NAME : PRANKUR SHARMA

BATCH : 45

SAP CODE: 500124514

ROLL NO. R-2142231501

: : :

PYTHON LAB FILE

FACULTY : Ms SUGANDHA SHARMA

About python language

What is Python used for?

Python is commonly used for developing websites and software, task automation, data analysis, and data visualisation. Since it’s relatively easy to learn, Python has been adopted by many non-programmers, such as accountants and scientists, for a variety of everyday tasks, like organising finances.
"Writing programs is a very creative and rewarding activity," says University of Michigan and Coursera instructor Charles R Severance in his book Python for Everybody. "You can write programs for many reasons, ranging from making your living to solving a difficult data analysis problem to having fun to helping someone else solve a problem."
What can you do with Python? Some things include:

- Data analysis and machine learning
- Web development
- Automation or scripting
- Software testing and prototyping
- Everyday tasks

About   Python installation

Installing Python is a relatively simple process that begins with downloading the appropriate installer from the official Python website. It's crucial to select the Python 3.x version, as Python 2.x is now considered legacy. Once the installer is downloaded, running it typically involves following straightforward prompts on Windows or macOS systems, or using package managers on Linux distributions. After installation, verifying the installation can be done by opening a command prompt or terminal and typing `python --version` or `python3 --version`, which should display the installed Python version. Optionally, on Windows, you can choose to add Python to the system's PATH during installation for easier command-line access. With Python successfully installed, you're ready to start coding and leveraging its powerful capabilities for various projects and tasks.

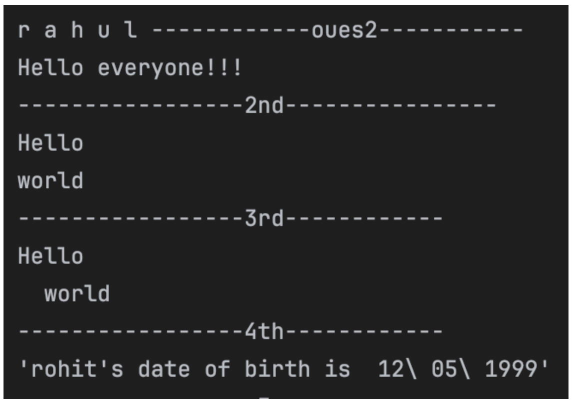
About functions in python

functions are essential building blocks for structuring code. They encapsulate reusable blocks of logic, promoting modularity and enhancing code readability and maintainability. Functions are defined using the def keyword, followed by the function name and optional parameters enclosed in parentheses. The function body is then indented, adhering to Python's indentation-based syntax. Parameters can be passed to functions, allowing them to operate on different data inputs. Functions can return values using the return statement, providing flexibility in how they interact with other parts of the program. Python also supports default parameters, enabling functions to have optional arguments with predefined values. Additionally, keyword arguments allow for more flexible function calls by specifying arguments using parameter names. Python functions can accept a variable number of arguments using *args and **kwargs, facilitating the creation of functions that can handle varying inputs. Overall, functions in Python serve as powerful tools for organizing code, promoting reusability, and enhancing the overall structure and readability of programs.

Python experiments

Experiment 1. Python installation and starting

```
print(“------------ques -------")
"""2. Write Python programs to print strings in the given manner: a)
Hello Everyone !!!
b) Hello
World c) Hello
World
d) ‘ Rohit’ s date of birth is 12\05\1999’"""


print("Hello everyone!!!")

print("-----------------2nd----------------")

print("Hello\nworld")

print("-----------------3rd------------")
print("Hello\n  world")

print("-----------------4th------------")

print("'rohit's date of birth is  12\ 05\ 1999'")
```

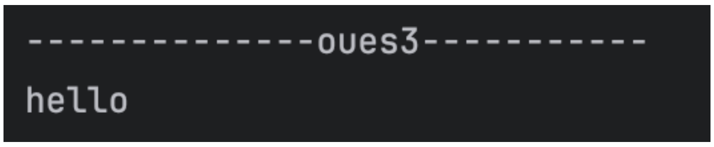


```
r a h u l ------------oues2----------
Hello everyone!!!
-----------------2nd----------------
Hello
world
-----------------3rd------------
Hello
  world
-----------------4th------------
'rohit's date of birth is  12\ 05\ 1999'
```

```
print("--------------ques3-----------")
```

#3 Declare a string variable called x and assign it the value “Hello”.
Print out the value of x

```
x="hello"
print(x)
```



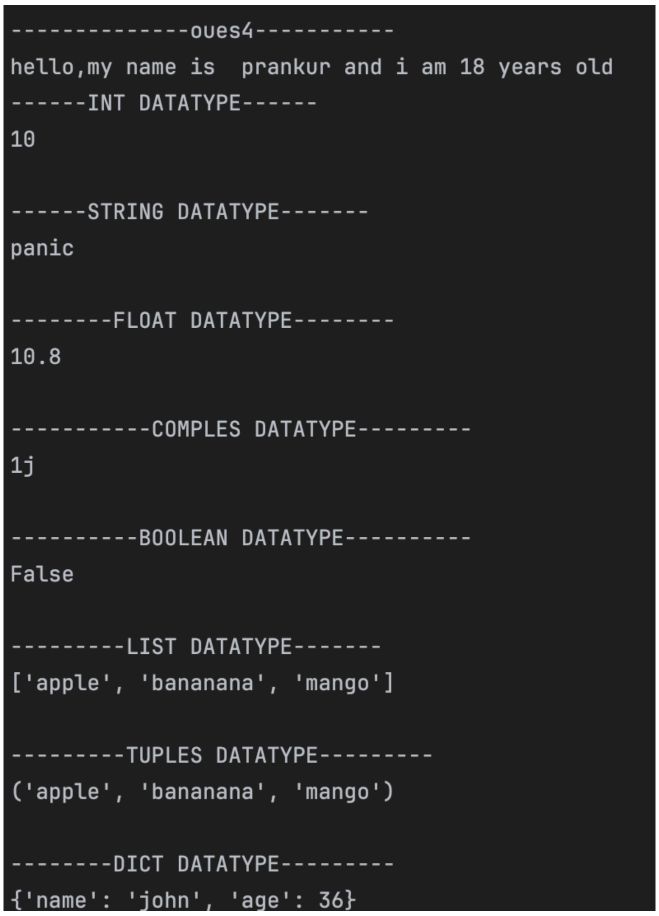
```
--------------ques3-----------
hello
```

```
print("--------------ques4-----------")
```

 #4 Take different data types and print values using print function.

```
name="prankur"
age="18"
print("hello,my name is ", name ,"and i am", age ,"years old")

print("------INT DATATYPE------")
X=10
print(X)
print("")
print("------STRING DATATYPE-------")
x="panic"
print(x)
print("")
print("--------FLOAT DATATYPE--------")
x=10.8
print(x)
print("")
print("-----------COMPLES DATATYPE---------")
x=1j
print(x)
print("")
print("----------BOOLEAN DATATYPE----------")
a = 10
b = 20
print(a == b)
print("")
print("---------LIST DATATYPE-------")
x=["apple","banana","mango"]
print (x)
```

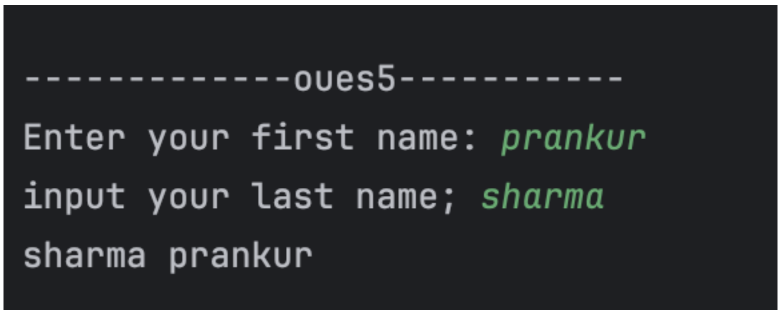
```
print("")
print("---------TUPLES DATATYPE---------")
x=("apple","bananana","mango")
print (x)
print("")
print("--------DICT DATATYPE---------")
x= {"name":"john" , "age" : 36}
print(x)

print("")
```



```
--------------oues4-----------
hello,my name is  prankur and i am 18 years old
------INT DATATYPE------
10

------STRING DATATYPE-------
panic

--------FLOAT DATATYPE--------
10.8

-----------COMPLES DATATYPE---------
1j

----------BOOLEAN DATATYPE----------
False

---------LIST DATATYPE-------
['apple', 'bananana', 'mango']

---------TUPLES DATATYPE---------
('apple', 'bananana', 'mango')

--------DICT DATATYPE---------
{'name': 'john', 'age': 36}
```

```
print("--------------ques5-----------")
""" 5 Take two variable a and b. Assign your first name and last name.
Print your Name after adding your First name and Last name
together."""

fname=input("Enter your first name: ")

lname=input("input your last name; ")
```

```
print(""  + lname  +  " " +  fname )
```

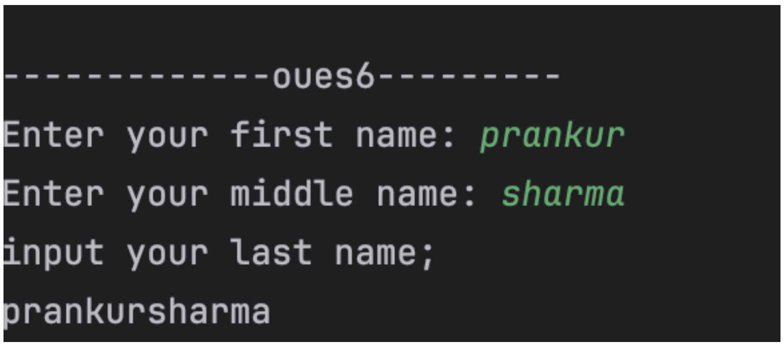


```
-------------ques5-----------
Enter your first name: prankur
input your last name; sharma
sharma prankur
```

```
print("-------------ques6---------")
```

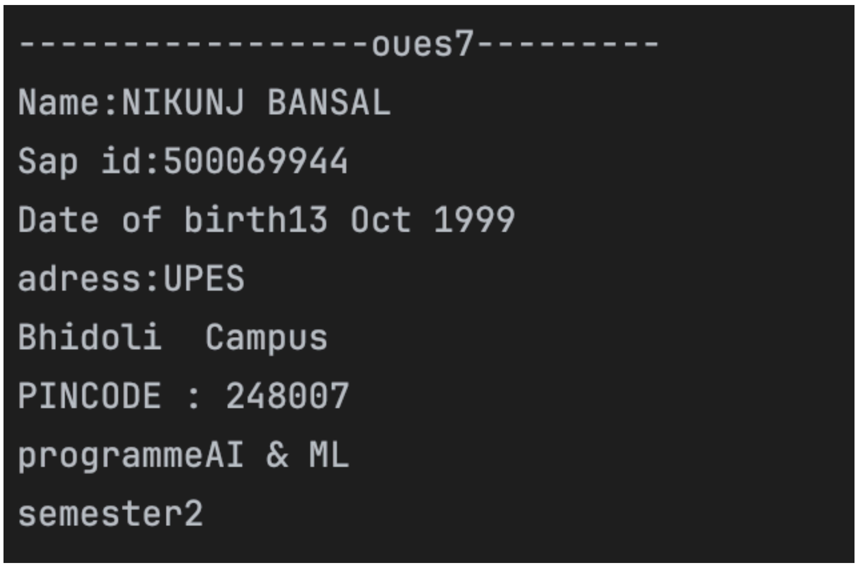
"""6 Declare three variables, consisting of your first name, your last name and Nickname.
 Write a program that prints out your first name, then your nickname in parenthesis and then your last name."""

```
fname=input("Enter your first name: ")
mname=input("Enter your middle name: ")
lname=input("input your last name; ")

print("" + fname + ""+ mname+ "" + lname + " “)
```



```
-------------ques6---------
Enter your first name: prankur
Enter your middle name: sharma
input your last name;
prankursharma
```

```
print("-----------------ques7---------")
"""7 Declare and assign values to suitable variables and print in the
following way : NAME : NIKUNJ BANSAL
SAP ID : 500069944
DATE OF BIRTH : 13 Oct 1999
ADDRESS : UPES
Bidholi Campus
Pincode : 248007 Programme : AI & ML"""


NAME ="NIKUNJ BANSAL"
SAP_ID = "500069944"
DATE_OF_BIRTH = "13 Oct 1999"
ADDRESS = ("UPES \nBhidoli  Campus \nPINCODE : 248007")

Programme = "AI & ML"
Semester = "2"
print("Name:"+NAME, "\nSap id:"+SAP_ID, "\nDate of
birth"+DATE_OF_BIRTH,"\nadress:"+ADDRESS,"\nprogramme"+Progra
mme,"\nsemester"+Semester )
```

