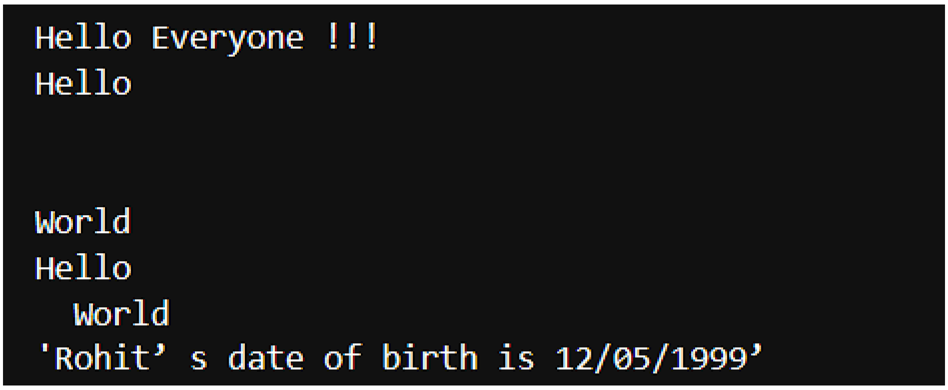
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Python Lab Experiments

Experiment 1 :-

Q.1)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\n  World")
print("Hello\n\n\nWorld")
print("Hello Everyone !!!")
```

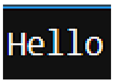


```
Hello Everyone !!!
Hello


World
Hello
  World
'Rohit’ s date of birth is 12/05/1999’
```

Q2.) Declare a string variable called x and assign it the value “Hello”.
Print out the value of x .

```
p = "Hello"
print(p)
```

Output:-



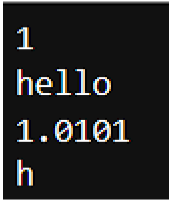
```
Hello
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Q3.)Take different data types and print values using print function.'''

```
x=1
y='hello'
h=1.0101
z='h'
print(x)
print(y)
print(h)
print(z)
```

Output:-



```
1
hello
1.0101
h
```

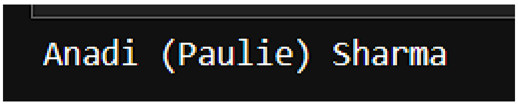
Q4.)Take two variable a and b. Assign your first name and last name. Print your Name after adding your First name and Last name together.

```
a="Anadi "
b="Sharma"
print(a+b)
```

```
Anadi Sharma
```

Q5.)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.

```
a="Anadi "
b=" Sharma"
c="Paulie"
print(a + "("+c+")" +b)
```



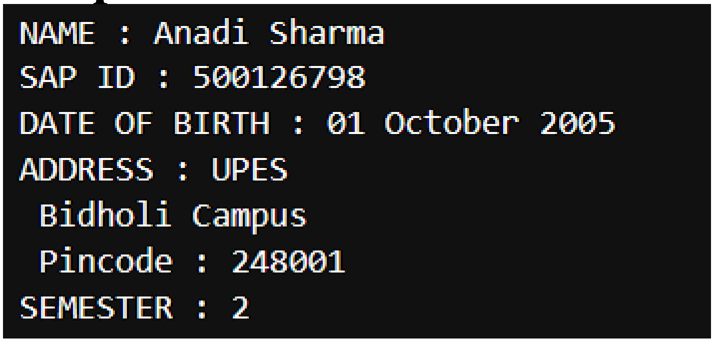
```
Anadi (Paulie) Sharma
```

Q6.)Declare and assign values to suitable variables and print in the following way :
NAME : NIKUNJ BANSAL
SAP ID : 500069944

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086
DATE OF BIRTH : 13 Oct 1999
ADDRESS : UPES
 Bidholi Campus
 Pincode : 248007
Programme : AI & ML
Semester : 2

```
n="Anadi Sharma"
sap="500126798"
dob="01 October 2005"
ad="UPES\n Bidholi Campus\n Pincode : 248001"
prog="AI & ML"
sem="2"

print("NAME : " +n)
print("SAP ID : "+sap)
print("DATE OF BIRTH : " +dob)
print("ADDRESS : " +ad)
print("SEMESTER : " +sem)
```

Output:-



```
NAME : Anadi Sharma
SAP ID : 500126798
DATE OF BIRTH : 01 October 2005
ADDRESS : UPES
 Bidholi Campus
 Pincode : 248001
SEMESTER : 2
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Experiment 2 :-

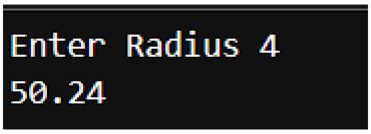
Q1. Declare these variables (x, y and z) as integers. Assign a value of 9 to x, Assign a value of 7 to y, perform addition, multiplication, division and subtraction on these two variables and Print out the result.

```
x=9
y=7
z=1
print(x+y)
print(x*y)
print(x-y)
```

```
16
63
2
```

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

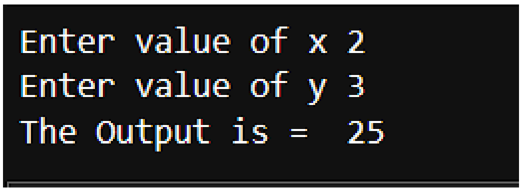
```
r=int(input("Enter Radius"))
ar=3.14*r*r
print(ar)
```



```
Enter Radius 4
50.24
```

'''3. Write a Python program to solve (x+y)*(x+y)
Test data : x = 4 , y = 3
Expected output: 49 '''

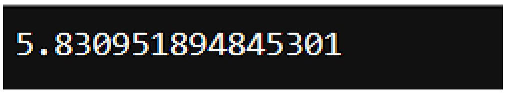
```
x=int(input("Enter value of x"))
y=int(input("Enter value of y"))
p=(x+y)*(x+y)
print("The Output is = ",p)
```



```
Enter value of x 2
Enter value of y 3
The Output is =  25
```

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

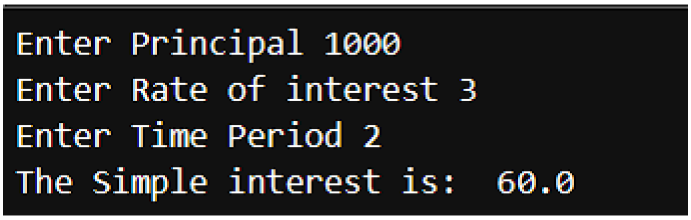
```
b=3
p=4
import math
h=math.sqrt((b*b)+(h*h))
print(h)
```



```
5.830951894845301
```

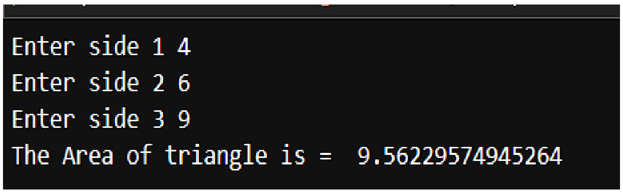
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
'''5. Write a program to find simple interest. '''
p=int(input("Enter Principal"))
r=int(input("Enter Rate of interest"))
t=int(input("Enter Time Period"))
si=(p*r*t)/100
print("The Simple interest is: ",si)
```



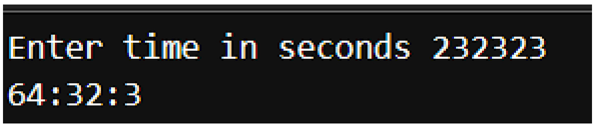
```
Enter Principal 1000
Enter Rate of interest 3
Enter Time Period 2
The Simple interest is:  60.0
```

```
'''6. Write a program to find area of triangle when length of sides are given.
'''
s1=int(input("Enter side 1"))
s2=int(input("Enter side 2"))
s3=int(input("Enter side 3"))
s=(s1+s2+s3)/2
import math
area=math.sqrt((s)*(s-s1)*(s-s2)*(s-s3))
print("The Area of triangle is = ",area)
```



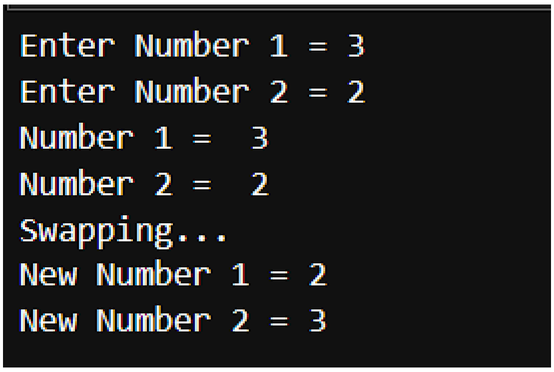
```
Enter side 1 4
Enter side 2 6
Enter side 3 9
The Area of triangle is =  9.56229574945264
```

```
'''7. Write a program to convert given seconds into hours, minutes and
remaining
seconds. '''
t=int(input("Enter time in seconds"))
hours=t//3600
t=t%3600
min=t//60
t=t%60
print(f"{hours}:{min}:{t}")
```



```
Enter time in seconds 232323
64:32:3
```

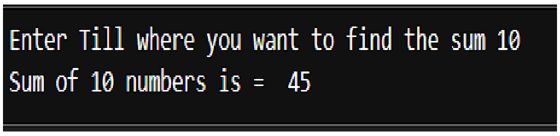
```
'''8. Write a program to swap two numbers without taking additional
variable. '''
a=int(input("Enter Number 1 ="))
b=int(input("Enter Number 2 ="))
print("Number 1 = ",a)
print("Number 2 = ",b)
print("Swapping...")
a,b=b,a
print(f"New Number 1 = {a}")
print(f"New Number 2 = {b}")
```



```
Enter Number 1 = 3
Enter Number 2 = 2
Number 1 =  3
Number 2 =  2
Swapping...
New Number 1 = 2
New Number 2 = 3
```

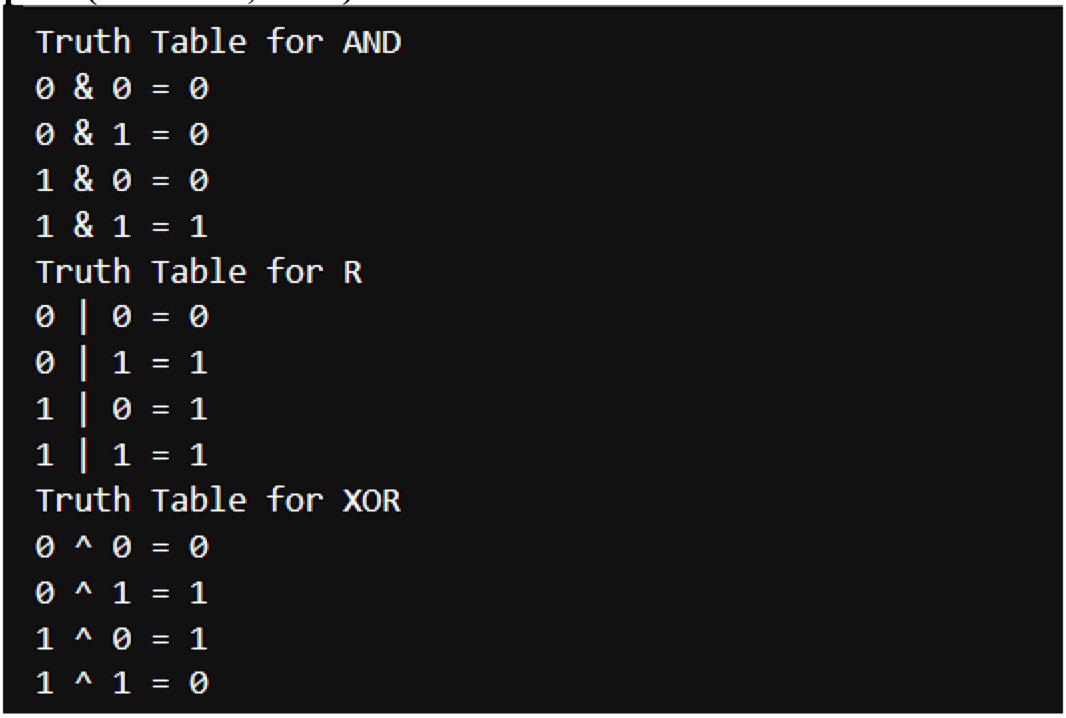
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
'''9. Write a program to find sum of first n natural numbers. '''
n=int(input("Enter Till where you want to find the sum"))
sum=0
for x in range(n):
   sum=sum+x
print(f"Sum of {n} numbers is = ",sum)
```



```
Enter Till where you want to find the sum 10
Sum of 10 numbers is =  45
```

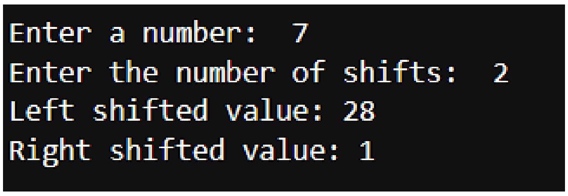
```
 '''10. Write a program to print truth table for bitwise operators( & , | and
^ operators) '''
print("Truth Table for AND")
print("0 & 0 =", 0 & 0)
print("0 & 1 =", 0 & 1)
print("1 & 0 =", 1 & 0)
print("1 & 1 =", 1 & 1)
print("Truth Table for R")
print("0 | 0 =", 0 | 0)
print("0 | 1 =", 0 | 1)
print("1 | 0 =", 1 | 0)
print("1 | 1 =", 1 | 1)
print("Truth Table for XOR")
print("0 ^ 0 =", 0 ^ 0)
print("0 ^ 1 =", 0 ^ 1)
print("1 ^ 0 =", 1 ^ 0)
print("1 ^ 1 =", 1 ^ 1)
```



```
Truth Table for AND
0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1
Truth Table for R
0 | 0 = 0
0 | 1 = 1
1 | 0 = 1
1 | 1 = 1
Truth Table for XOR
0 ^ 0 = 0
0 ^ 1 = 1
1 ^ 0 = 1
1 ^ 1 = 0
```

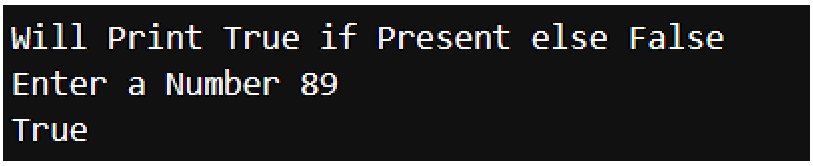
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
'''11. Write a program to find left shift and right shift values of a given
number.'''
n=int(input("Enter a number: "))
s=int(input("Enter the number of shifts: "))
l=n<<s
r=n>>s
print(f"Left shifted value: {l}")
print(f"Right shifted value: {r}")
```



```
Enter a number:  7
Enter the number of shifts:  2
Left shifted value: 28
Right shifted value: 1
```

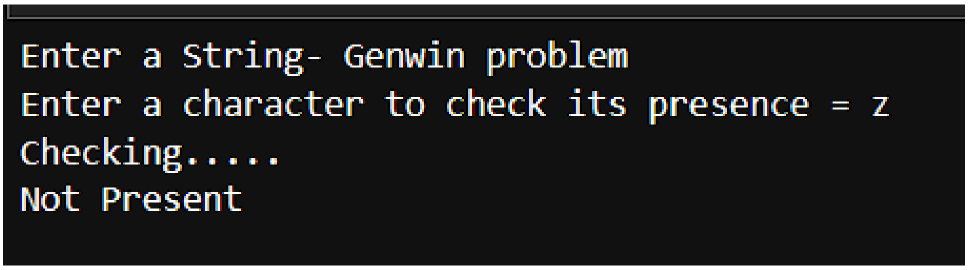
```
'''12. Using membership operator find whether a given number is in
sequence
(10,20,56,78,89) '''
t=(10,20,56,78,89)
print("Will Print True if Present
else False")
n=int(input("Enter a Number"))
print(n in t )
```



```
Will Print True if Present else False
Enter a Number 89
True
```

```
'''13.)Using membership operator find whether a given character is in a
string.'''
s=input("Enter a String-")
c=input("Enter a character to check its presence =")
print("Checking.....")
if (c in s) == True:
   print("Is Present")
else:
   print("Not Present")
```

Output:-



```
Enter a String- Genwin problem
Enter a character to check its presence = z
Checking.....
Not Present
```

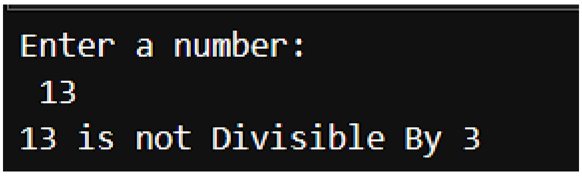
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Experiment 3 :-

'''1. Check whether given number is divisible by 3 and 5 both.
2. Check whether a given number is multiple of five or not.
3. Find the greatest among two numbers. If numbers are equal than print “numbers
are equal”.
4. Find the greatest among three numbers assuming no two values are same.
5. Check whether the quadratic equation has real roots or imaginary roots. Display
the roots.
6. Find whether a given year is a leap year or not.
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
8. Print the grade sheet of a student for the given range of cgpa. Scan marks of five
subjects and calculate the percentage.
CGPA=percentage/10
CGPA range:
0 to 3.4 -> F
3.5 to 5.0->C+
5.1 to 6->B
6.1 to 7-> B+
7.1 to 8-> A
8.1 to 9->A+
9.1 to 10-> O (Outstanding)
Sample Gradesheet
Name: Rohit Sharma
Roll Number: R17234512 SAPID: 50005673
Sem: 1 Course: B.Tech. CSE AI&ML
Subject name: Marks
PDS: 70
Python: 80
Chemistry: 90
English: 60
Physics: 50

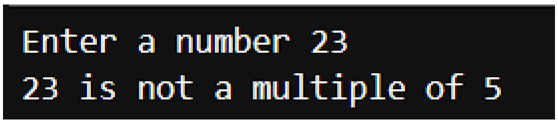
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086
Percentage: 70%
CGPA:7.0
Grade: ""

```
#Q1.)
#Solution
print("Enter a number: ")
n=int(input())
if n%3==0:
   if n%5==0:
       print(f"Yes, {n} is divisible by both 3 & 5")
   else:
      print(n,"is not divisible by 5 ")
else:
   print(n,"is not Divisible By 3")
```



```
Enter a number:
 13
13 is not Divisible By 3
```

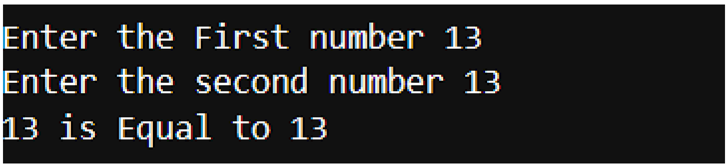
```
#Q2.)
#Solution
n=int(input("Enter a number"))
if n%5==0:
   print(f"Yes {n} is a multiple of 5")
else:
   print(f"{n} is not a multiple of 5")
```



```
Enter a number 23
23 is not a multiple of 5
```

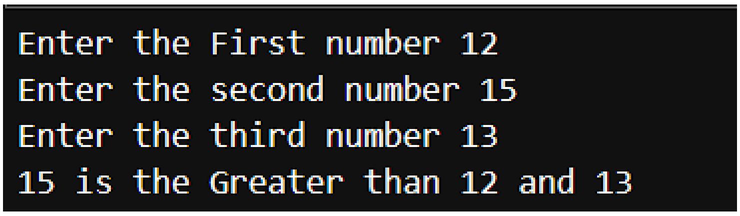
```
#Q3.)
#Solution
n1=int(input("Enter the First number"))
n2=int(input("Enter the second number"))
if n1>n2:
     print(f"{n1} is Greater than {n2}")
elif n1<n2:
          print(f"{n2} is Greater than {n1}")
else:
               print(f"{n1} is Equal to {n2}")
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



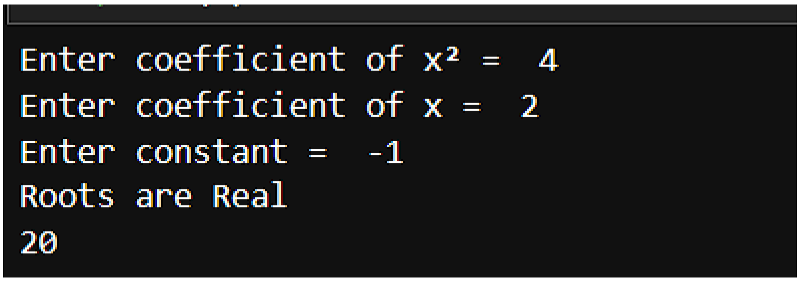
```
Enter the First number 13
Enter the second number 13
13 is Equal to 13
```

```
#Q4.)
#Solution
n1=int(input("Enter the First number"))
n2=int(input("Enter the second number"))
n3=int(input("Enter the third number"))
if n1>n2 and n1>n3:
   print(f"{n1} is the Greater than {n2} and {n3}")
elif n2>n1 and n2>n3:
    print(f"{n2} is the Greater than {n1} and {n3}")
else:
    print(f"{n3} is the Greater than {n1} and {n2}")
```



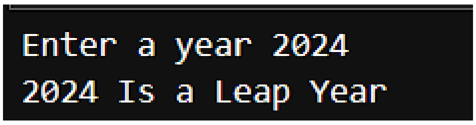
```
Enter the First number 12
Enter the second number 15
Enter the third number 13
15 is the Greater than 12 and 13
```

```
#Q5.)
#Solution
a=int(input("Enter coefficient of x² = "))
b=int(input("Enter coefficient of x = "))
c=int(input("Enter constant = "))
x=((b*b)-(4*a*c))
if x<0:
   print("Roots are imaginary")
   print(x)
else:
   print("Roots are Real")
   print(x)
```



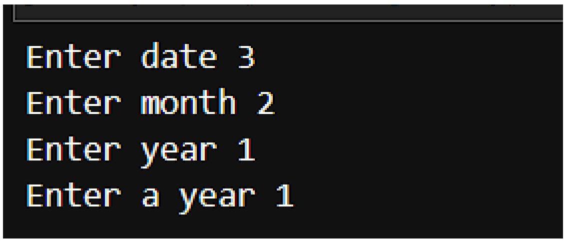
```
Enter coefficient of x² =  4
Enter coefficient of x =  2
Enter constant =  -1
Roots are Real
20
```

```
#Q6.)
#Solution
y=int(input("Enter a year"))
if y%4 == 0:
    if y%100 == 0:
            if y%400 == 0:
                print(y,"Is a Leap year")
            else:
                print(y, "Is Not a Leap Year")
    else:
        print(y,"Is a Leap Year")
else:
    print(y,"Is not Not a Leap Year")
```



```
Enter a year 2024
2024 Is a Leap Year
```

```
#Q7.)
#Solution
d=int(input("Enter date"))
m=int(input("Enter month"))
y=int(input("Enter year"))
```



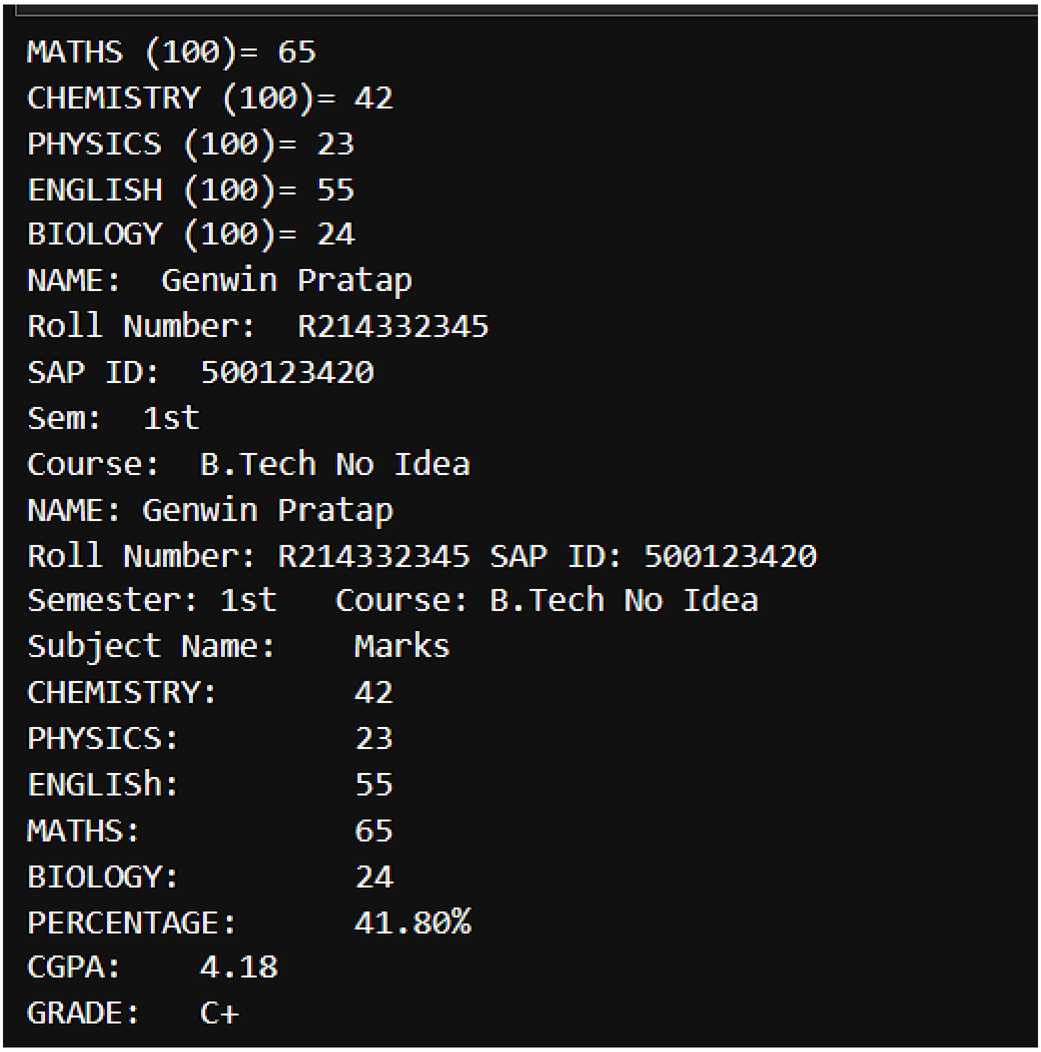
```
Enter date 3
Enter month 2
Enter year 1
Enter a year 1
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q8.)
#Solution
import sys
maths=int(input("MATHS (100)="))
chem=int(input("CHEMISTRY (100)="))
phy=int(input("PHYSICS (100)="))
eng=int(input("ENGLISH (100)="))
bio=int(input("BIOLOGY (100)="))
if (maths>100) or (chem>100) or (phy>100) or (eng>100) or (bio>100):
    print("Marks cannot be greater than 100")
else:
    per=float((((maths+chem+phy+eng+bio)/500)*100))
    cgpa=per/10
    if (cgpa>=0) and (cgpa<=3.4):
        g='F'
    elif (cgpa>=3.5) and (cgpa<=5):
        g='C+'
    elif (cgpa>=5.1) and (cgpa<=6):
        g='B'
    elif (cgpa>=6.1) and (cgpa<=7):
        g='B+'
    elif (cgpa>=7.1) and (cgpa<=8):
        g='A'
    elif (cgpa>=8.1) and (cgpa<=9):
        g='A+'
    elif (cgpa>=9.1) and (cgpa<=10):
        g='O'
name=input("NAME: ")
roll,sap=input("Roll Number: "),input("SAP ID: ")
sem,cou=input("Sem: "),input("Course: ")
print(f"NAME: {name}")
print("Roll Number: "+roll+"\tSAP ID: "+sap)
print("Semester: " + sem + "\tCourse: " + cou)
print(f"Subject Name:\t Marks ")
print(f"CHEMISTRY:\t {chem}")
print(f"PHYSICS:\t {phy}")
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
print(f"ENGLISh:\t {eng}")
print(f"MATHS:\t\t {maths}")
print(f"BIOLOGY:\t {bio}")
print(f"PERCENTAGE:\t {per:.2f}%")
print(f"CGPA:\t {cgpa}")
print(f"GRADE:\t {g}")
```



```
MATHS (100)= 65
CHEMISTRY (100)= 42
PHYSICS (100)= 23
ENGLISH (100)= 55
BIOLOGY (100)= 24
NAME:  Genwin Pratap
Roll Number:  R214332345
SAP ID:  500123420
Sem:  1st
Course:  B.Tech No Idea
NAME: Genwin Pratap
Roll Number: R214332345 SAP ID: 500123420
Semester: 1st   Course: B.Tech No Idea
Subject Name:    Marks
CHEMISTRY:       42
PHYSICS:         23
ENGLISh:         55
MATHS:           65
BIOLOGY:         24
PERCENTAGE:      41.80%
CGPA:    4.18
GRADE:   C+
```

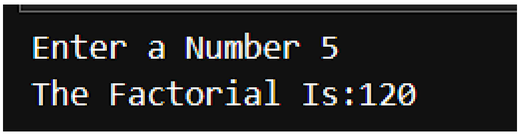
Experiment 4:-

'''1. Find a factorial of given number.
2. Find whether the given number is Armstrong number.
3. Print Fibonacci series up to given term.
4. Write a program to find if given number is prime number or not.

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

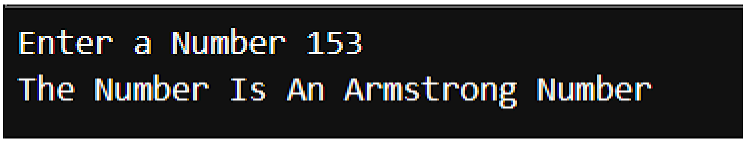
5. Check whether given number is palindrome or not.
6. Write a program to print sum of digits.
7. Count and print all numbers divisible by 5 or 7 between 1 to 100.
8. Convert all lower cases to upper case in a string.
9. Print all prime numbers between 1 and 100.
10. Print the table for a given number:
5 * 1 = 5
5 * 2 = 10…… '''

```
#Q1.)
x=int(input("Enter a Number"))
f=1
for i in range(1,x+1):
   f=f*i
print(f"The Factorial Is:{f}")
```



```
Enter a Number 5
The Factorial Is:120
```

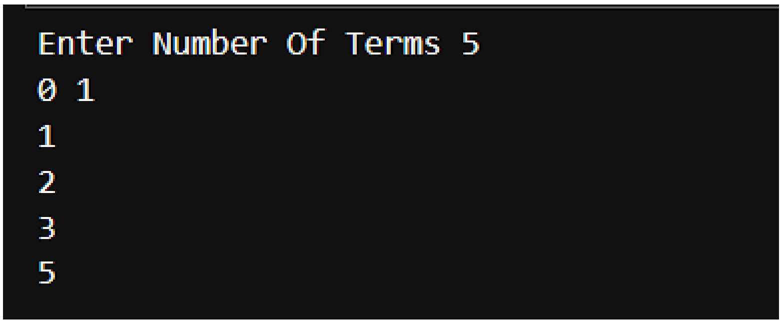
```
#Q2.)
x=int(input("Enter a Number"))
dup=x
sum=0
while x!=0:
   d=x%10
   sum=sum+(d*d*d)
   x=x//10
if dup==sum:
   print("The Number Is An Armstrong Number")
else:
   print("The Number Is Not An Armstrong Number")
```



```
Enter a Number 153
The Number Is An Armstrong Number
```

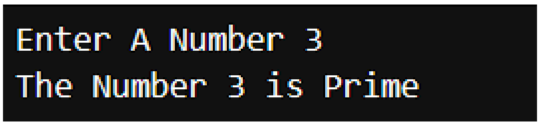
```
#Q3.)
n=int(input("Enter Number Of Terms"))
n1=0
n2=1
print(n1,n2)
```

```
for i in range(2,n+1):
   n3=n1+n2
   print(n3)
   n1=n2
   n2=n3
```



```
Enter Number Of Terms 5
0 1
1
2
3
5
```

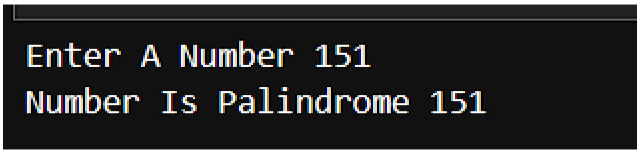
```
#Q4.)
x=int(input("Enter A Number"))
count=0
for i in range(1,x+1):
     if(x%i==0):
          count+=1
if(count==2):
     print(f"The Number {x} is Prime")
else:
      print(f"The Number {x} is Not Prime")
```



```
Enter A Number 3
The Number 3 is Prime
```

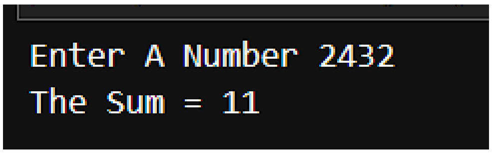
```
#Q5.)
x=int(input("Enter A Number"))
dup=x
rev=0
while dup!=0:
   d=dup%10
   rev=(rev*10)+d
   dup=dup//10
if(rev==x):
   print(f"Number Is Palindrome {x}")
else:
   print(f"Number Is Not Palindrome {x}")
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



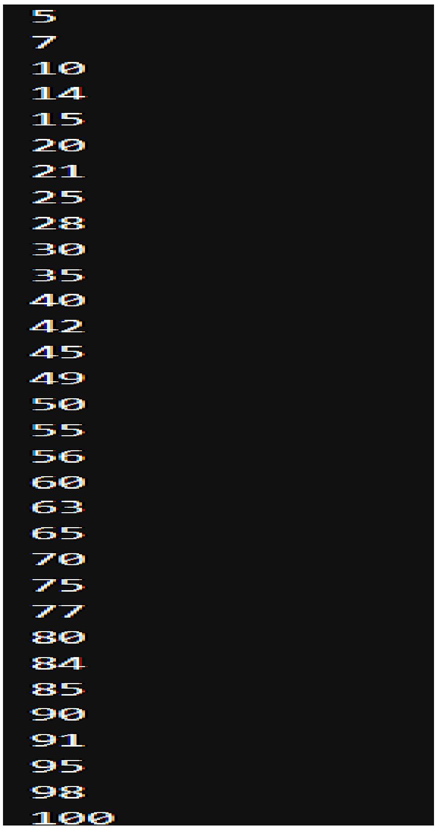
```
Enter A Number 151
Number Is Palindrome 151
```

```
#Q6.)
x=int(input("Enter A Number"))
sum=0
while x!=0:
    d=x%10
    sum=sum+d
    x=x//10
print(f"The Sum = {sum}")
```



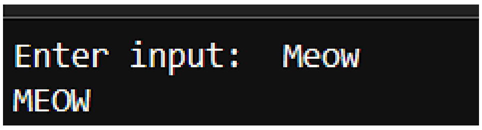
```
Enter A Number 2432
The Sum = 11
```

```
#Q7.)
for i in range(1,101):
    if i%5 == 0:
        print(i)
    elif i%7==0:
        print(i)
```



```
5
7
10
14
15
20
21
25
28
30
35
40
42
45
49
50
55
56
60
63
65
70
75
77
80
84
85
90
91
95
98
100
```

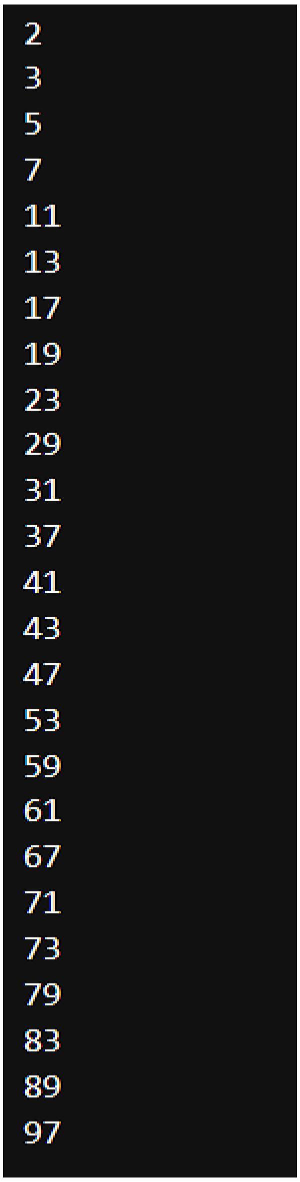
```
#Q8.)
x=input("Enter input: ")
y=x.upper()
print(y)
```



```
Enter input:  Meow
MEOW
```

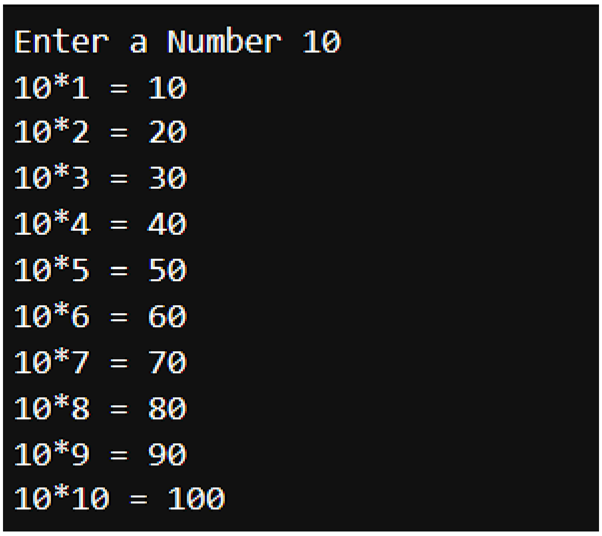
```
#Q9.)
for i in range(2,101):
    c=0
    for j in range(1,i+1):
```

```
        if i%j == 0:
            c=c+1
    if c==2:
        print(i)
```



```
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
```

```
#Q10.)
n=int(input("Enter a Number"))
for i in range(1,11):
    p=i*n
    print(f"{n}*{i} = {p}")
```



```
Enter a Number 10
10*1 = 10
10*2 = 20
10*3 = 30
10*4 = 40
10*5 = 50
10*6 = 60
10*7 = 70
10*8 = 80
10*9 = 90
10*10 = 100
```

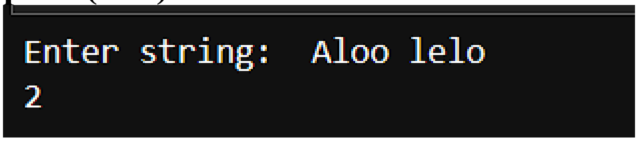
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Experiment 5:-

'''1. Write a program to count and display the number of capital letters in a given string.
2. Count total number of vowels in a given string.
3. Input a sentence and print words in separate lines.
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
2
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
6. Program to count number of unique words in a given sentence using sets.
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
8. Take two sets and apply various set operations on them :
S1 = {Red ,yellow, orange , blue }
S2 = {violet, blue , purple}'''

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q1.)
s=input("Enter string: ")
add=0
for i in range(len(s)):
    if(s[i]==s[i].upper()):
        add+=1
print(add)
```

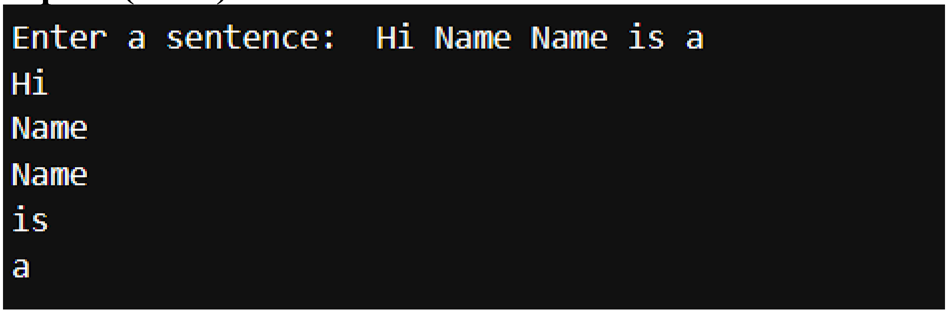


```
Enter string:  Aloo lelo
2
```

```
#Q2.)
s=input("Enter string: ")
vowels = 'aeiouAEIOU'
count = 0
for char in s:
    if char in vowels:
        count += 1
print("Total number of vowels:", count)
```

```
Enter string:  Mew Meow Potato
Total number of vowels: 6
```

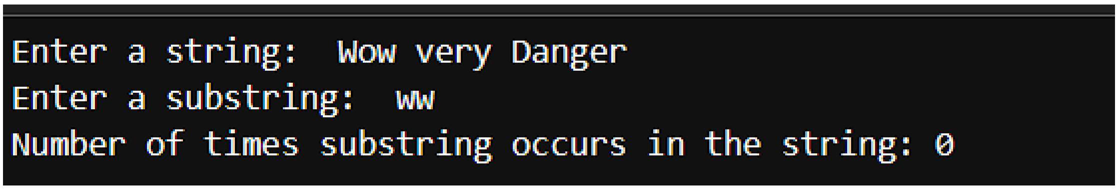
```
#Q3.)
s = input("Enter a sentence: ")
# Split the sentence into words
words = s.split()
# Print each word on a separate line
for word in words:
    print(word)
```



```
Enter a sentence:  Hi Name Name is a
Hi
Name
Name
is
a
```

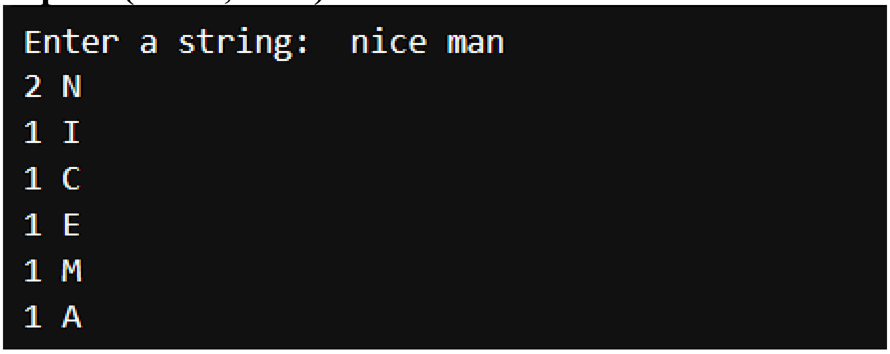
```
#Q4.)
user_string = input("Enter a string: ")
user_substring = input("Enter a substring: ")
```

```
count = user_string.count(user_substring)
print("Number of times substring occurs in the string:", count)
```



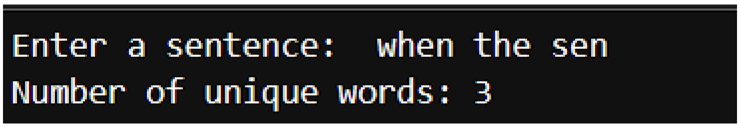
```
Enter a string:  Wow very Danger
Enter a substring:  ww
Number of times substring occurs in the string: 0
```

```
#Q5.)
user_input = input("Enter a string: ")
char_frequency = {}
for char in user_input:
    if char.isalpha():
        char = char.upper()
        if char in char_frequency:
            char_frequency[char] += 1
        else:
            char_frequency[char] = 1

for char, count in char_frequency.items():
    print(count, char)
```



```
Enter a string:  nice man
2 N
1 I
1 C
1 E
1 M
1 A
```

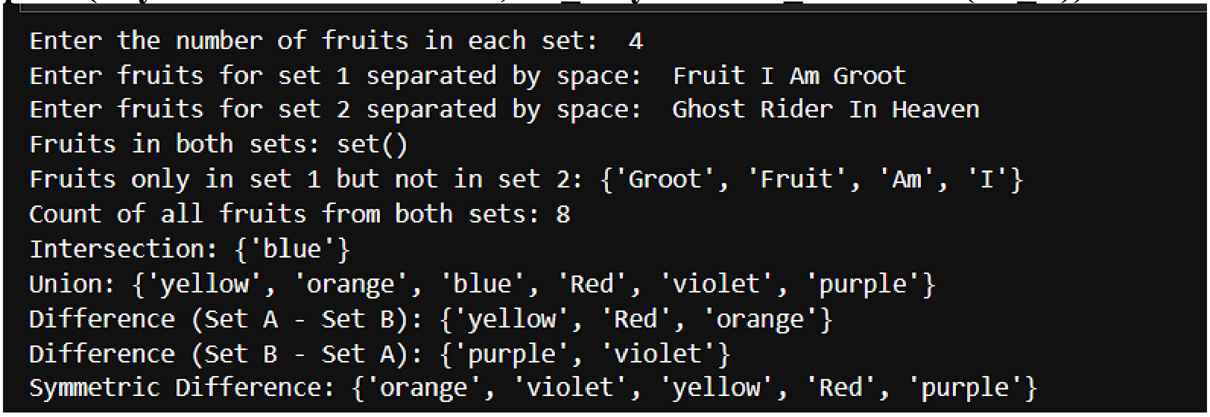
```
#Q6.)
user_sentence = input("Enter a sentence: ")
word_list = user_sentence.split()
unique_word_set = set(word_list)
print("Number of unique words:", len(unique_word_set))
```



```
Enter a sentence:  when the sen
Number of unique words: 3
```

```
#Q7.)
num_fruits = int(input("Enter the number of fruits in each set: "))
set1 = set(input("Enter fruits for set 1 separated by space: ").split())
```

```
set2 = set(input("Enter fruits for set 2 separated by space: ").split())
print("Fruits in both sets:", set1.intersection(set2))
print("Fruits only in set 1 but not in set 2:", set1.difference(set2))
print("Count of all fruits from both sets:", len(set1.union(set2)))
# Set operations on predefined sets
set_A = {'Red', 'yellow', 'orange', 'blue'}
set_B = {'violet', 'blue', 'purple'}
print("Intersection:", set_A.intersection(set_B))
print("Union:", set_A.union(set_B))
print("Difference (Set A - Set B):", set_A.difference(set_B))
print("Difference (Set B - Set A):", set_B.difference(set_A))
print("Symmetric Difference:", set_A.symmetric_difference(set_B))
```



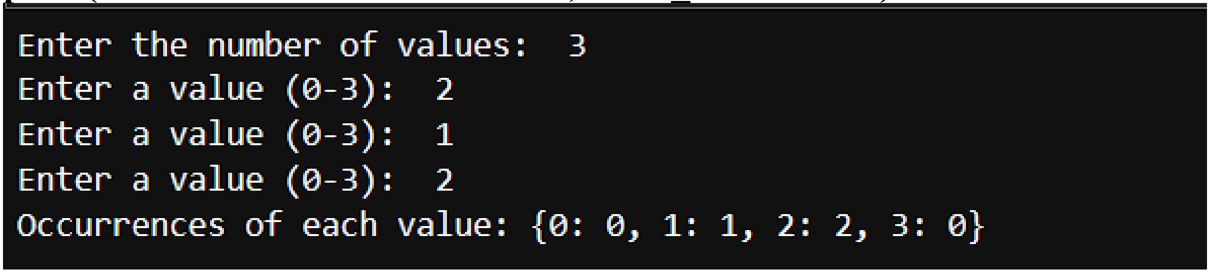
```
Enter the number of fruits in each set:  4
Enter fruits for set 1 separated by space:  Fruit I Am Groot
Enter fruits for set 2 separated by space:  Ghost Rider In Heaven
Fruits in both sets: set()
Fruits only in set 1 but not in set 2: {'Groot', 'Fruit', 'Am', 'I'}
Count of all fruits from both sets: 8
Intersection: {'blue'}
Union: {'yellow', 'orange', 'blue', 'Red', 'violet', 'purple'}
Difference (Set A - Set B): {'yellow', 'Red', 'orange'}
Difference (Set B - Set A): {'purple', 'violet'}
Symmetric Difference: {'orange', 'violet', 'yellow', 'Red', 'purple'}
```

Experiment 6:-

1. Scan n values in range 0-3 and print the number of times each value has occurred.
2. Create a tuple to store n numeric values and find average of all values.
3. WAP to input a list of scores for N students in a list data type. Find the score of the runner-up and print the output. Sample Input N = 5 Scores= 2 3 6 6 5 Sample output 5 Note: Given list is [2, 3, 6, 6, 5]. The maximum score is 6, second maximum is 5. Hence, we print 5 as the runner-up score.
4. Create a dictionary of n persons where key is name and value is city. a) Display all names b) Display all city names c) Display student name and city of all students. d) Count number of students in each city.
5. Store details of n movies in a dictionary by taking input from the user. Each movie must store details like name, year, director name, production cost, collection made (earning) & perform the following :- a) print all movie details b) display name of movies released before 2015 c) print movies that made a profit. d) print movies directed by a particular director

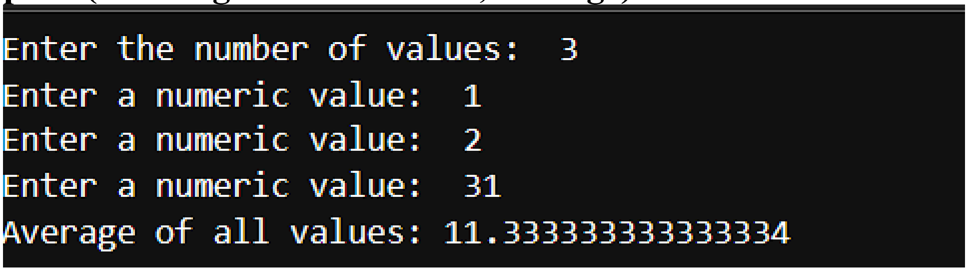
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q1.)
num_values = int(input("Enter the number of values: "))
value_occurrences = {0: 0, 1: 0, 2: 0, 3: 0}
for _ in range(num_values):
    value = int(input("Enter a value (0-3): "))
    if value in value_occurrences:
        value_occurrences[value] += 1
print("Occurrences of each value:", value_occurrences)
```



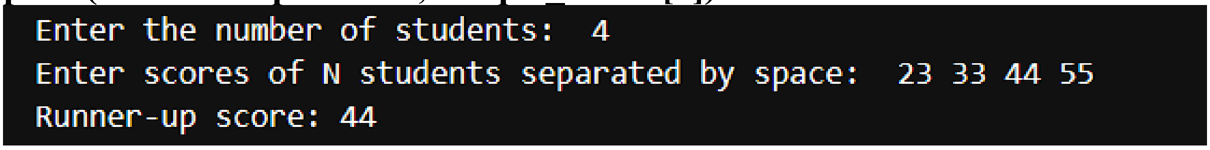
```
Enter the number of values:  3
Enter a value (0-3):  2
Enter a value (0-3):  1
Enter a value (0-3):  2
Occurrences of each value: {0: 0, 1: 1, 2: 2, 3: 0}
```

```
#Q2.)
n = int(input("Enter the number of values: "))
 values = tuple(float(input("Enter a numeric value: ")) for _ in range(n))
average = sum(values) / n
print("Average of all values:", average)
```



```
Enter the number of values:  3
Enter a numeric value:  1
Enter a numeric value:  2
Enter a numeric value:  31
Average of all values: 11.333333333333334
```

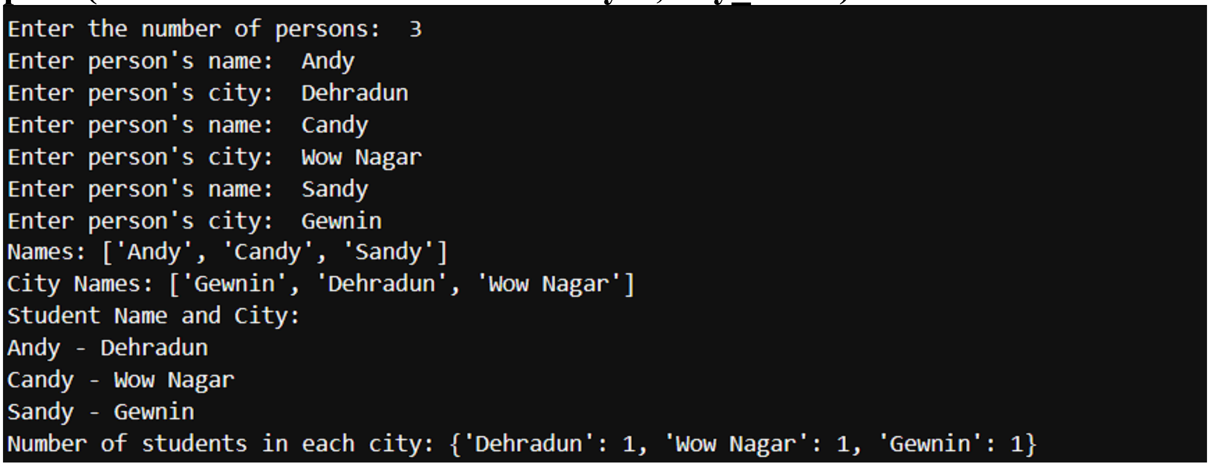
```
#Q3.)
num_students = int(input("Enter the number of students: "))
student_scores = list(map(int, input("Enter scores of N students separated by space: ").split()))
unique_scores = list(set(student_scores))
unique_scores.sort(reverse=True)
print("Runner-up score:", unique_scores[1])
```



```
Enter the number of students:  4
Enter scores of N students separated by space:  23 33 44 55
Runner-up score: 44
```

```
#Q4.)
num_persons = int(input("Enter the number of persons: "))
person_details = {}
for _ in range(num_persons):
```

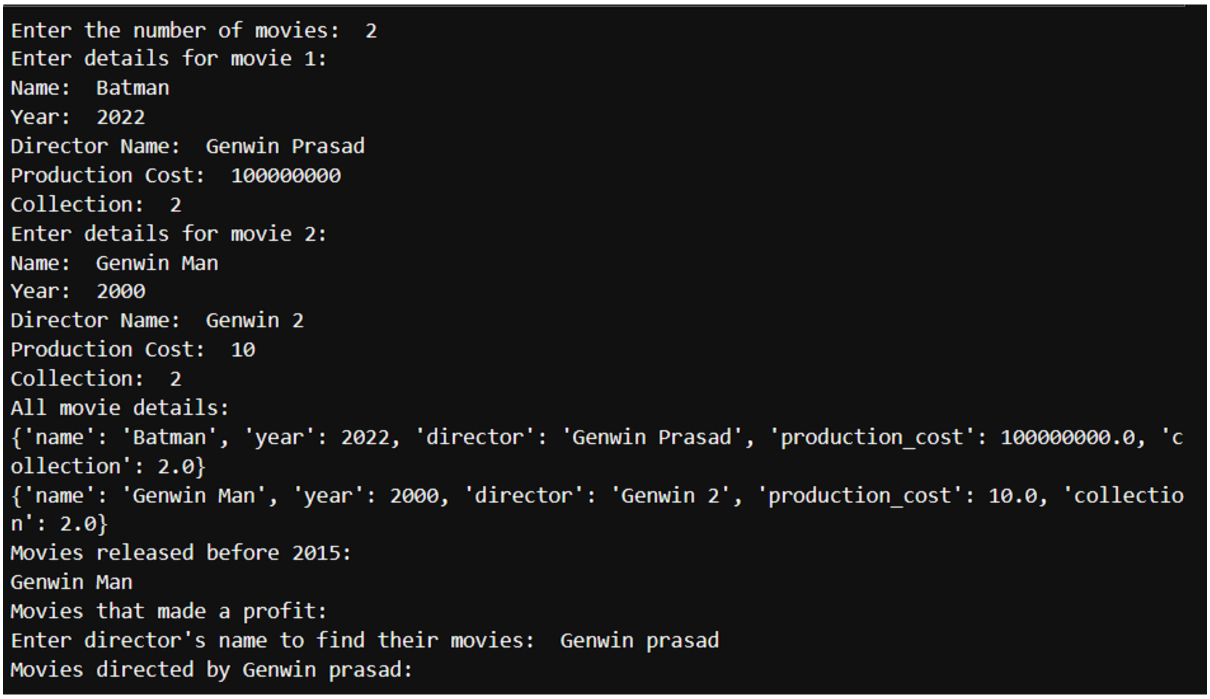
```
    name = input("Enter person's name: ")
    city = input("Enter person's city: ")
    person_details[name] = city

# a) Display all names
print("Names:", list(person_details.keys()))

# b) Display all city names
print("City Names:", list(set(person_details.values())))

# c) Display student name and city of all students.
print("Student Name and City:")
for name, city in person_details.items():
    print(name, "-", city)

# d) Count the number of students in each city.
city_count = {}
for city in person_details.values():
    if city in city_count:
        city_count[city] += 1
    else:
        city_count[city] = 1
print("Number of students in each city:", city_count)
```



```
Enter the number of persons:  3
Enter person's name:  Andy
Enter person's city:  Dehradun
Enter person's name:  Candy
Enter person's city:  Wow Nagar
Enter person's name:  Sandy
Enter person's city:  Gewnin
Names: ['Andy', 'Candy', 'Sandy']
City Names: ['Gewnin', 'Dehradun', 'Wow Nagar']
Student Name and City:
Andy - Dehradun
Candy - Wow Nagar
Sandy - Gewnin
Number of students in each city: {'Dehradun': 1, 'Wow Nagar': 1, 'Gewnin': 1}
```

```
#Q5.)
num_movies = int(input("Enter the number of movies: "))
movies_list = []
for i in range(num_movies):
    movie_details = {}
    print(f"Enter details for movie {i+1}:")
    movie_details['name'] = input("Name: ")
```

```
    movie_details['year'] = int(input("Year: "))
    movie_details['director'] = input("Director Name: ")
    movie_details['production_cost'] = float(input("Production Cost: "))
    movie_details['collection'] = float(input("Collection: "))
    movies_list.append(movie_details)

# a) Print all movie details
print("All movie details:")
for movie in movies_list:
    print(movie)

# b) Display names of movies released before 2015
print("Movies released before 2015:")
for movie in movies_list:
    if movie['year'] < 2015:
        print(movie['name'])

# c) Print movies that made a profit
print("Movies that made a profit:")
for movie in movies_list:
    if movie['collection'] > movie['production_cost']:
        print(movie['name'])

# d) Print movies directed by a particular director
director_name = input("Enter director's name to find their movies: ")
print(f"Movies directed by {director_name}:")
for movie in movies_list:
    if movie['director'] == director_name:
        print(movie['name'])
```



```
Enter the number of movies:  2
Enter details for movie 1:
Name:  Batman
Year:  2022
Director Name:  Genwin Prasad
Production Cost:  100000000
Collection:  2
Enter details for movie 2:
Name:  Genwin Man
Year:  2000
Director Name:  Genwin 2
Production Cost:  10
Collection:  2
All movie details:
{'name': 'Batman', 'year': 2022, 'director': 'Genwin Prasad', 'production_cost': 100000000.0, 'c
ollection': 2.0}
{'name': 'Genwin Man', 'year': 2000, 'director': 'Genwin 2', 'production_cost': 10.0, 'collectio
n': 2.0}
Movies released before 2015:
Genwin Man
Movies that made a profit:
Enter director's name to find their movies:  Genwin prasad
Movies directed by Genwin prasad:
```

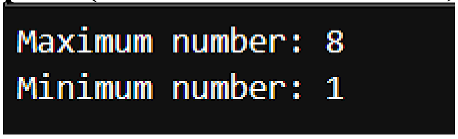
Experiment 7:-

1. Write a Python function to find the maximum and minimum numbers from a sequence of numbers. (Note: Do not use built-in functions.)
2. Write a Python function that takes a positive integer and returns the sum of the cube of all the positive integers smaller than the specified number.
3. Write a Python function to print 1 to n using recursion. (Note: Do not use loop)
4. Write a recursive function to print Fibonacci series upto n terms.
5. Write a lambda function to find volume of cone.
6. Write a lambda function which gives tuple of max and min from a list. Sample input: [10, 6, 8, 90, 12, 56] Sample output: (90,6)
7. Write functions to explain mentioned concepts: a. Keyword argument b. Default argument c. Variable length argument

```
#Q1.)
def find_max_min(sequence):
   # Check if the sequence is empty
   if not sequence:
      return None, None
   # Initialize maximum and minimum values with the first element of the
sequence
   maximum = sequence[0]
   minimum = sequence[0]
```

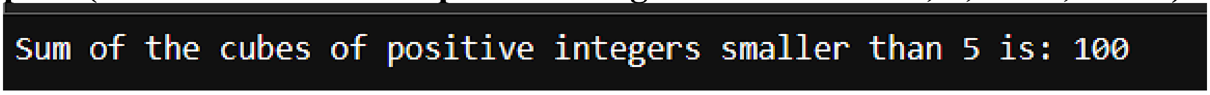
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
    # Traverse through the sequence to find maximum and minimum values
    for num in sequence:
        if num > maximum:
            maximum = num
        elif num < minimum:
            minimum = num
    return maximum, minimum
# Example usage:
sequence = [3, 7, 2, 8, 1, 5, 4]
max_num, min_num = find_max_min(sequence)
print("Maximum number:", max_num)
print("Minimum number:", min_num)
```



```
Maximum number: 8
Minimum number: 1
```

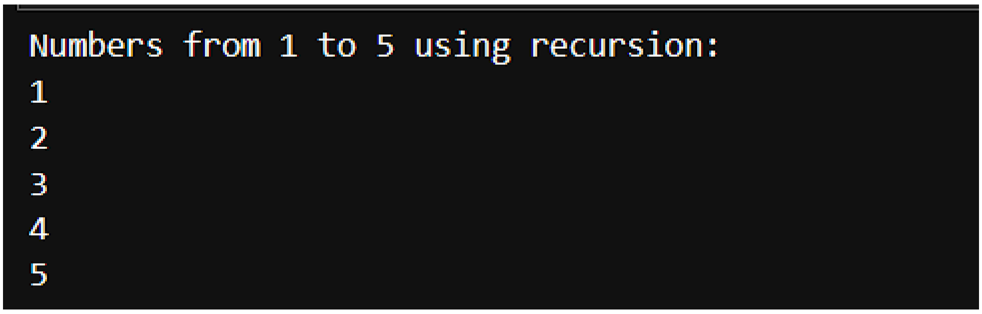
```
#Q2.)
def sum_cube_positive_integers(n):
    # Initialize sum to store the sum of cubes
    sum_of_cubes = 0
    # Iterate from 1 to n-1 and add the cube of each number to the sum
    for i in range(1, n):
        sum_of_cubes += i ** 3
    return sum_of_cubes
# Example usage:
n = 5
result = sum_cube_positive_integers(n)
print("Sum of the cubes of positive integers smaller than", n, "is:", result)
```



```
Sum of the cubes of positive integers smaller than 5 is: 100
```

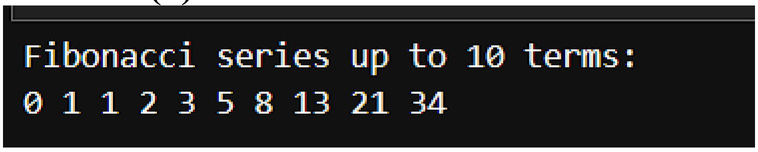
```
#Q3.)
def print_numbers(n):
    if n >= 1:
        print_numbers(n - 1)  # Recursively call the function with n-1
        print(n)              # Print the current number
# Example usage:
n = 5
print("Numbers from 1 to", n, "using recursion:")
print_numbers(n)
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



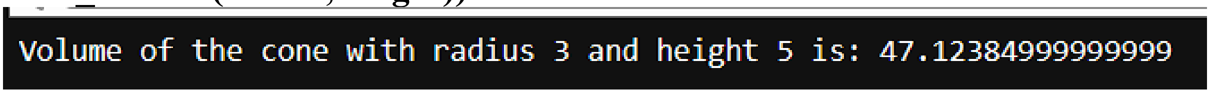
```
Numbers from 1 to 5 using recursion:
1
2
3
4
5
```

```
#Q4.)
def fibonacci(n, a=0, b=1):
   if n <= 0:
      return
   print(a, end=" ")
   fibonacci(n-1, b, a+b)
# Example usage:
n = 10
print("Fibonacci series up to", n, "terms:")
fibonacci(n)
```



```
Fibonacci series up to 10 terms:
0 1 1 2 3 5 8 13 21 34
```

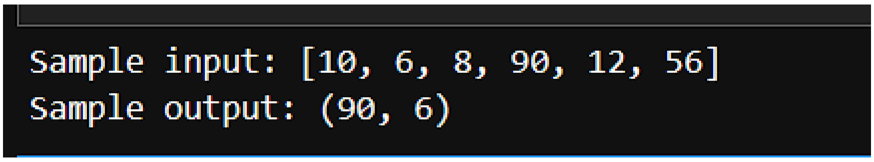
```
#Q5.)
cone_volume = lambda r, h: (1/3) * 3.14159 * r**2 * h
# Example usage:
radius = 3
height = 5
print("Volume of the cone with radius", radius, "and height", height, "is:",
cone_volume(radius, height))
```



```
Volume of the cone with radius 3 and height 5 is: 47.123849999999999
```

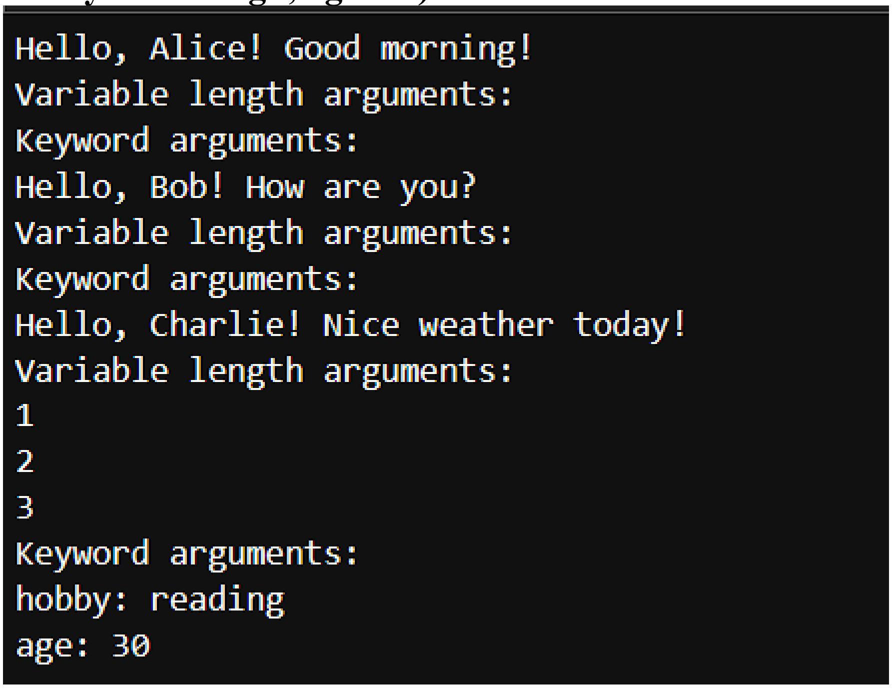
```
#Q6.)
max_min_tuple = lambda lst: (max(lst), min(lst))
# Example usage:
input_list = [10, 6, 8, 90, 12, 56]
output_tuple = max_min_tuple(input_list)
print("Sample input:", input_list)
print("Sample output:", output_tuple)
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



```
Sample input: [10, 6, 8, 90, 12, 56]
Sample output: (90, 6)
```

```
#Q7.)
def function_demo(name, message="Good morning!", *args, **kwargs):
    print(f"Hello, {name}! {message}")
    print("Variable length arguments:")
    for arg in args:
        print(arg)
    print("Keyword arguments:")
    for key, value in kwargs.items():
        print(f"{key}: {value}")
# Example usage:
function_demo("Alice")  # Using default message
function_demo("Bob", "How are you?")  # Specifying message
function_demo("Charlie", "Nice weather today!", 1, 2, 3,
hobby="reading", age=30)
```



```
Hello, Alice! Good morning!
Variable length arguments:
Keyword arguments:
Hello, Bob! How are you?
Variable length arguments:
Keyword arguments:
Hello, Charlie! Nice weather today!
Variable length arguments:
1
2
3
Keyword arguments:
hobby: reading
age: 30
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

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
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
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
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. Sample input 1 0 2 $ 3 1 Sample Output : Error Code: integer division or modulo by zero Error Code: invalid literal for int() with base 10: '$' 3
5. Create multiple suitable exceptions for a file handling program.

```
#Q1.)
with open("name.txt", "w") as file:
    file.write("Alice\n")
    file.write("Bob\n")
    file.write("Charlie\n")
    file.write("David\n")
    file.write("Eva\n")

# a. Count the number of names
with open("name.txt", "r") as file:
    names = file.readlines()
    num_of_names = len(names)
    print("Number of names:", num_of_names)

# b. Count all names starting with a vowel
vowels = "aeiouAEIOU"
num_starting_with_vowel = sum(1 for name in names if name[0] in vowels)
print("Number of names starting with a vowel:",
num_starting_with_vowel)

# c. Find the longest name
longest_name = max(names, key=len).strip()
print("Longest name:", longest_name)
```

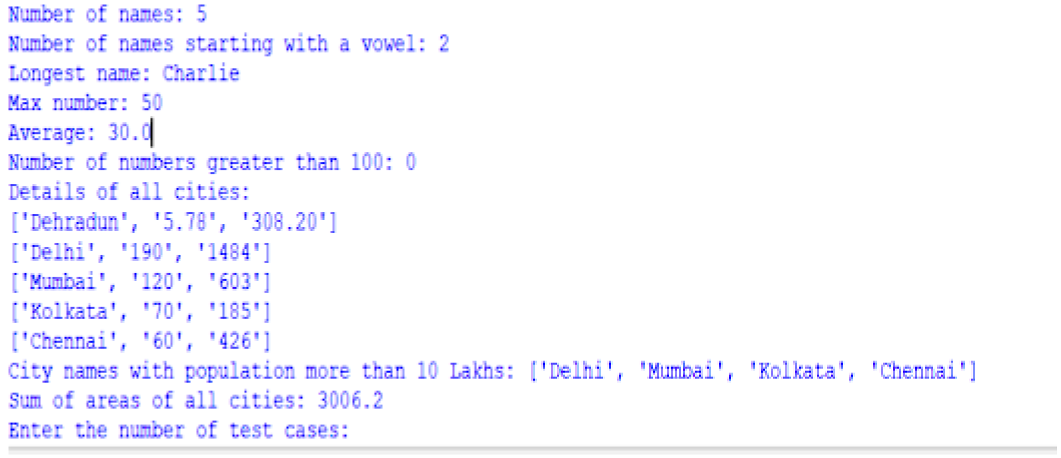
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q2.)
with open("integers.txt", "w") as file:
    file.write("10\n")
    file.write("20\n")
    file.write("30\n")
    file.write("40\n")
    file.write("50\n")
# Read integers from file
with open("integers.txt", "r") as file:
    integers = [int(line.strip()) for line in file]
# a. Find the max number
max_number = max(integers)
print("Max number:", max_number)
# b. Find average of all numbers
average = sum(integers) / len(integers)
print("Average:", average)
# c. Count number of numbers greater than 100
num_greater_than_100 = sum(1 for num in integers if num > 100)
print("Number of numbers greater than 100:", num_greater_than_100)

#Q3.)
with open("city.txt", "w") as file:
    file.write("Dehradun 5.78 308.20\n")
    file.write("Delhi 190 1484\n")
    file.write("Mumbai 120 603\n")
    file.write("Kolkata 70 185\n")
    file.write("Chennai 60 426\n")

# a. Display details of all cities
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)

# b. Display city names with population more than 10 Lakhs
population_more_than_10_lakhs = [city[0] for city in cities if float(city[1])
> 10]
print("City names with population more than 10 Lakhs:",
population_more_than_10_lakhs)

# c. Display sum of areas of all cities
sum_of_areas = sum(float(city[2]) for city in cities)
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
print("Sum of areas of all cities:", sum_of_areas)

#Q4.)
try:
    N = int(input("Enter the number of test cases: "))
    for _ in range(N):
        a, b = map(int, input().split())
        print(a // b)
except ZeroDivisionError:
    print("Error Code: integer division or modulo by zero")
except ValueError as e:
    print(f"Error Code: {e}")

#Q5.)
try:
    with open("non_existent_file.txt", "r") as file:
        content = file.read()
except FileNotFoundError:
    print("File not found.")
except PermissionError:
    print("Permission denied to access the file.")
except Exception as e:
    print(f"An error occurred: {e}")
```

Output:-



```
Number of names: 5
Number of names starting with a vowel: 2
Longest name: Charlie
Max number: 50
Average: 30.0
Number of numbers greater than 100: 0
Details of all cities:
['Dehradun', '5.78', '308.20']
['Delhi', '190', '1484']
['Mumbai', '120', '603']
['Kolkata', '70', '185']
['Chennai', '60', '426']
City names with population more than 10 Lakhs: ['Delhi', 'Mumbai', 'Kolkata', 'Chennai']
Sum of areas of all cities: 3006.2
Enter the number of test cases:
```

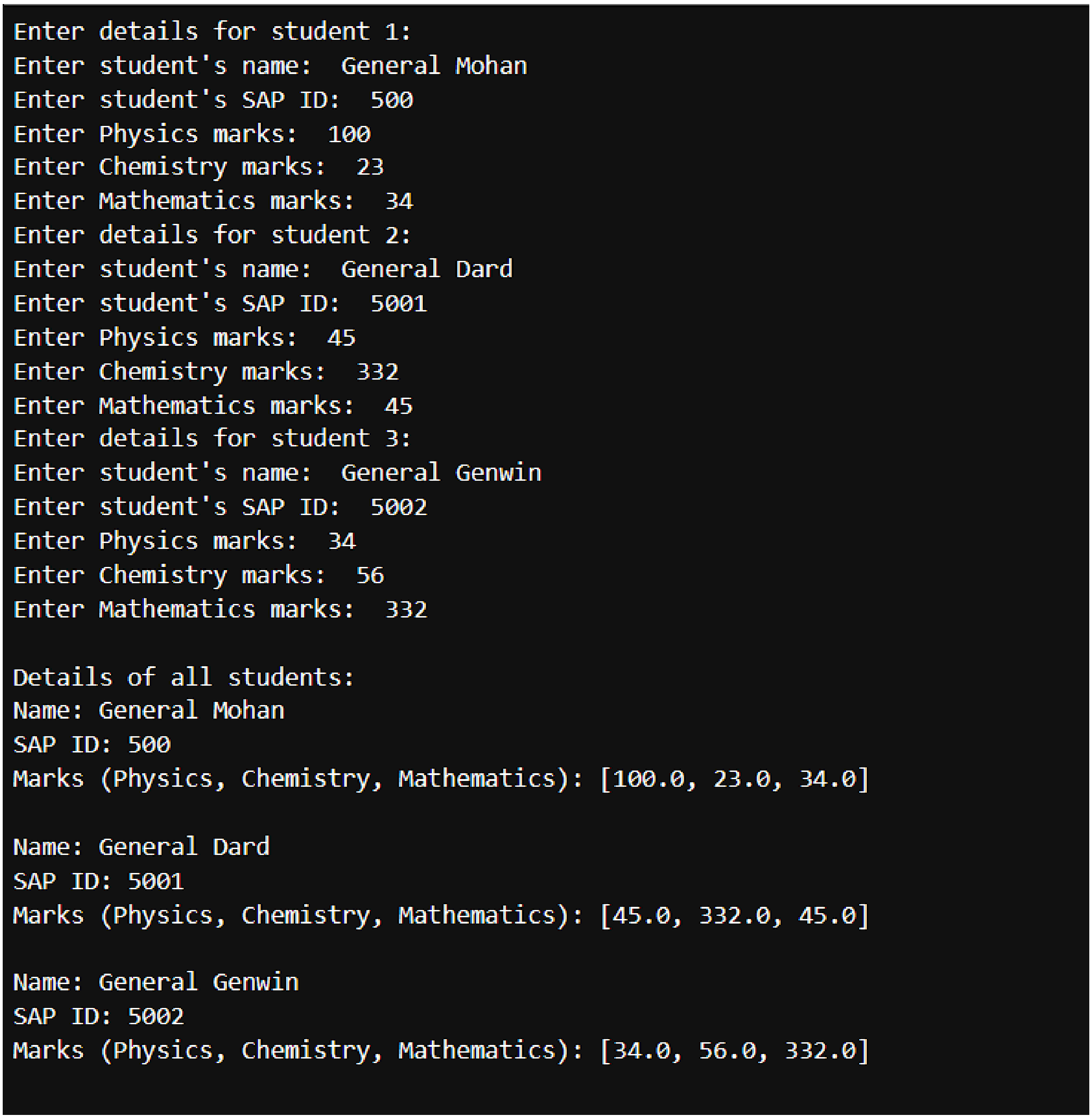
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

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] Write a Function to find average of the class.

```
#Q1.)
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, Mathematics):", self.marks)


# Create 3 student objects by taking inputs from the user
students = []
for i in range(3):
   print(f"Enter details for student {i+1}:")
   name = input("Enter student's name: ")
   sap_id = input("Enter student's SAP ID: ")
   phy_marks = float(input("Enter Physics marks: "))
   chem_marks = float(input("Enter Chemistry marks: "))
   maths_marks = float(input("Enter Mathematics marks: "))
   marks = [phy_marks, chem_marks, maths_marks]
   student = Student(name, sap_id, marks)
   students.append(student)

# Display details of all students
print("\nDetails of all students:")
for student in students:
   student.display_details()
   print()  # Empty line for separation
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



```
Enter details for student 1:
Enter student's name:  General Mohan
Enter student's SAP ID:  500
Enter Physics marks:  100
Enter Chemistry marks:  23
Enter Mathematics marks:  34
Enter details for student 2:
Enter student's name:  General Dard
Enter student's SAP ID:  5001
Enter Physics marks:  45
Enter Chemistry marks:  332
Enter Mathematics marks:  45
Enter details for student 3:
Enter student's name:  General Genwin
Enter student's SAP ID:  5002
Enter Physics marks:  34
Enter Chemistry marks:  56
Enter Mathematics marks:  332

Details of all students:
Name: General Mohan
SAP ID: 500
Marks (Physics, Chemistry, Mathematics): [100.0, 23.0, 34.0]

Name: General Dard
SAP ID: 5001
Marks (Physics, Chemistry, Mathematics): [45.0, 332.0, 45.0]

Name: General Genwin
SAP ID: 5002
Marks (Physics, Chemistry, Mathematics): [34.0, 56.0, 332.0]
```

```
#Q2.)
class Student:
    def __init__(self, student_name, sap_id, phy_marks, chem_marks, maths_marks):
        self.student_name = student_name
        self.sap_id = sap_id
        self.phy_marks = phy_marks
        self.chem_marks = chem_marks
        self.maths_marks = maths_marks

    def display_details(self):
        print("Student Name:", self.student_name)
        print("SAP ID:", self.sap_id)
```

```
        print("Physics Marks:", self.phy_marks)
        print("Chemistry Marks:", self.chem_marks)
        print("Mathematics Marks:", self.maths_marks)

    def marks_percentage(self):
        total_marks = self.phy_marks + self.chem_marks + self.maths_marks
        percentage = (total_marks / 300) * 100
        return percentage

    def result(self):
        if self.phy_marks >= 40 and self.chem_marks >= 40 and
self.maths_marks >= 40:
            return "Pass"
        else:
            return "Fail"


# Function to find average marks of the class
def class_average(students):
    total_percentage = sum(student.marks_percentage() for student in
students)
    average_percentage = total_percentage / len(students)
    return average_percentage


# Create list to store student objects
students = []

# Take input for number of students
num_students = int(input("Enter the number of students: "))

# Input student details and create student objects
for i in range(num_students):
    print(f"Enter details for student {i+1}:")
    student_name = input("Enter student's name: ")
    sap_id = input("Enter student's SAP ID: ")
    phy_marks = float(input("Enter Physics marks: "))
    chem_marks = float(input("Enter Chemistry marks: "))
    maths_marks = float(input("Enter Mathematics marks: "))
    student = Student(student_name, sap_id, phy_marks, chem_marks,
maths_marks)
    students.append(student)
```

```
# Display details, marks percentage, and result for each student
print("\nDetails of all students:")
for student in students:
    print("\nStudent Details:")
    student.display_details()
    print("Marks Percentage:", student.marks_percentage())
    print("Result:", student.result())

# Calculate and display class average
average = class_average(students)
print("\nClass Average Marks Percentage:", average)
```

Experiment 10:-

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

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

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

4. Write a Pandas program to get the powers of an array values element-wise. Note: First array elements raised to powers from second array Sample data: {'X':[78,85,96,80,86], 'Y':[84,94,89,83,86],'Z':[86,97,96,72,83]} Expected Output: X Y Z 0 78 84 86 1 85 94 97 2 96 89 96 3 80 83 72 4 86 86 83

5. Write a Pandas program to get the first 3 rows of a given DataFrame. Sample Python dictionary data and list labels: 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'] Expected Output: First three rows of the data frame: attempts name qualify score a 1 Anastasia yes 12.5 b 3 Dima no 9.0 c 2 Katherine yes 16.5

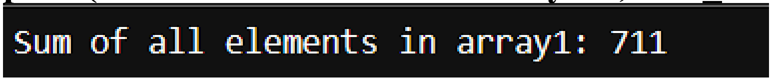
6. Write a Pandas program to find and replace the missing values in a given DataFrame which do not have any valuable information.

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

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

#Q1.)

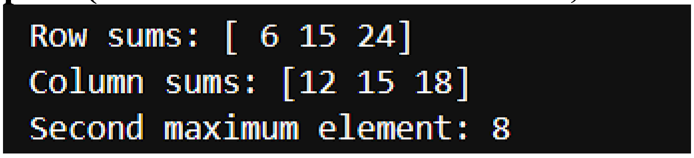
```
import numpy as np
import pandas as pd
array1 = np.array([11, 32, 30, 40, 598])
sum_array1 = np.sum(array1)
print("Sum of all elements in array1:", sum_array1)
```



```
Sum of all elements in array1: 711
```

#Q2.)

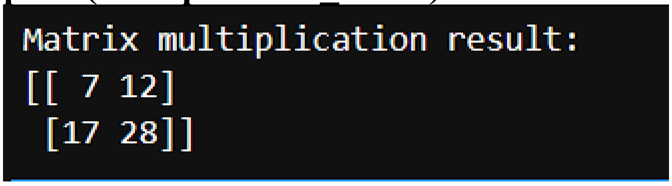
```
import numpy as np
import pandas as pd
array2 = np.array([[1, 2, 3],
                   [4, 5, 6],
                   [7, 8, 9]])
row_sums = np.sum(array2, axis=1)
col_sums = np.sum(array2, axis=0)
second_max_element = np.partition(array2.flatten(), -2)[-2]
print("Row sums:", row_sums)
print("Column sums:", col_sums)
print("Second maximum element:", second_max_element)
```



```
Row sums: [ 6 15 24]
Column sums: [12 15 18]
Second maximum element: 8
```

#Q3.)

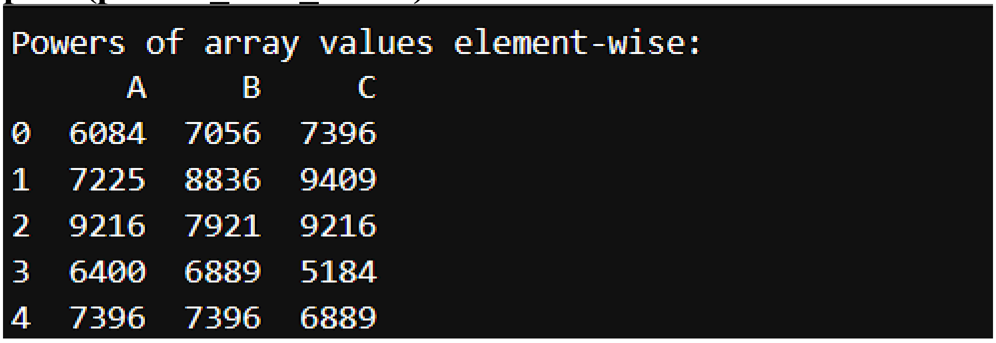
```
import numpy as np
import pandas as panda
matrix_A = np.array([[2, 1],
                     [2, 3]])
matrix_B = np.array([[1, 2],
                     [5, 8]])
multiplication_result = np.dot(matrix_A, matrix_B)
print("Matrix multiplication result:")
print(multiplication_result)
```



```
Matrix multiplication result:
[[ 7 12]
 [17 28]]
```

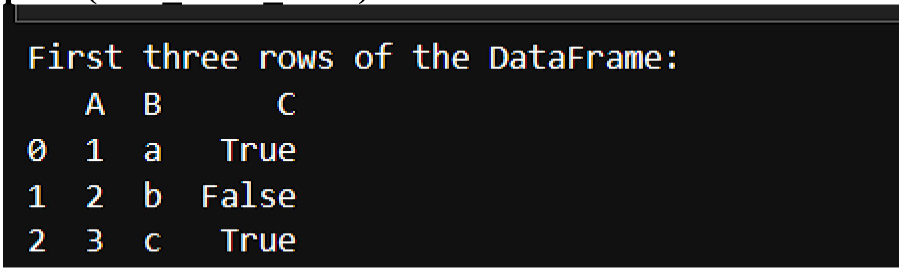
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q4.)
import pandas as pd
import numpy as np
data_dict = {'A': [78, 85, 96, 80, 86],
            'B': [84, 94, 89, 83, 86],
            'C': [86, 97, 96, 72, 83]}

data_frame = pd.DataFrame(data_dict)
# Define a function to calculate powers element-wise
def calculate_power(value, exponent):
    return np.power(value, exponent)
# Apply the function to each element of the DataFrame
powers_data_frame = data_frame.apply(calculate_power, exponent=2)
print("Powers of array values element-wise:")
print(powers_data_frame)
```



```
Powers of array values element-wise:
      A     B     C
0  6084  7056  7396
1  7225  8836  9409
2  9216  7921  9216
3  6400  6889  5184
4  7396  7396  6889
```

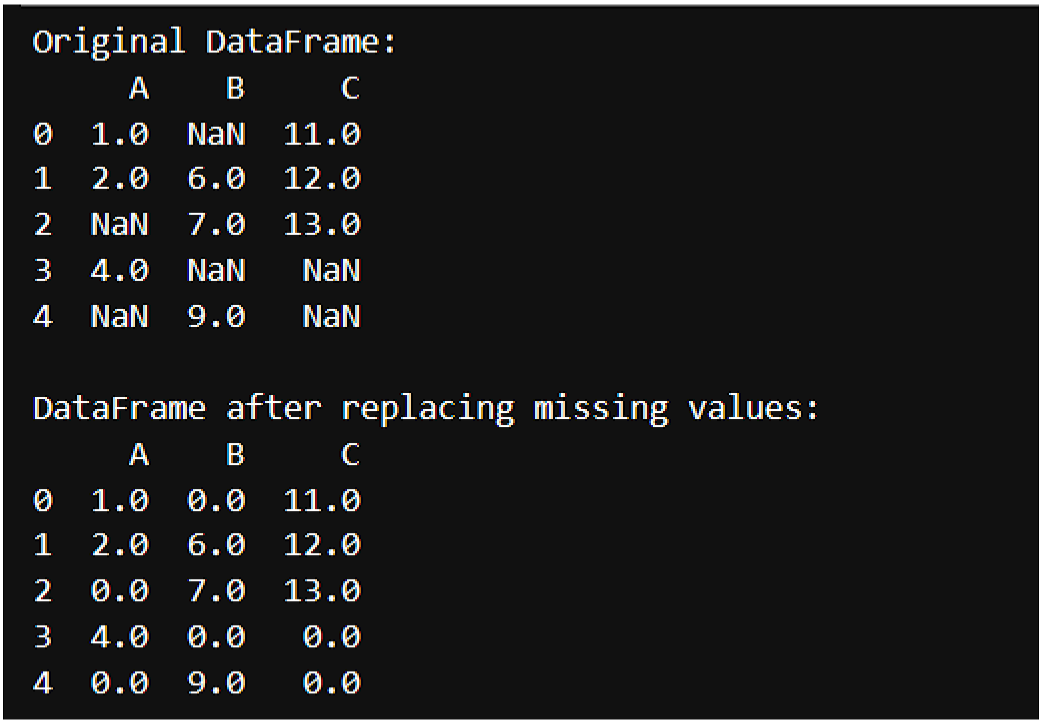
```
#Q5.)
import pandas as pd
# Create a sample DataFrame
data = {'A': [1, 2, 3, 4, 5],
        'B': ['a', 'b', 'c', 'd', 'e'],
        'C': [True, False, True, False, True]}
df = pd.DataFrame(data)
# Get the first 3 rows of the DataFrame
first_three_rows = df.head(3)
# Print the first 3 rows
print("First three rows of the DataFrame:")
print(first_three_rows)
```



```
First three rows of the DataFrame:
   A  B      C
0  1  a   True
1  2  b  False
2  3  c   True
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
#Q6.)
import pandas as pd
import numpy as np
# Creating sample DataFrame with missing values
data = {'A': [1, 2, np.nan, 4, np.nan],
        'B': [np.nan, 6, 7, np.nan, 9],
        'C': [11, 12, 13, np.nan, np.nan]}
df = pd.DataFrame(data)
# Display original DataFrame
print("Original DataFrame:")
print(df)
# Replace missing values with a specified value (e.g., 0)
df.fillna(0, inplace=True)
# Display DataFrame after replacing missing values
print("\nDataFrame after replacing missing values:")
print(df)
```



```
Original DataFrame:
     A    B     C
0  1.0  NaN  11.0
1  2.0  6.0  12.0
2  NaN  7.0  13.0
3  4.0  NaN   NaN
4  NaN  9.0   NaN

DataFrame after replacing missing values:
     A    B     C
0  1.0  0.0  11.0
1  2.0  6.0  12.0
2  0.0  7.0  13.0
3  4.0  0.0   0.0
4  0.0  9.0   0.0
```

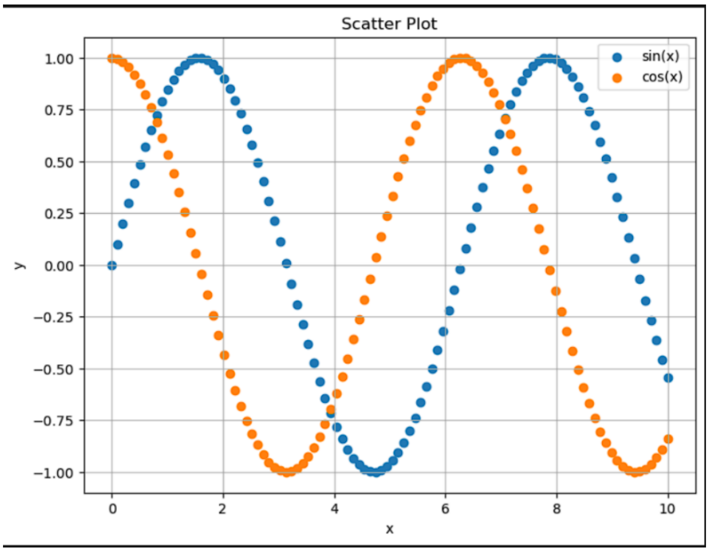
```
#Q7.)
import matplotlib.pyplot as plt
import numpy as np
# Sample data
x = np.linspace(0, 10, 100)
y1 = np.sin(x)
y2 = np.cos(x)
```

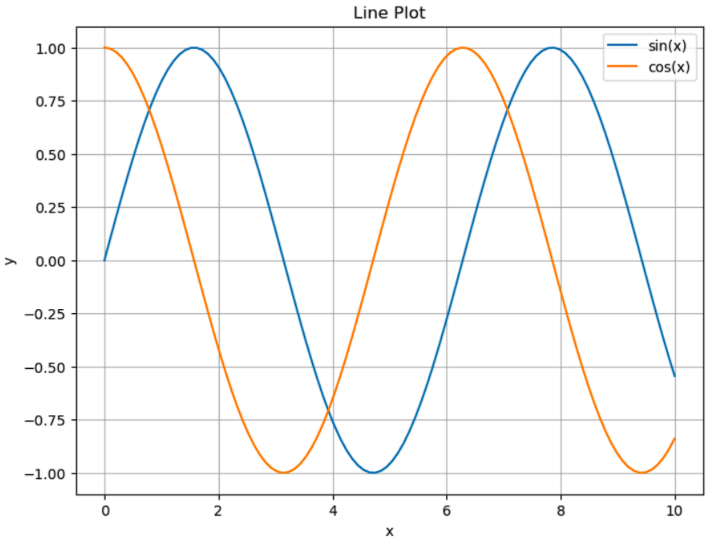
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

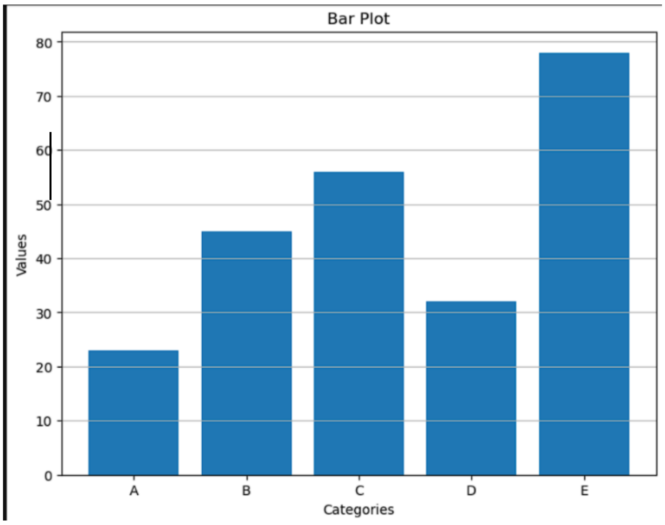
```
# Line plot
plt.figure(figsize=(8, 6))
plt.plot(x, y1, label='sin(x)')
plt.plot(x, y2, label='cos(x)')
plt.title('Line Plot')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid(True)
plt.show()

# Scatter plot
plt.figure(figsize=(8, 6))
plt.scatter(x, y1, label='sin(x)')
plt.scatter(x, y2, label='cos(x)')
plt.title('Scatter Plot')
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.grid(True)
plt.show()

# Bar plot
categories = ['A', 'B', 'C', 'D', 'E']
values = [23, 45, 56, 32, 78]
plt.figure(figsize=(8, 6))
plt.bar(categories, values)
plt.title('Bar Plot')
plt.xlabel('Categories')
plt.ylabel('Values')
plt.grid(axis='y')
plt.show()

# Histogram
data = np.random.normal(loc=0, scale=1, size=1000)
plt.figure(figsize=(8, 6))
plt.hist(data, bins=30, color='skyblue', edgecolor='black')
plt.title('Histogram')
plt.xlabel('Value')
plt.ylabel('Frequency')
plt.grid(axis='y')
plt.show()
```

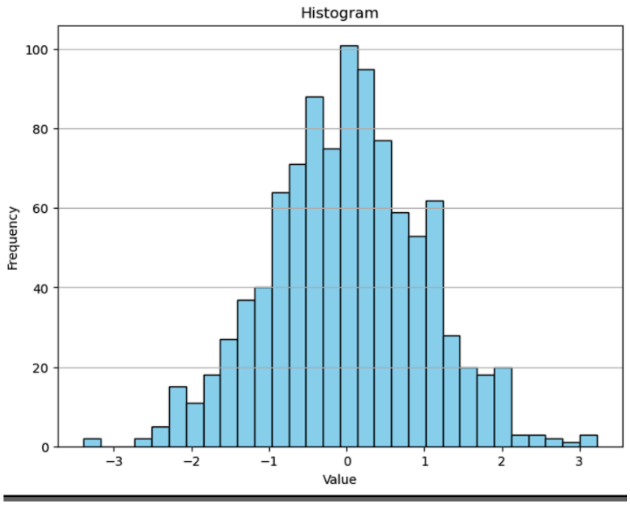
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

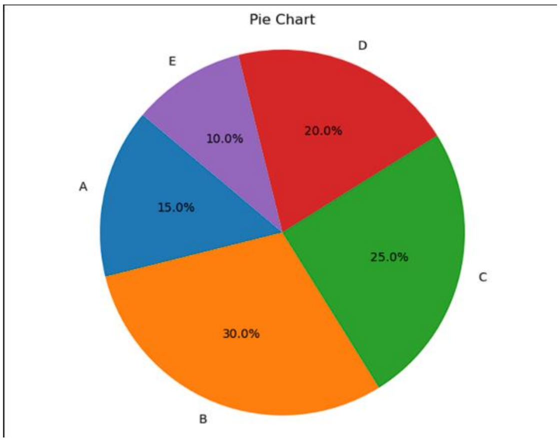
```
# Pie chart
labels = ['A', 'B', 'C', 'D', 'E']
sizes = [15, 30, 25, 20, 10]
plt.figure(figsize=(8, 6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)
plt.title('Pie Chart')
plt.axis('equal')
plt.show()
```











Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Python Project:-
Optical Fibre & Laser Calculations

Breakdown of the key concepts and elements used in this project :-

1. Functions:
   - Modularize different calculations and actions.
   - Optical fiber and laser calculations have their own functions.

2. Global Variables:
   - `calculation_history` list stores the history of calculations.

3. Mathematical Calculations:
   - Trigonometric functions: math.asin()
   - Square roots: math.sqrt()
   - Exponentials: math.exp()
   - Basic arithmetic operations.

4. User Input:
   - Obtained using input() function.
   - Validated to ensure it meets criteria (e.g., positive values).

5. Conditional Statements:
   - Handles user choices and input validation.
   - Determines which calculations to perform based on user-selected options.

6. Looping Constructs:
   - Uses while loop to prompt the user for input and perform calculations until exit.

7. Printing Information:
   - Provides instructions, results, error messages, and calculation history using print().

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

8. Data Structures:
   - Dictionary stores inputs and results for each calculation in calculation_history list.

9. Error Handling:
   - Validates user input (e.g., positive values).
   - Includes error messages to guide the user.

10. