN
3    NaN   NaN      NaN
4    Eve  22.0  55000.0

DataFrame after Replacement:
    Name   Age   Salary
0   John  25.0  50000.0
1  Alice  -1.0  60000.0
2    Bob  30.0     -1.0
3     -1  -1.0     -1.0
4    Eve  22.0  55000.0
```

#7. Create a program to demonstrate different visual forms using Matplotlib.

```
import matplotlib.pyplot as plt
import numpy as np x =

np.linspace(0, 10, 100) y1 =

np.sin(x) y2 = np.cos(x)

plt.figure(figsize=(8, 4))
```

```
plt.subplot(2, 2, 1) plt.plot(x,
y1, label='sin(x)') plt.plot(x,
y2, label='cos(x)')
plt.title('Line Plot') plt.legend()
plt.subplot(2, 2, 2)
plt.scatter(x, y1, label='sin(x)')
plt.scatter(x, y2, label='cos(x)')
plt.title('Scatter Plot')
plt.legend()
categories = ['Category A', 'Category B', 'Category C']
values = [20, 35, 25] plt.subplot(2, 2, 3)
plt.bar(categories, values) plt.title('Bar Chart') data =
np.random.randn(1000) plt.subplot(2, 2, 4)
plt.hist(data, bins=30, edgecolor='black'
```