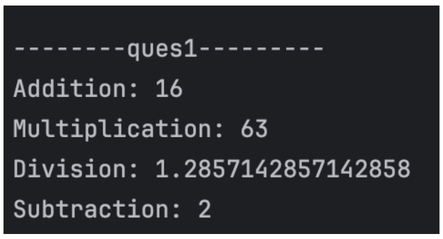


```
-----------------oues7---------
Name:NIKUNJ BANSAL
Sap id:500069944
Date of birth13 Oct 1999
adress:UPES
Bhidoli  Campus
PINCODE : 248007
programmeAI & ML
semester2
```

Experiment 2. Use of input statements

```
"""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."""
print("--------ques1---------")

x = 9
y = 7
print("Addition:", x + y)
print("Multiplication:", x * y)
print("Division:", x / y)
print("Subtraction:", x - y)
```

Output



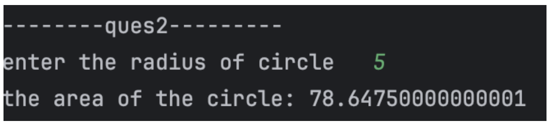
```
--------ques1---------
Addition: 16
Multiplication: 63
Division: 1.2857142857142858
Subtraction: 2
```

```
print("--------ques2---------")
```

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

```
radius = float(input("enter the radius of circle"))
area = 3.1459 * radius ** 2
print("the area of the circle:”,area)
```

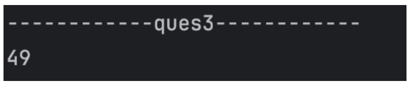
Output



```
--------ques2---------
enter the radius of circle   5
the area of the circle: 78.64750000000001
```

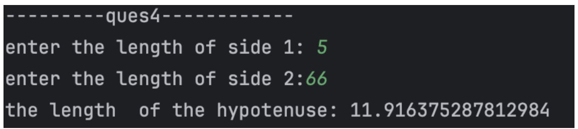
```
print("------------ques3-----------")

"""3. Write a Python program to  solve(x+y)*(x+y)
Test data : x = 4 , y = 3"""

x=4
y=3
area=(x+y)*(x+y)
print(area)
```

Output



```
------------ques3-----------
49
```

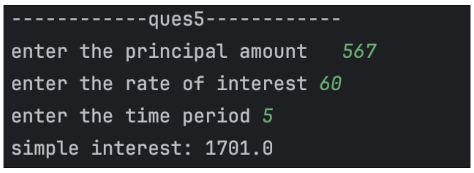
```
print("---------ques4-----------")
"""4. Write a program to compute the length of the hypotenuse (c) of a right triangle
using Pythagoras theorem."""
import math
side1=float(input("enter the length of side 1:"))
side2=float(input("enter the length of side 2:"))
hypotenuse=math.sqrt(side1* 2 + side2 *2)
print("the length  of the hypotenuse:”,hypotenuse)
```



```
---------ques4-----------
enter the length of side 1: 5
enter the length of side 2:66
the length  of the hypotenuse: 11.916375287812984
```

```
print("------------ques5------------")
"""5. Write a program to  find simple interest."""
principal=float(input("enter the principal amount"))
rate=float(input("enter the rate of interest"))
time=float(input("enter the time period"))
simple_interest=(principal *rate *time )/100
print("simple interest:”,simple_interest)
```

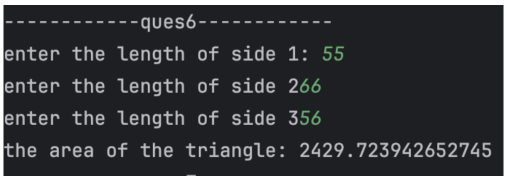
Output



```
------------ques5------------
enter the principal amount   567
enter the rate of interest 60
enter the time period 5
simple interest: 1701.0
```

```
print("------------ques6------------")
#6. Write a program to find area of triangle when length of sides are 
given
import math

a=float(input("enter the length of side 1:"))
b=float(input("enter the length of side 2"))
c=float(input("enter the length of side 3"))
s=(a+b+c)/2
area_triangle=math.sqrt(s*(s-a)*(s-b)*(s))
print("the area of the triangle:”,area_triangle)
```

Output

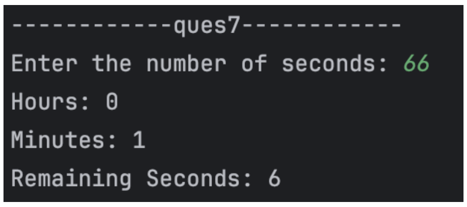


```
------------ques6------------
enter the length of side 1: 55
enter the length of side 266
enter the length of side 356
the area of the triangle: 2429.723942652745
```

```
print("------------ques7------------")
```

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

seconds = int(input("Enter the number of seconds: "))
hours = seconds // 3600
minutes = (seconds % 3600) // 60
remaining_seconds = (seconds % 3600) % 60
print("Hours:", hours)
print("Minutes:", minutes)
print("Remaining Seconds:", remaining_seconds)
```

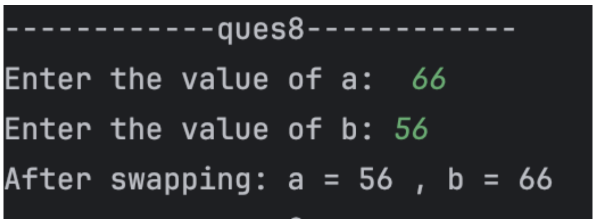
Output



```
------------ques7-----------
Enter the number of seconds: 66
Hours: 0
Minutes: 1
Remaining Seconds: 6
```

```
print("------------ques8------------")
# 8. Write a program to swap two numbers without taking additional
variable.
a = int(input("Enter the value of a: "))
b = int(input("Enter the value of b: "))
a = a + b
b = a - b
a = a - b
print("After swapping: a =", a, ", b =", b)
```

Output

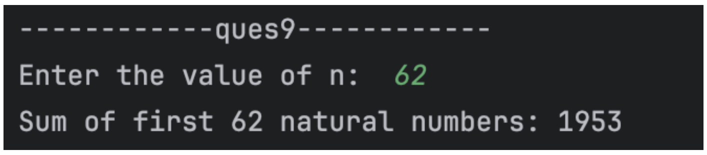


```
------------ques8------------
Enter the value of a:  66
Enter the value of b: 56
After swapping: a = 56 , b = 66
```

```
print("------------ques9------------")
```

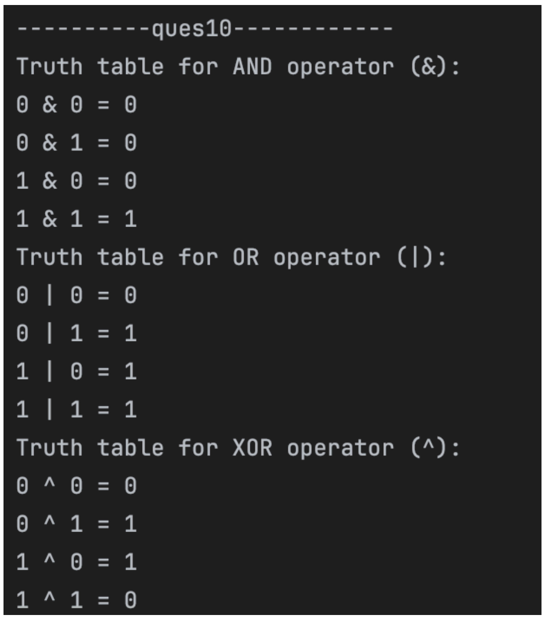
```
# 9. Write a program to find the sum of the first n natural numbers.
n = int(input("Enter the value of n: "))
sum_natural_numbers = (n * (n + 1)) // 2
print("Sum of first", n, "natural numbers:", sum_natural_numbers)
```

Output



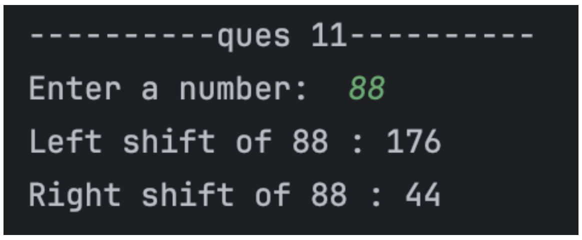
```
------------ques9------------
Enter the value of n:  62
Sum of first 62 natural numbers: 1953
```

```
print("----------ques10------------")
# 10. Write a program to print the truth table for bitwise operators (& , |,
and ^ operators).
print("Truth table for AND operator (&):")
print("0 & 0 =", 0 & 0)
print("0 & 1 =", 0 & 1)
print("1 & 0 =", 1 & 0)
print("1 & 1 =", 1 & 1)

print("Truth table for OR operator (|):")
print("0 | 0 =", 0 | 0)
print("0 | 1 =", 0 | 1)
print("1 | 0 =", 1 | 0)
print("1 | 1 =", 1 | 1)

print("Truth table for XOR operator (^):")
print("0 ^ 0 =", 0 ^ 0)
print("0 ^ 1 =", 0 ^ 1)
print("1 ^ 0 =", 1 ^ 0)
print("1 ^ 1 =", 1 ^ 1)
```



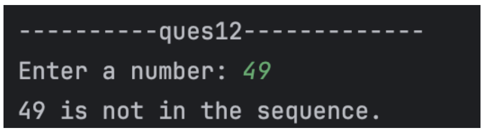
```
----------ques10------------
Truth table for AND operator (&):
0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1
Truth table for OR operator (|):
0 | 0 = 0
0 | 1 = 1
1 | 0 = 1
1 | 1 = 1
Truth table for XOR operator (^):
0 ^ 0 = 0
0 ^ 1 = 1
1 ^ 0 = 1
1 ^ 1 = 0
```

```
print("----------ques 11----------")
# 11. Write a program to find left shift and right shift values of a given number.
number = int(input("Enter a number: "))
left_shift = number << 1
right_shift = number >> 1
print("Left shift of", number, ":", left_shift)
print("Right shift of", number, ":", right_shift)
```



```
----------ques 11----------
Enter a number:  88
Left shift of 88 : 176
Right shift of 88 : 44
```

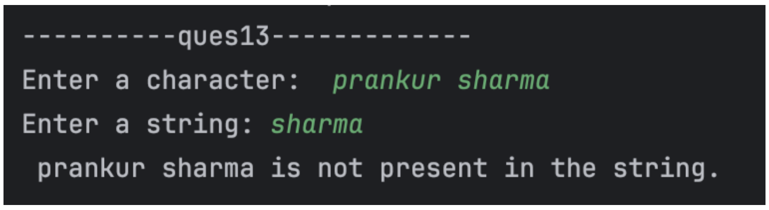
```
print("----------ques12-------------")
# 12. Using a membership operator find whether a given number is in the sequence (10, 20, 56, 78, 89)
number = int(input("Enter a number: "))
sequence = [10, 20, 56, 78, 89]
if number in sequence:
   print(number, "is in the sequence.")
else:
   print(number, "is not in the sequence.”)
```



```
----------ques12-------------
Enter a number: 49
49 is not in the sequence.
```

```
print("----------ques13------------")
# 13. Using membership operator find whether a given character is in a string.
character = input("Enter a character: ")
string = input("Enter a string: ")
if character in string:
    print(character, "is present in the string.")
else:
    print(character, "is not present in the string.")
```

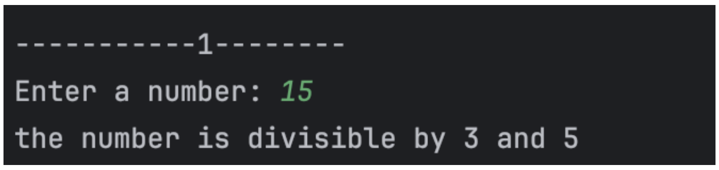
Output



```
----------ques13------------
Enter a character:  prankur sharma
Enter a string: sharma
 prankur sharma is not present in the string.
```

Experiment 3. Conditional statement

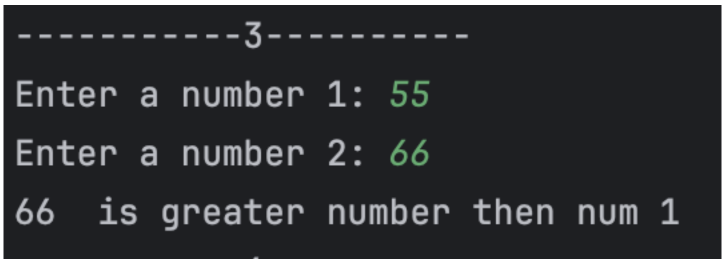
```
#1. Check whether given  number  is  divisible by  3 and 5 both.
print("-----------1--------")
num=int(input("Enter a number: "))
if num%3==0:
 if num %5== 0:
    print("the number is divisible by 3 and 5”)
```



```
-----------1--------
Enter a number: 15
the number is divisible by 3 and 5
```

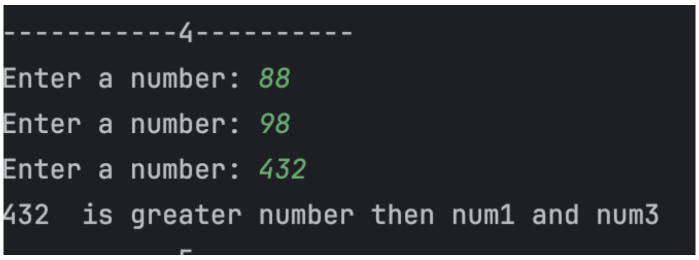
```
print("-----------2----------")
#2. Check   whether a given number is multiple  of five or not.
num=int(input("Enter a number: "))
if num%5==0:
    print(f"the number", num,"is multiple of  5")
```

```
#3. Find the greatest among two numbers.If numbers are equal than
print“numbers are equal”.
print("-----------3----------")
num1=int(input("Enter a number 1: "))
num2=int(input("Enter a number 2: "))
if num1 > num2:
  print(num1," is greater number then num2")
elif num1<num2:
  print(num2," is greater number then num 1")
else:
    print("both  numbers are equal")
    print(“")
```



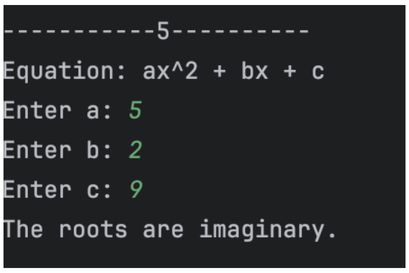
```
-----------3----------
Enter a number 1: 55
Enter a number 2: 66
66  is greater number then num 1
```

```
print("-----------4----------")
#4. Find the greatest among three numbers assuming no two values are
same.
num1=int(input("Enter a number: "))
num2=int(input("Enter a number: "))
num3=int(input("Enter a number: "))
if num1>num2 and num1>num3:
    print(num1, " is greater number then num2 and num 3")

elif num2<num1 and num2<num3:
    print(num2, " is greater number then num1 and num3")

else:
    print(num3, " is greater number then num1 and num3”)
```



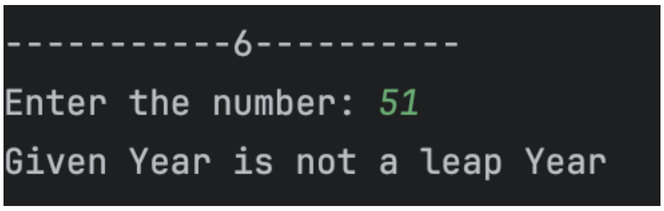
```
-----------4----------
Enter a number: 88
Enter a number: 98
Enter a number: 432
432  is greater number then num1 and num3
```

```
print("-----------5----------")
#5. Check whether the quadratic equation has real roots or imaginary
roots.Display the roots.
#Check whether the quadratic equation has real roots or imaginary roots.
Display the roots.
import math
print("Equation: ax^2 + bx + c ")
a=int(input("Enter a: "))
b=int(input("Enter b: "))
c=int(input("Enter c: "))
d=b**2-4*a*c
if(d<0):
    print("The roots are imaginary. ")
else:
    r1=((-b+ math.sqrt(d))/2 *a )
    r2=((-b- math.sqrt(d))/2 *a )
    print("The first root: ",(r1))
    print("The second root: ",(r2))
print("")
```



```
----------5----------
Equation: ax^2 + bx + c
Enter a: 5
Enter b: 2
Enter c: 9
The roots are imaginary.
```

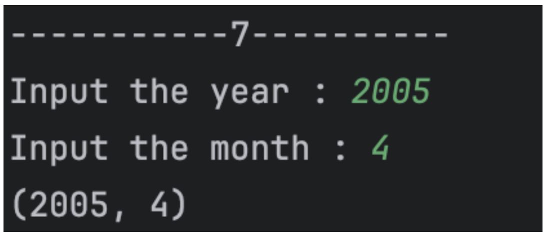
```
#6. Find whether a given year is a leap year or not.
print("-----------6----------")
def CheckLeap(Year):
  # Checking if the given year is leap year
  if((Year % 400 == 0) or
     (Year % 100 != 0) and
     (Year % 4 == 0)):
    print("Given Year is a leap Year");
  # Else it is not a leap year
  else:
    print ("Given Year is not a leap Year")
# Taking an input year from user
Year = int(input("Enter the number: "))
# Printing result
CheckLeap(Year)
```



```
-----------6----------
Enter the number: 51
Given Year is not a leap Year
```

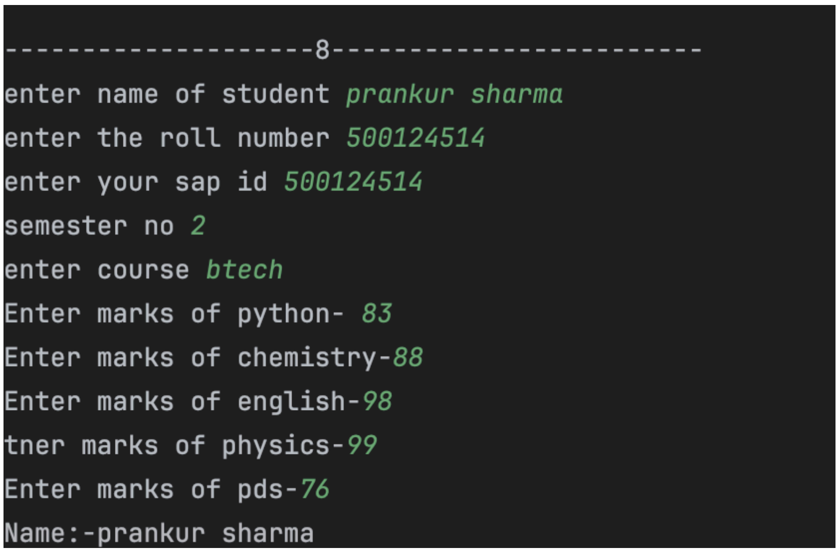
```
print("-----------7----------")
"""7. Write a program which takes any date as input and display next date of the
calendar
e.g.
I/P: day=20 month=9 year=2005 O/P: day=21 month=9 year 2005"""
# Prompt the user to input the year and month
y = int(input("Input the year : "))
m = int(input("Input the month : "))
z=y+1
```

```
# Print the calendar for the specified year and month
print((y, m))
print((z , m))
```

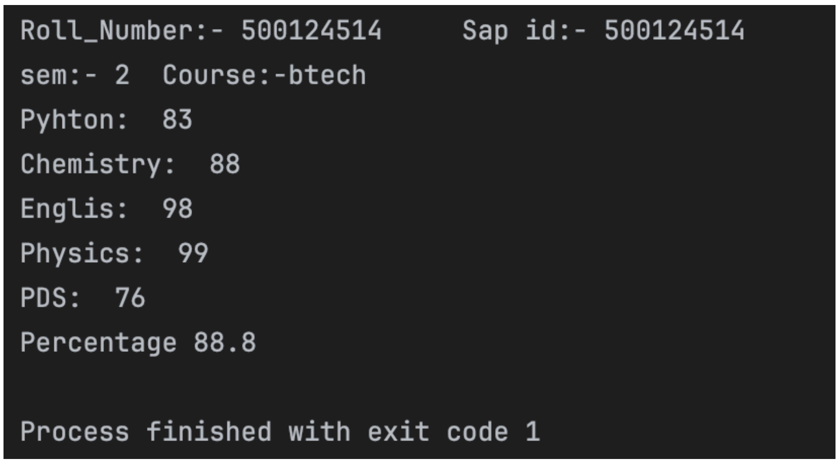


```
-----------7----------
Input the year : 2005
Input the month : 4
(2005, 4)
```

```
priint("--------------------8-----------------------")
name=str(input("enter name of student "))
Roll_Number=int(input("enter the roll number "))
Sap_ID=int(input("enter your sap id "))
sem=int(input("semester no "))
course=str(input("enter course "))
python=int(input("Enter marks of python-"))

Chemistry=int(input("Enter marks of chemistry-"))

English=int(input("Enter marks of english-"))
Physics=int(input("tner marks of physics-"))

pds=int(input("Enter marks of pds-"))
print(f"Name:-"+name,"\nRoll_Number:-",Roll_Number, f"\tSap id:-",
Sap_ID,f"\nsem:-", sem , f" Course:-"+course)
print("Pyhton: ",python,"\nChemistry: ",Chemistry, "\nEnglis: ",English,
"\nPhysics: ",Physics, "\nPDS: ",pds)
percentage=(python + Chemistry + English + Physics + pds)/5
print("Percentage",percentage)

cgp=percentage/10
print(round("CGPA",cgp))
if cgp >= 9.1:
   print("Grade :- O (Outstanding)")
elif cgp >= 8.1:
    print("Grade :- A+")
elif cgp >= 7.1:
    print("Grade :- A")
elif cgp >= 6.1:
   print("Grade :- B+")
elif cgp >= 5.1:
   print("Grade :- B")
elif cgp >= 3.5:
   print("Grade :- C+")
else:
   print("Grade :-  F")
```



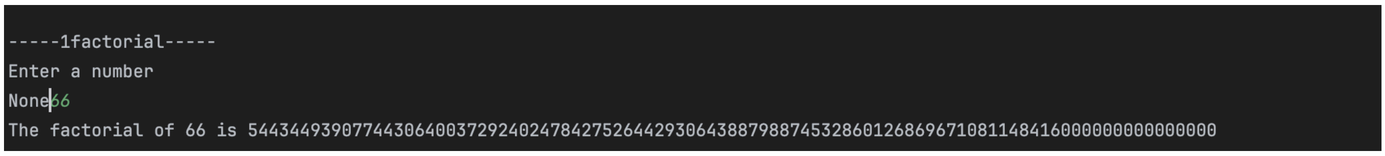
```
--------------------8-----------------------
enter name of student prankur sharma
enter the roll number 500124514
enter your sap id 500124514
semester no 2
enter course btech
Enter marks of python- 83
Enter marks of chemistry-88
Enter marks of english-98
tner marks of physics-99
Enter marks of pds-76
Name:-prankur sharma
```



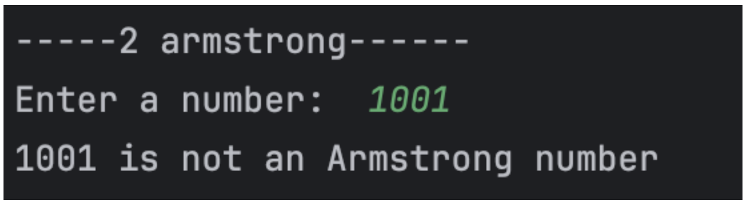
```
Roll_Number:- 500124514     Sap id:- 500124514
sem:- 2  Course:-btech
Pyhton:  83
Chemistry:  88
Englis:  98
Physics:  99
PDS:  76
Percentage 88.8

Process finished with exit code 1
```

Experiment 4. Loops

```
# loops
#Experiment 4: Loops


#1. Find a factorial of given number.
print("------1factorial-----")
num = int(input(print("Enter a number")))
# To take input from the user
#num = int(input("Enter a number: "))

factorial = 1

# check if the number is negative, positive or zero
if num < 0:

    print("Sorry, factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
   for i in range(1,num + 1):
       factorial = factorial*i
   print("The factorial of",num,"is",factorial)
```



```
-----1factorial-----
Enter a number
None66
The factorial of 66 is 544344939077443064003729240247842752644293064388798874532860126869671081148416000000000000000
```



```
-----2 armstrong------
Enter a number:  1001
1001 is not an Armstrong number
```

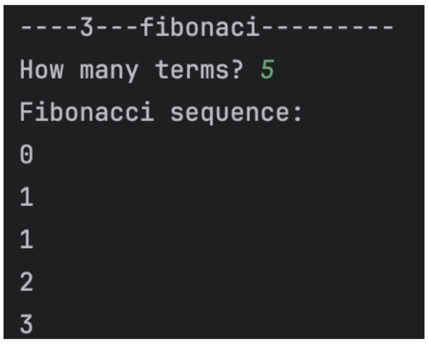
#2. Find whether the given number is Armstrong number.

```
print("-----2 armstrong------")
# Python program to check if the number is an Armstrong number or
not

# take input from the user
num = int(input("Enter a number: "))

sum = 0


temp = num
while temp > 0:
   digit = temp % 10
   sum += digit ** 3
   temp //= 10

if num == sum:
   print(num,"is an Armstrong number")
else:
   print(num,"is not an Armstrong number")
```

#3. Print Fibonacci series upto given term.

```
print("----3---fibonaci---------")
# Program to display the Fibonacci sequence up to n-th term
```

```
nterms = int(input("How many terms? "))

# first two terms
n1, n2 = 0, 1
count = 0

# check if the number of terms is valid
if nterms <= 0:
   print("Please enter a positive integer")
# if there is only one term, return n1
elif nterms == 1:
   print("Fibonacci sequence upto",nterms,":")
   print(n1)
# generate fibonacci sequence
else:
   print("Fibonacci sequence:")
   while count < nterms:
       print(n1)
       nth = n1 + n2
       # update values
       n1 = n2
       n2 = nth
       count += 1
```

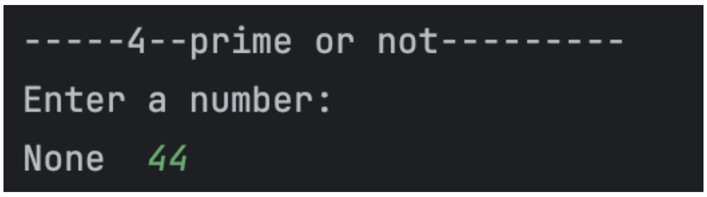


```
----3---fibonaci---------
How many terms? 5
Fibonacci sequence:
0
1
1
2
3
```

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

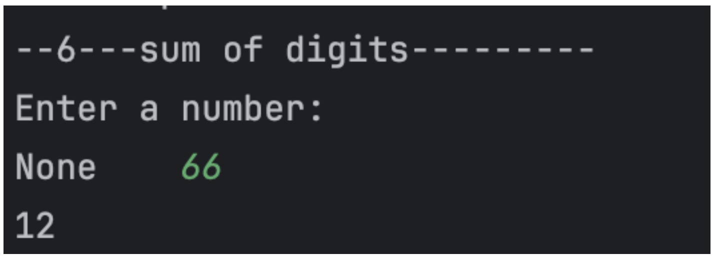
```
print("-----4--prime or not---------")


num = int(input(print("Enter a number: ")))
flag = False

if num == 1:
    print(num, "is not a prime number")
```

```
elif num > 1:
    # check for factors
    for i in range(2, num):
        if (num % i) == 0:

            flag = True

            break

    if flag:
        print(num, "is not a prime number")
    else:
        print(num, "is a prime number”)
```



```
-----4--prime or not---------
Enter a number:
None  44
```

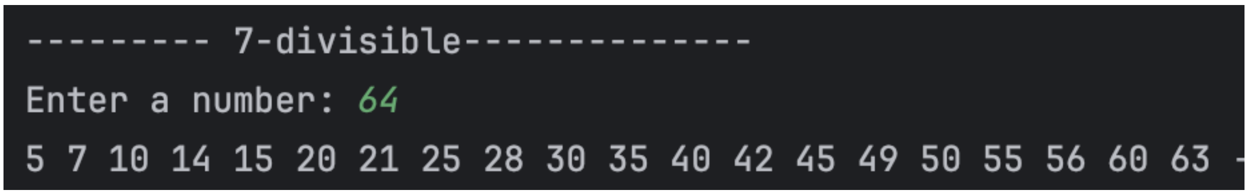
```
# 5. Check whether given number is palindrome or  not.
print("---- 5-palindrome---------")



#6. Write a program to print sum of digits.
print("--6---sum of digits---------")
# Function to get the sum of digits
def getSum(n):

    sum = 0
    while (n != 0):

        sum = sum + (n % 10)
        n = n//10
    return sum

n =int(input(print("Enter a number: ")))
print(getSum(n))
```



```
--6---sum of digits---------
Enter a number:
None    66
12
```

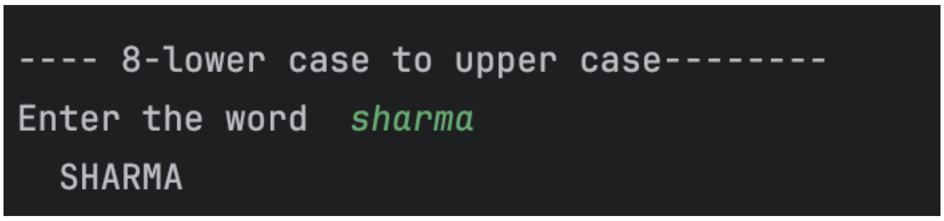
#7. Count and print all numbers divisible by 5 or 7 between 1 to 100.

```
print("--------- 7-divisible-------------")
# divisible by 5 or 7 for a given number

# Result generator with N

def NumGen(n):
        # iterate from 0 to N

        for j in range(1, n + 1):

                # Short-circuit operator is used

                if j % 5 == 0 or j % 7 == 0:
                        yield j


# Driver code

if __name__ == "__main__":

        # input goes here

        N =int(input("Enter a number: "))

        # Iterating over generator function

        for j in NumGen(N):
                print(j, end=" “)
```


```
--------- 7-divisible-------------
Enter a number: 64
5 7 10 14 15 20 21 25 28 30 35 40 42 45 49 50 55 56 60 63 -
```

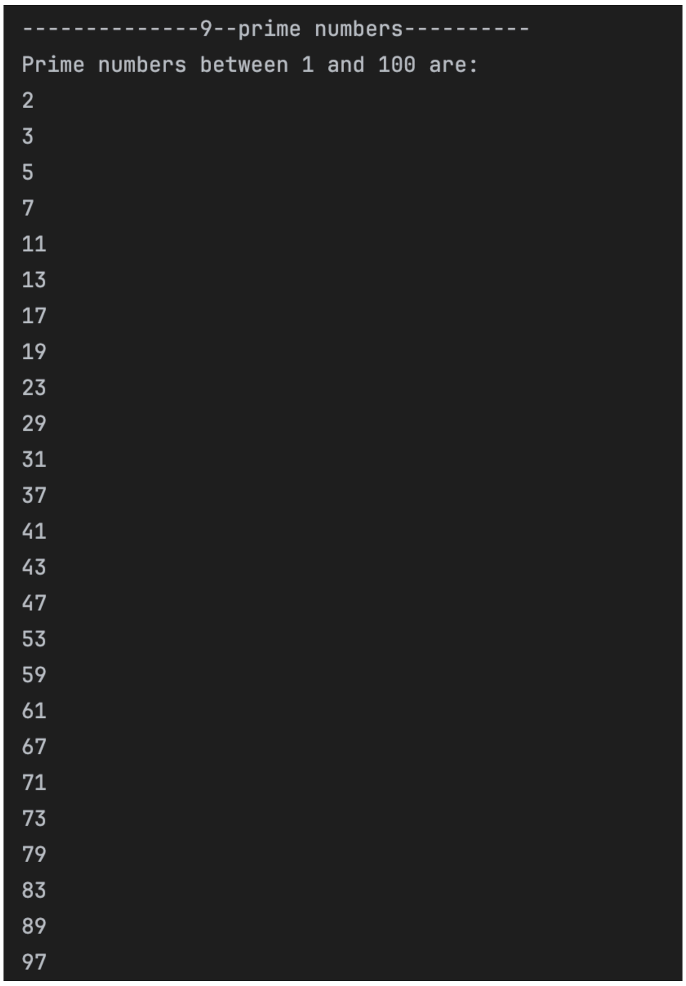
#8. Convert all lower cases to upper case in a string.

```
print("---- 8-lower case to upper case--------")
x=(input("Enter the word"))
print(x.upper())
```

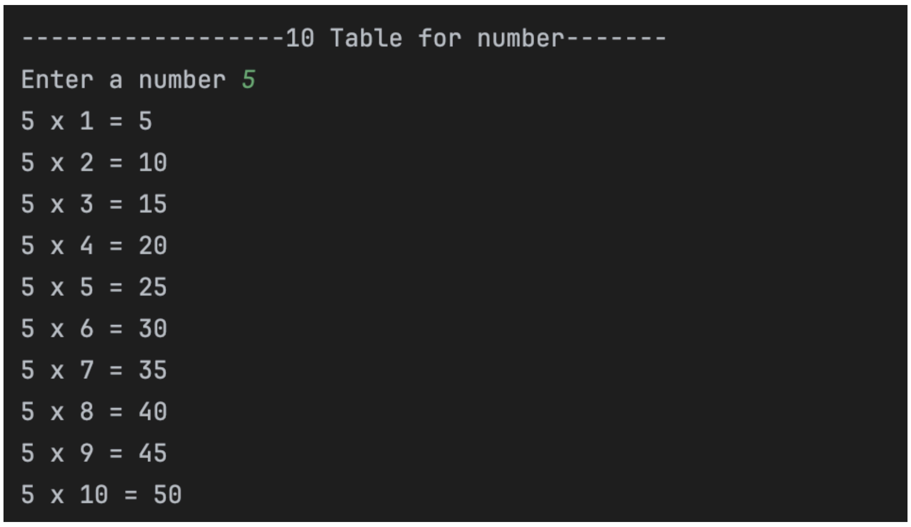

```
---- 8-lower case to upper case--------
Enter the word  sharma
  SHARMA
```

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

```
print("--------------9--prime numbers----------")
lower = 1
upper = 100

print("Prime numbers between", lower, "and", upper, "are:")

for num in range(lower, upper + 1):
   # all prime numbers are greater than 1
   if num > 1:
       for i in range(2, num):
           if (num % i) == 0:
               break
       else:
           print(num)
```


```
-------------9--prime numbers----------
Prime numbers between 1 and 100 are:
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
```

```
"""10.Print the table for a given number:
5*1=5
5 * 2 = 10.........."""

print("------------------10 Table for number-------")
num = int(input("Enter a number "))

# To take input from the user
# num = int(input("Display multiplication table of? "))

# Iterate 10 times from i = 1 to 10
for i in range(1, 11):
   print(num, 'x', i, '=', num*i)
```

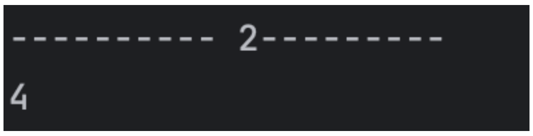


```
------------------10 Table for number-------
Enter a number 5
5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50
```

Experiment 5. String and sets

'''2. Count total number of vowels in a given string.'''

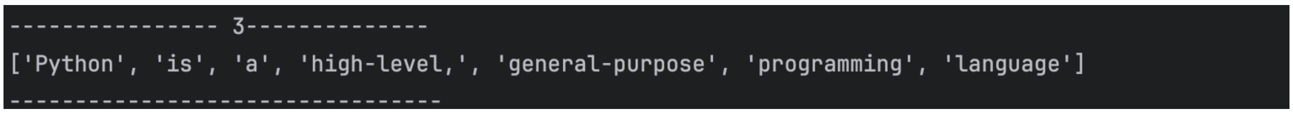
```
print("---------- 2---------")
string = "PythonProgramming"
vowels = "aeiouAEIOU"
count = sum(string.count(vowel) for vowel in vowels)
print(count)
```



```
---------- 2---------
4
```

'''3. Input a sentence and print words in separate lines. '''

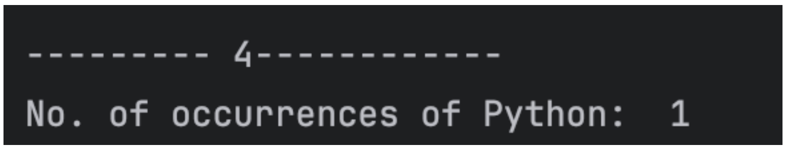
```
print("---------------- 3")
lst = "Python is a high-level, general-purpose programming language"
print(lst.split())
print("--------------------------------")
```



```
--------------- 3-------------
['Python', 'is', 'a', 'high-level,', 'general-purpose', 'programming', 'language']
--------------------------------
```

'''4. WAP to enter a string and a substring. You have to print the number of times that
the substring occurs in the given string. String traversal will take place from left to
right, not from right to left.
Sample Input
ABCDCDC
CDC
Sample Output '''

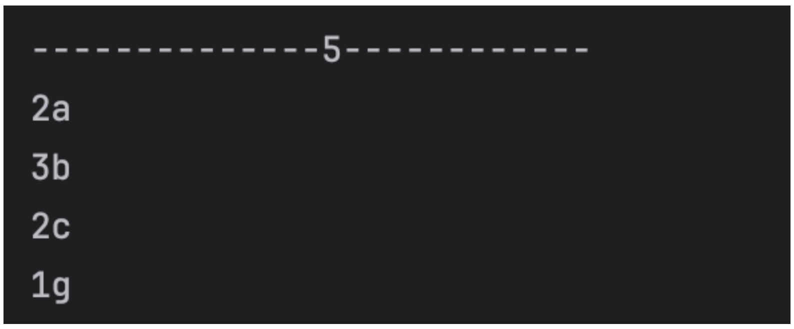
```
print("--------- 4------------")
txt = 'I love Python programming'
substring = 'Python'
count = txt.count(substring)
print('No. of occurrences of Python: ', count)
print("--------------------------------------")
```



```
--------- 4------------
No. of occurrences of Python:  1
```

'''5. Given a string containing both upper and lower case alphabets.
Write a Python
program to count the number of occurrences of each alphabet (case
insensitive)
and display the same.
Sample Input
ABaBCbGc
Sample Output
2A
3B
2C
1G '''

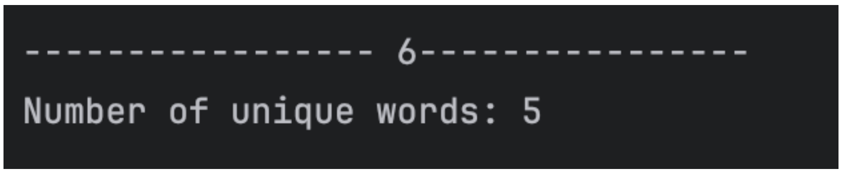
```
print("--------------5------------")
input_string = "ABaBCbGc"
input_string = input_string.lower()
alphabet_counts = {}
for char in input_string:
    if char.isalpha():
        alphabet_counts[char] = alphabet_counts.get(char, 0) + 1

for char, count in alphabet_counts.items():
    print(f"{count}{char}")
print("---------------------------------------------")
```



```
-------------5------------
2a
3b
2c
1g
```

'''6. Program to count number of unique words in a given sentence
using sets. '''

```
print("----------------- 6----------------")
sentence = "my name is janvi agrawal"
words = sentence.split()
unique_words = set(words)
num_unique_words = len(unique_words)
print("Number of unique words:", num_unique_words)
print("_______________________________________________")
```

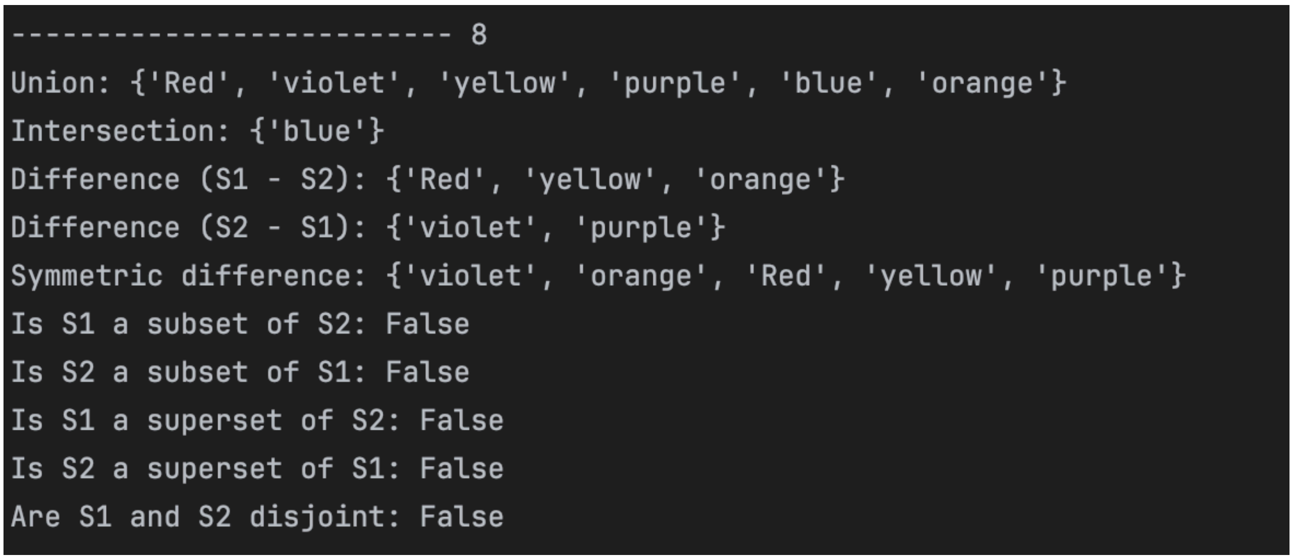


```
----------------- 6----------------
Number of unique words: 5
```

```
'''7. Create 2 sets s1 and s2 of n fruits each by taking input from user
and find:
a) Fruits which are in both sets s1 and s2
b) Fruits only in s1 but not in s2
c) Count of all fruits from s1 and s2 '''

print("------------------------7-----------------")
s1 = {"apple", "banana", "orange", "mango", "grape"}
s2 = {"mango", "grapes", "watermelon", "pineapple", "apple"}

common_fruits = s1.intersection(s2)
print("Fruits in both sets:", common_fruits)

fruits_only_in_s1 = s1.difference(s2)
print("Fruits only in s1:", fruits_only_in_s1)

total_fruits = len(s1.union(s2))
print("Count of all fruits:", total_fruits)
print("-----------------------------------")
```

```
'''8. Take two sets and apply various set operations on them :
S1 = {Red ,yellow, orange , blue }
S2 = {violet, blue , purple} '''

print("-------------------------- 8")
S1 = {"Red", "yellow", "orange", "blue"}
S2 = {"violet", "blue", "purple"}

# Union of S1 and S2
union = S1.union(S2)
print("Union:", union)

# Intersection of S1 and S2
intersection = S1.intersection(S2)
print("Intersection:", intersection)

# Difference of S1 and S2
difference_S1_S2 = S1.difference(S2)
print("Difference (S1 - S2):", difference_S1_S2)

# Difference of S2 and S1
difference_S2_S1 = S2.difference(S1)
print("Difference (S2 - S1):", difference_S2_S1)

# Symmetric difference of S1 and S2
symmetric_difference = S1.symmetric_difference(S2)
print("Symmetric difference:", symmetric_difference)

# Check for subset, superset, and disjointness
```

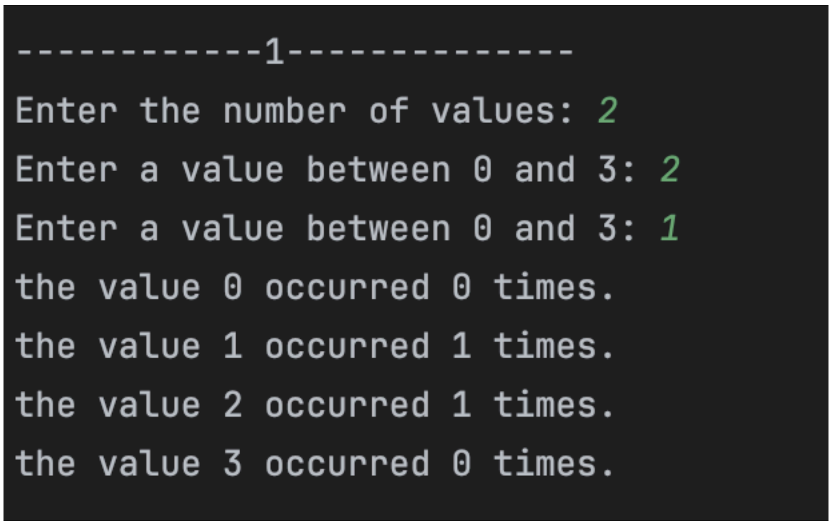
```
print("Is S1 a subset of S2:", S1.issubset(S2))
print("Is S2 a subset of S1:", S2.issubset(S1))
print("Is S1 a superset of S2:", S1.issuperset(S2))
print("Is S2 a superset of S1:", S2.issuperset(S1))
print("Are S1 and S2 disjoint:", S1.isdisjoint(S2))
print("-------------------------------------------")
```



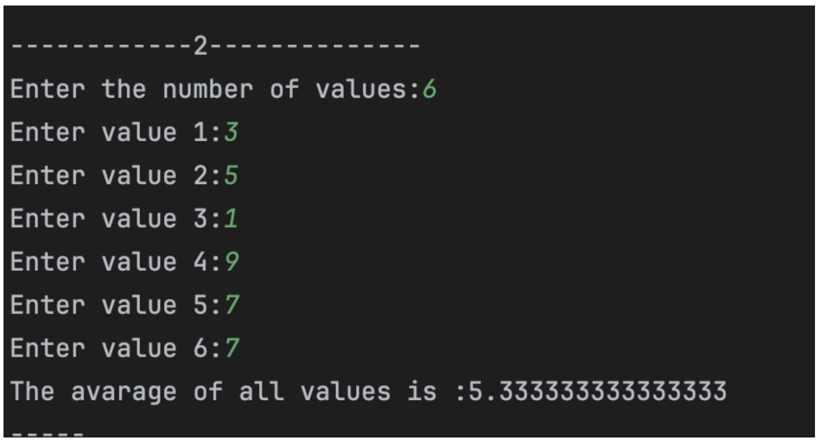
```
-------------------------- 8
Union: {'Red', 'violet', 'yellow', 'purple', 'blue', 'orange'}
Intersection: {'blue'}
Difference (S1 - S2): {'Red', 'yellow', 'orange'}
Difference (S2 - S1): {'violet', 'purple'}
Symmetric difference: {'violet', 'orange', 'Red', 'yellow', 'purple'}
Is S1 a subset of S2: False
Is S2 a subset of S1: False
Is S1 a superset of S2: False
Is S2 a superset of S1: False
Are S1 and S2 disjoint: False
```

#EXPERIMENT 6   list. Tuples and dictionary

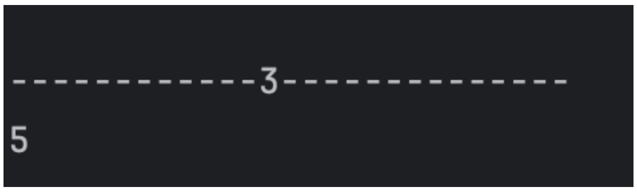
```
#EXPERIMENT 6

#1
print("-----\n\n------------1-------------")

n = int(input("Enter the number of values: "))
count=[0,0,0,0]
for i in range(n):
   value=int(input("Enter a value between 0 and 3: "))
   count[value]+=1

for i in range(4):
   print(f"the value {i} occurred {count[i]} times.”)
```


```
-----------1-------------
Enter the number of values: 2
Enter a value between 0 and 3: 2
Enter a value between 0 and 3: 1
the value 0 occurred 0 times.
the value 1 occurred 1 times.
the value 2 occurred 1 times.
the value 3 occurred 0 times.
```

```
#2
print("-----\n\n-----------2-------------")

n=int(input("Enter the number of values:"))
values=[]
for i in range(n):
    value=float(input(f"Enter value {i+1}:"))
    values.append(value)
values_tuple=tuple(values)
avg=sum(values)/len(values)
print(f"The avarage of all values is :{avg}”)
```


```
-----------2-------------
Enter the number of values:6
Enter value 1:3
Enter value 2:5
Enter value 3:1
Enter value 4:9
Enter value 5:7
Enter value 6:7
The avarage of all values is :5.333333333333333
-----
```

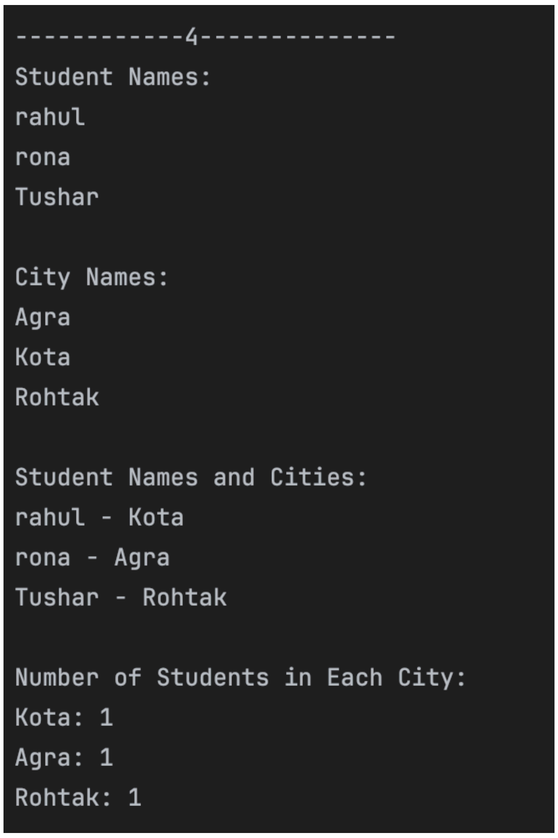
```
#3
print("-----\n\n------------3-------------")
N = 5
scores = [2, 3, 6, 6, 5]

unique_scores = list(set(scores))
unique_scores.sort(reverse=True)

runner_up_score = unique_scores[1]
print(runner_up_score)
```



```
------------3-------------
5
```

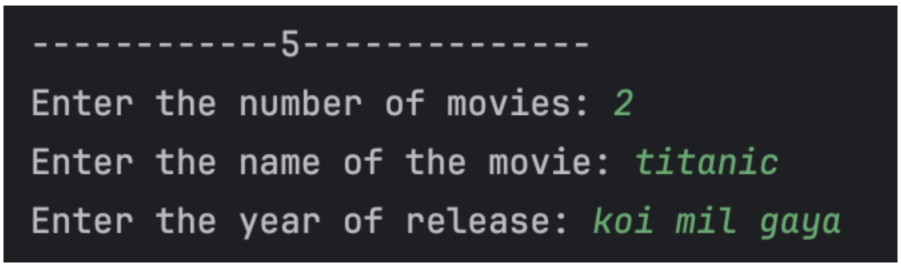
```
#4
print("-----\n\n------------4-------------")

students = {
    "rahul": "Kota",
    "rona": "Agra",
    "Tushar": "Rohtak",
}

print("Student Names:")
for name in students:
    print(name)

print("\nCity Names:")
city_set = set(students.values())  # Remove duplicates using a set
for city in city_set:
    print(city)


print("\nStudent Names and Cities:")
```

```
for name, city in students.items():
    print(f"{name} - {city}")


print("\nNumber of Students in Each City:")
from collections import Counter
city_counts = Counter(students.values())
for city, count in city_counts.items():
    print(f"{city}: {count}")
```



```
------------4-------------
Student Names:
rahul
rona
Tushar

City Names:
Agra
Kota
Rohtak

Student Names and Cities:
rahul - Kota
rona - Agra
Tushar - Rohtak

Number of Students in Each City:
Kota: 1
Agra: 1
Rohtak: 1
```

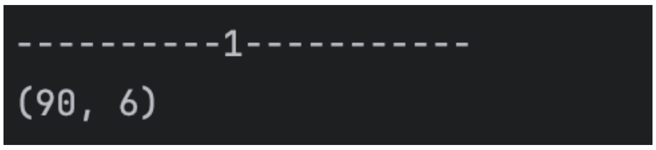
```
#5
print("\n------------5-------------")
n = int(input("Enter the number of movies: "))

movies = {}
for i in range(n):
    name = input("Enter the name of the movie: ")
    year = int(input("Enter the year of release: "))
    director = input("Enter the name of the director: ")
    production_cost = float(input("Enter the production cost: "))
    collection = float(input("Enter the collection made (earning): "))
    movies[name] = {
        "year": year,
        "director": director,
        "production_cost": production_cost,
        "collection": collection
    }

print("\nAll movie details:")
for name, details in movies.items():
    print(f"{name}: {details}")

print("\nMovies released before 2015:")
for name, details in movies.items():
    if details["year"] < 2015:
        print(name)

print("\nMovies that made a profit:")
for name, details in movies.items():
    if details["collection"] > details["production_cost"]:
        print(name)

director_name = input("\nEnter the name of a director to search for
their movies: ")
print(f"\nMovies directed by {director_name}:")
for name, details in movies.items():
    if details["director"] == director_name:
        print(name)
```



```
------------5--------------
Enter the number of movies: 2
Enter the name of the movie: titanic
Enter the year of release: koi mil gaya
```

Experiment 7 functions

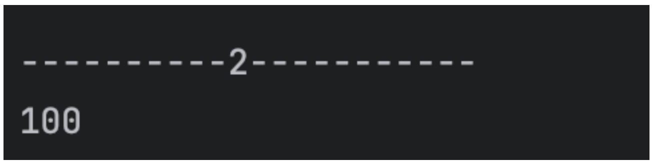
```
# EXPERIMENT-7

# 1)
print("\n\n----------1-----------")
def find_max_min(seq):
    if not seq:
        return None, None
    max_num = min_num = seq[0]
    for num in seq:
        if num > max_num:
            max_num = num
        if num < min_num:
            min_num = num
    return max_num, min_num


sample_input = [10, 6, 8, 90, 12, 56]
print(find_max_min(sample_input))
```



```
----------1-----------
(90, 6)
```

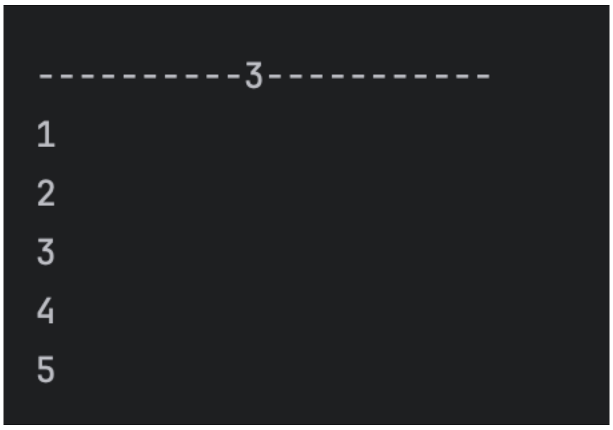
```
# 2)
def sum_cubes(n):
    return sum(i ** 3 for i in range(1, n))
```

```
print(sum_cubes(5))
```



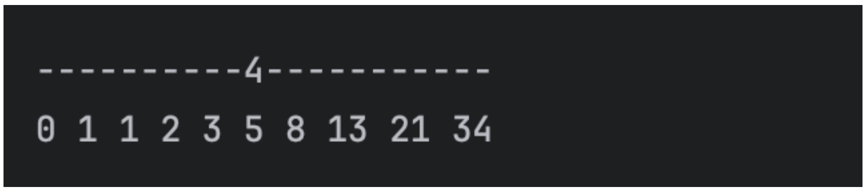
```
----------2-----------
100
```

```
# 3)
def print_numbers(n):
    if n > 0:
        print_numbers(n - 1)
        print(n)


print_numbers(5)
```



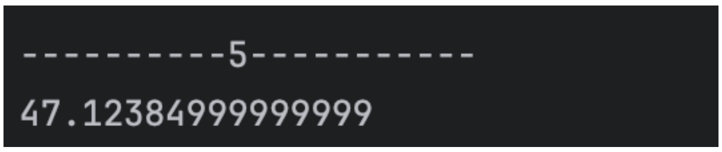
```
----------3-----------
1
2
3
4
5
```

```
# 4)
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n - 1) + fibonacci(n - 2)


for i in range(10):
    print(fibonacci(i), end=' ‘)
```



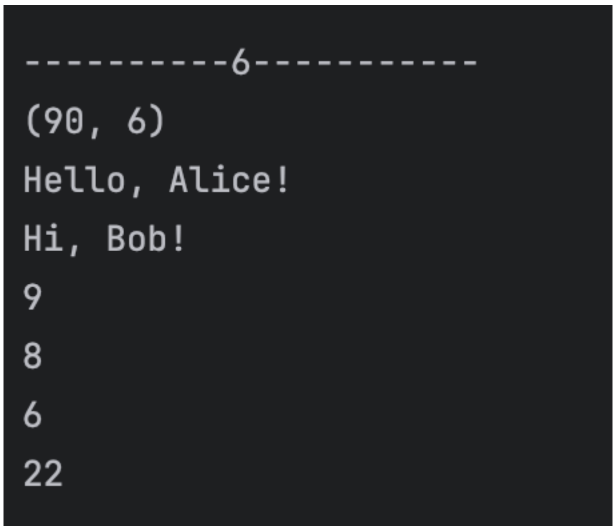
```
----------4-----------
0 1 1 2 3 5 8 13 21 34
```

```
# 5)
volume_of_cone = lambda r, h: (1 / 3) * 3.14159 * r ** 2 * h

print(volume_of_cone(3, 5))
```



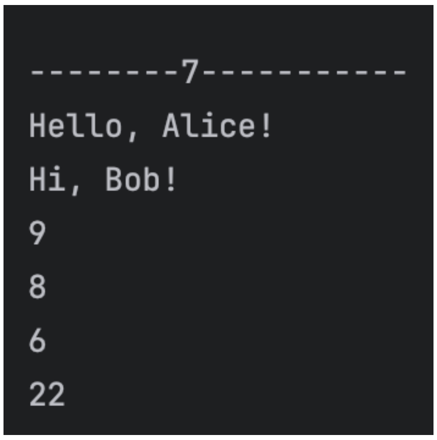
```
----------5-----------
47.12384999999999
```

```
# 6)
get_max_min = lambda lst: (max(lst), min(lst))

sample_input = [10, 6, 8, 90, 12, 56]
print(get_max_min(sample_input))
```



```
----------6-----------
(90, 6)
Hello, Alice!
Hi, Bob!
9
8
6
22
```

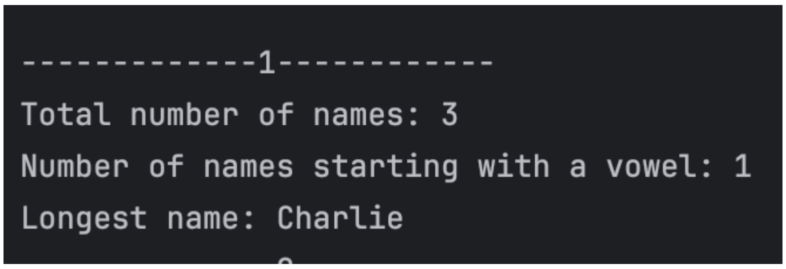
```
# 7)
# a) Keyword argument
def greet(name, message="Hello"):
    print(f"{message}, {name}!")


greet("Alice")
greet("Bob", message="Hi")


# b) Default argument
def exponentiate(base, power=2):
    return base ** power


print(exponentiate(3))
print(exponentiate(2, 3))


# c)Variable-length argument
def sum_values(*args):
    return sum(args)


print(sum_values(1, 2, 3))
print(sum_values(4, 5, 6, 7))
```



```
--------7-----------
Hello, Alice!
Hi, Bob!
9
8
6
22
```

Experiment 8. File handling

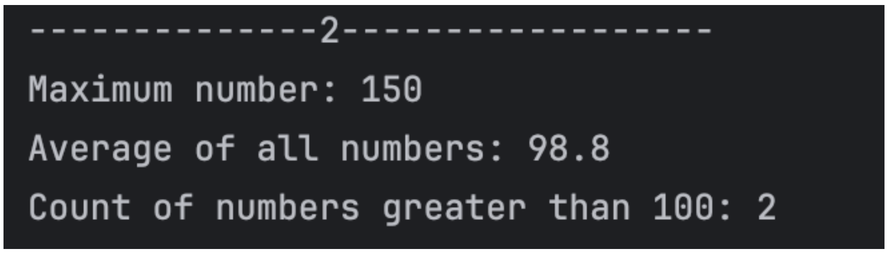
```
#Experiment-8
#1) add few names ,one names in each row ,in “name.txt.file”


print("\n\n-------------1------------")
with open("names.txt", "w") as file:
    file.write("Alice\n")
    file.write("Bob\n")
    file.write("Charlie\n")

with open("names.txt", "r") as file:
    names = file.readlines()
    num_names = len(names)

num_vowel_names = sum(1 for name in names if name.strip().lower()[0]
in 'aeiouAEIOU')

longest_name = max(names, key=len).strip()

print("Total number of names:", num_names)
print("Number of names starting with a vowel:", num_vowel_names)
print("Longest name:", longest_name)
```



```
-------------1------------
Total number of names: 3
Number of names starting with a vowel: 1
Longest name: Charlie
```

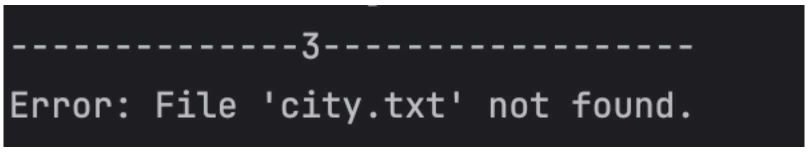
```
#2) store integers in  a file

print("-------------2----------------")
```

```
numbers = [56, 78, 90, 120, 150]
with open("integers.txt", "w") as file:
    file.write('\n'.join(map(str, numbers)))

with open("integers.txt", "r") as file:
    numbers = [int(num) for num in file.readlines()]

max_number = max(numbers)
average = sum(numbers) / len(numbers)
count_gt_100 = sum(1 for num in numbers if num > 100)

print("Maximum number:", max_number)
print("Average of all numbers:", average)
print("Count of numbers greater than 100:", count_gt_100)
```



```
-------------2-----------------
Maximum number: 150
Average of all numbers: 98.8
Count of numbers greater than 100: 2
```

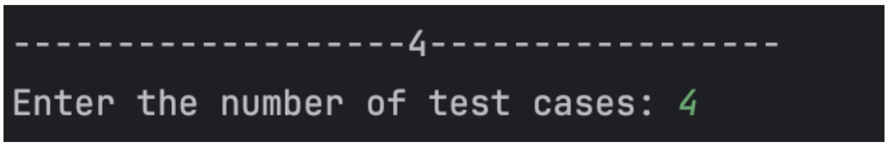
```
#3) assume a file city.txt with details of 5 city  in given format
print("--------------3-----------------")
try:
    with open("city.txt", "r") as file:
        cities = [line.strip().split() for line in file]

    print("Details of all cities:")
    for city in cities:
        print(city[0], "Population:", city[1], "Area:", city[2])

    print("\nCity names with population more than 10 Lakhs:")
    for city in cities:
        if float(city[1]) > 10:
            print(city[0])

    sum_areas = sum(float(city[2]) for city in cities)
    print("\nSum of areas of all cities:", sum_areas)

except FileNotFoundError:
    print("Error: File 'city.txt' not found.")
except ValueError:
    print("Error: Invalid data format in 'city.txt'.")
except IndexError:
    print("Error: Insufficient data in 'city.txt'. Ensure each city has name,
population, and area.")
except Exception as e:
    print("An unexpected error occurred:", e)
```



```
--------------3------------------
Error: File 'city.txt' not found.
```

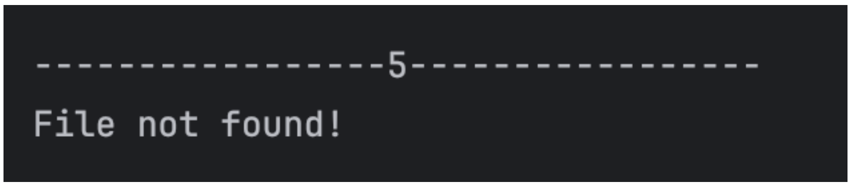
#4) input two value from user where the first line contains n the no of test cases

```
print("-------------------4-----------------")
def perform_division(a, b):
    try:
        result = int(a) / int(b)
        print(result)
    except ZeroDivisionError:
        print("Error Code: integer division or modulo by zero")
    except ValueError:
        print("Error Code: invalid literal for int() with base 10:", a)

N = int(input("Enter the number of test cases: "))
for _ in range(N):
    a, b = input().split()
    perform_division(a, b)
```



```
-------------------4-----------------
Enter the number of test cases: 4
```

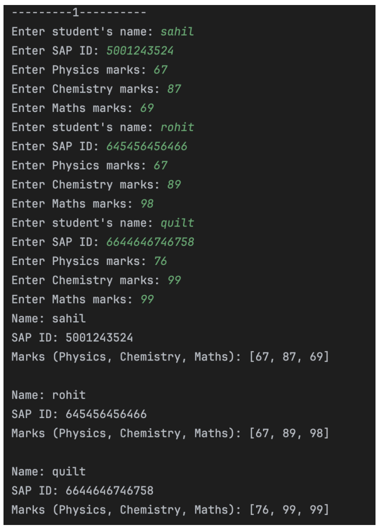
```
#5) create multiple suitable exceptions for a file handling.
print("------------------5-----------------")
try:
    with open("non_existing_file.txt", "r") as file:
        content = file.read()
except FileNotFoundError:
    print("File not found!")
except PermissionError:
    print("Permission denied to open the file!")
except IsADirectoryError:
    print("The provided path is a directory, not a file!")
except Exception as e:
    print("An unexpected error occurred:", e)
```



```
-----------------5-----------------
File not found!
```

Experiment 9 classes and objects

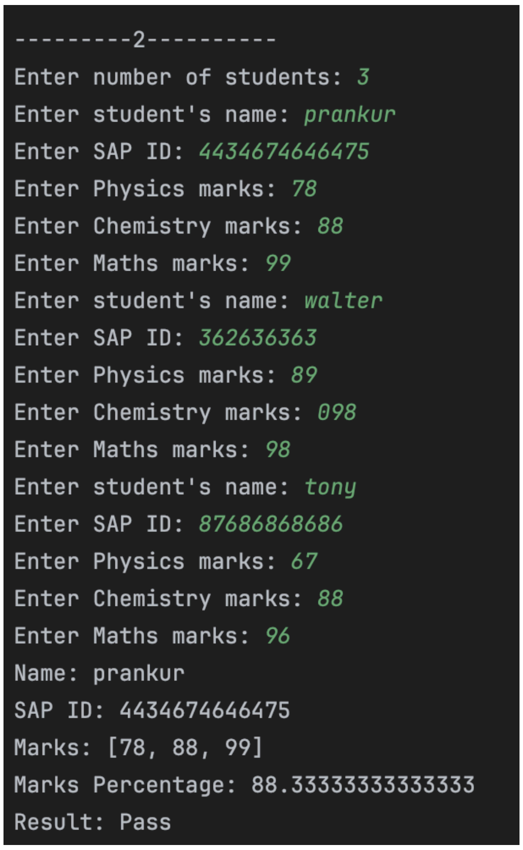
```
#1)
print("-\n\n---------1----------")
class Student:
    def __init__(self, name, sap_id, marks):
        self.name = name
        self.sap_id = sap_id
        self.marks = marks

    def display_details(self):
        print("Name:", self.name)
        print("SAP ID:", self.sap_id)
        print("Marks (Physics, Chemistry, Maths):", self.marks)


students = []
for i in range(3):
    name = input("Enter student's name: ")
    sap_id = input("Enter SAP ID: ")
    phy_marks = int(input("Enter Physics marks: "))
    chem_marks = int(input("Enter Chemistry marks: "))
    math_marks = int(input("Enter Maths marks: "))
    marks = [phy_marks, chem_marks, math_marks]
    students.append(Student(name, sap_id, marks))

for student in students:
    student.display_details()
    print()
```



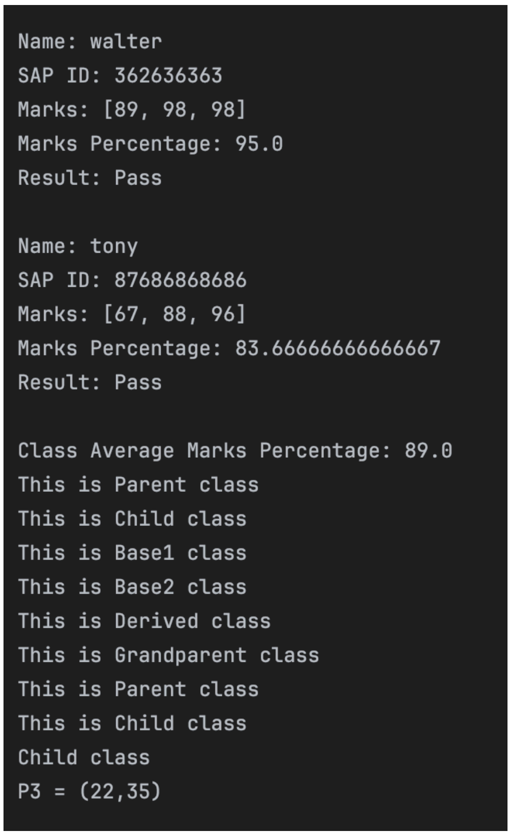
```
---------1----------
Enter student's name: sahil
Enter SAP ID: 5001243524
Enter Physics marks: 67
Enter Chemistry marks: 87
Enter Maths marks: 69
Enter student's name: rohit
Enter SAP ID: 645456456466
Enter Physics marks: 67
Enter Chemistry marks: 89
Enter Maths marks: 98
Enter student's name: quilt
Enter SAP ID: 6644646746758
Enter Physics marks: 76
Enter Chemistry marks: 99
Enter Maths marks: 99
Name: sahil
SAP ID: 5001243524
Marks (Physics, Chemistry, Maths): [67, 87, 69]

Name: rohit
SAP ID: 645456456466
Marks (Physics, Chemistry, Maths): [67, 89, 98]

Name: quilt
SAP ID: 6644646746758
Marks (Physics, Chemistry, Maths): [76, 99, 99]
```

```
#2)
class Student:
    def __init__(self, name, sap_id, marks):
        self.name = name
        self.sap_id = sap_id
        self.marks = marks

    def display(self):
        print("Name:", self.name)
        print("SAP ID:", self.sap_id)
```

```python
        print("Marks:", self.marks)

    def marks_percentage(self):
        total_marks = sum(self.marks)
        percentage = (total_marks / (len(self.marks) * 100)) * 100
        return percentage

    def result(self):
        if all(mark >= 40 for mark in self.marks):
            return "Pass"
        else:
            return "Fail"


def class_average(students):
    total_percentage = sum(student.marks_percentage() for student in
students)
    return total_percentage / len(students)


n = int(input("Enter number of students: "))
students = []
for i in range(n):
    name = input("Enter student's name: ")
    sap_id = input("Enter SAP ID: ")
    phy_marks = int(input("Enter Physics marks: "))
    chem_marks = int(input("Enter Chemistry marks: "))
    math_marks = int(input("Enter Maths marks: "))
    marks = [phy_marks, chem_marks, math_marks]
    students.append(Student(name, sap_id, marks))

for student in students:
    student.display()
    print("Marks Percentage:", student.marks_percentage())
    print("Result:", student.result())
    print()

print("Class Average Marks Percentage:", class_average(students))
```

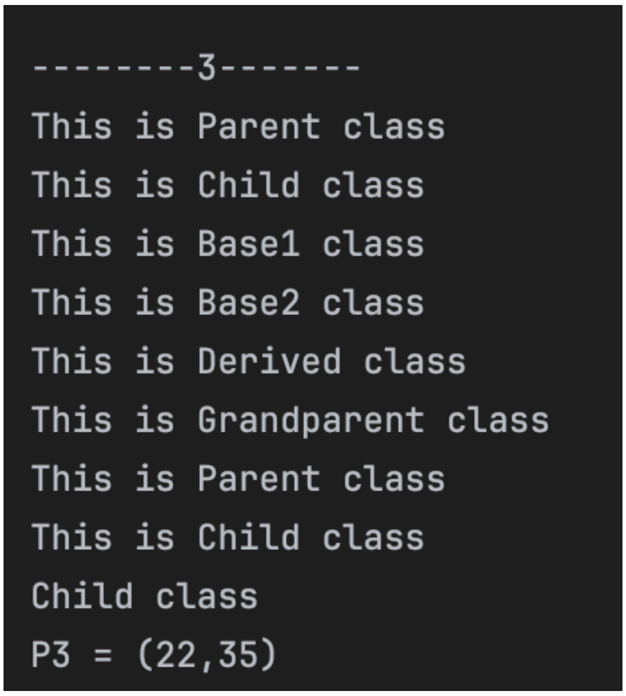


```
---------2----------
Enter number of students: 3
Enter student's name: prankur
Enter SAP ID: 4434674646475
Enter Physics marks: 78
Enter Chemistry marks: 88
Enter Maths marks: 99
Enter student's name: walter
Enter SAP ID: 362636363
Enter Physics marks: 89
Enter Chemistry marks: 098
Enter Maths marks: 98
Enter student's name: tony
Enter SAP ID: 87686868686
Enter Physics marks: 67
Enter Chemistry marks: 88
Enter Maths marks: 96
Name: prankur
SAP ID: 4434674646475
Marks: [78, 88, 99]
Marks Percentage: 88.33333333333333
Result: Pass
```



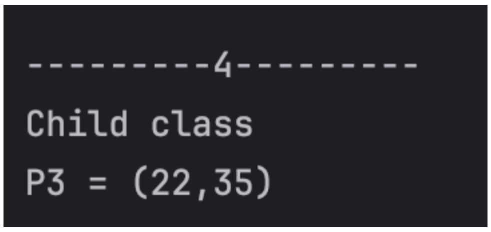
```
Name: walter
SAP ID: 362636363
Marks: [89, 98, 98]
Marks Percentage: 95.0
Result: Pass

Name: tony
SAP ID: 87686868686
Marks: [67, 88, 96]
Marks Percentage: 83.66666666666667
Result: Pass

Class Average Marks Percentage: 89.0
This is Parent class
This is Child class
This is Base1 class
This is Base2 class
This is Derived class
This is Grandparent class
This is Parent class
This is Child class
Child class
P3 = (22,35)
```

```
#3)
# Single Inheritance
class Parent:
    def show_parent(self):
        print("This is Parent class")


class Child(Parent):
    def show_child(self):
        print("This is Child class")


child_obj = Child()
child_obj.show_parent()  # Accessing parent class method from child 
class
child_obj.show_child()   # Accessing child class method


# Multiple Inheritance
class Base1:
    def show_base1(self):
        print("This is Base1 class")


class Base2:
    def show_base2(self):
        print("This is Base2 class")


class Derived(Base1, Base2):
    def show_derived(self):
        print("This is Derived class")


derived_obj = Derived()
derived_obj.show_base1()
derived_obj.show_base2()
derived_obj.show_derived()


class Grandparent:
    def show_grandparent(self):
        print("This is Grandparent class")


class Parent(Grandparent):
    def show_parent(self):
        print("This is Parent class")


class Child(Parent):
    def show_child(self):
        print("This is Child class")


child_obj = Child()
child_obj.show_grandparent()  # Accessing Grandparent class method
from Child class
child_obj.show_parent()       # Accessing Parent class method from
Child class
child_obj.show_child()
```



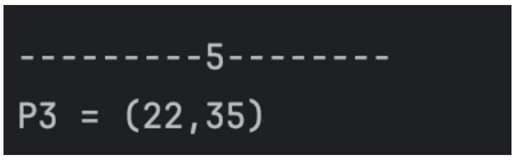
```
--------3-------
This is Parent class
This is Child class
This is Base1 class
This is Base2 class
This is Derived class
This is Grandparent class
This is Parent class
This is Child class
Child class
P3 = (22,35)
```

```
#4)
class Parent:
    def show(self):
        print("Parent class")


class Child(Parent):
    def show(self):
        print("Child class")


child_obj = Child()
child_obj.show()
```



```
---------4---------
Child class
P3 = (22,35)
```

```
#5)
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __add__(self, other):
        return Point(self.x + other.x, self.y + other.y)

    def __str__(self):
        return f"({self.x},{self.y})"


p1 = Point(10, 20)
p2 = Point(12, 15)
p3 = p1 + p2
print("P3 =", p3)
```

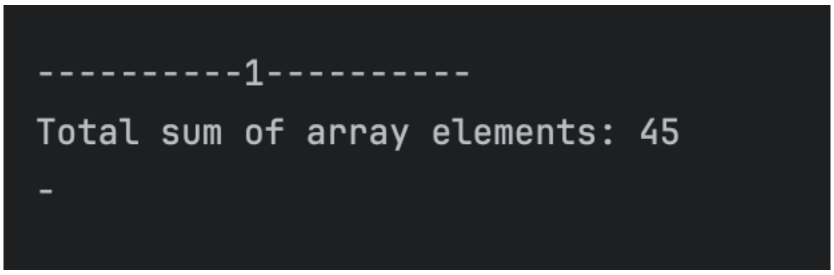


```
---------5--------
P3 = (22,35)
```

Experiment 10 data analysis

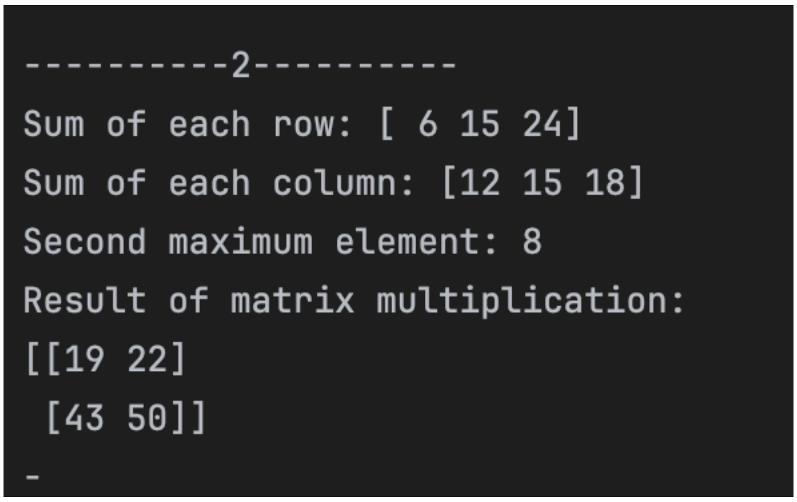
#Experiment-10

```
#1)
print("-\n\n----------1----------")
import numpy as np

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

total_sum = np.sum(arr)
print("Total sum of array elements:", total_sum)
```



```
----------1----------
Total sum of array elements: 45
-
```

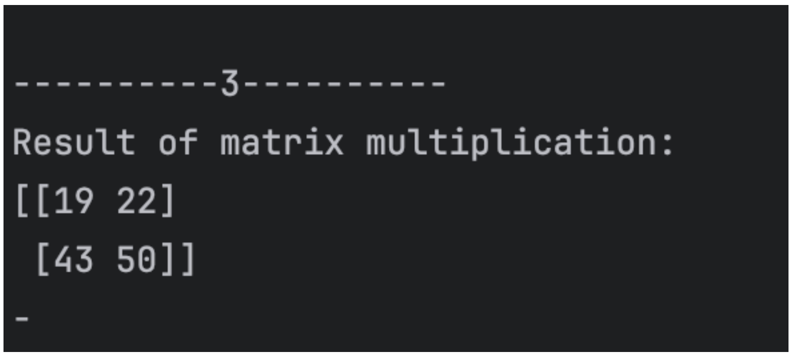
```
#2)
print("-\n\n----------2----------")
import numpy as np

arr_3x3 = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

row_sum = np.sum(arr_3x3, axis=1)
col_sum = np.sum(arr_3x3, axis=0)
print("Sum of each row:", row_sum)
print("Sum of each column:", col_sum)

second_max = np.partition(arr_3x3.flatten(), -2)[-2]
print("Second maximum element:", second_max)

mat1 = np.array([[1, 2], [3, 4]])
mat2 = np.array([[5, 6], [7, 8]])
```



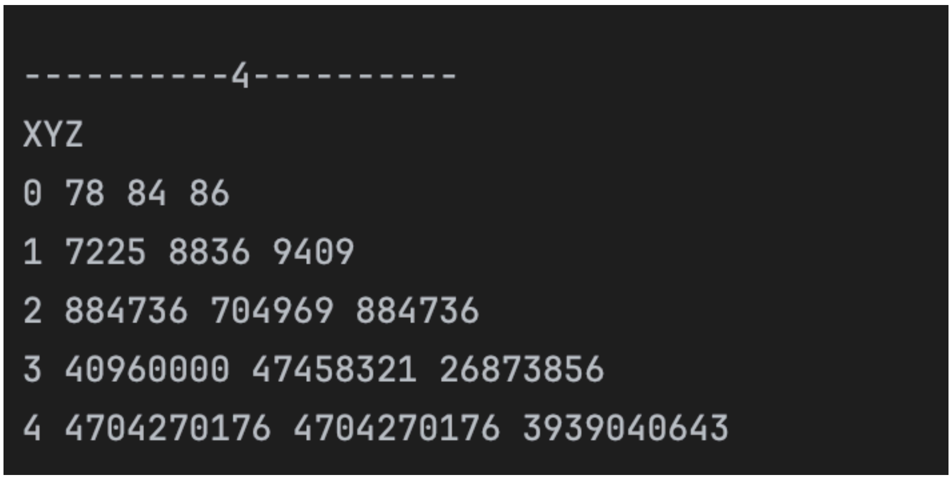
```
----------2----------
Sum of each row: [ 6 15 24]
Sum of each column: [12 15 18]
Second maximum element: 8
Result of matrix multiplication:
[[19 22]
 [43 50]]
-
```

```
result_matrix =
np.dot(mat1, mat2)
print("Result of
matrix
multiplication:")
print(result_matrix)
```

```
#3)
print("-\n\n----------3----------")
import numpy as np

def matrix_multiplication(mat1, mat2):
    result_matrix = np.dot(mat1, mat2)
    return result_matrix

mat1 = np.array([[1, 2], [3, 4]])
mat2 = np.array([[5, 6], [7, 8]])

result = matrix_multiplication(mat1, mat2)

print("Result of matrix multiplication:")
print(result)
```


```
----------3----------
Result of matrix multiplication:
[[19 22]
 [43 50]]
-
```

```
#4)
print("-\n\n----------4----------")
data = {'X': [78, 85, 96, 80, 86], 'Y': [84, 94, 89, 83, 86], 'Z': [86, 97,
96, 72, 83]}

arr = np.array([data[key] for key in data])

powers = np.array([[1, 2, 3, 4, 5]])

result = np.power(arr, powers)

print("XYZ")
for i, row in enumerate(result.T):
    print(i, *row)
```


```
----------4----------
XYZ
0 78 84 86
1 7225 8836 9409
2 884736 704969 884736
3 40960000 47458321 26873856
4 4704270176 4704270176 3939040643
```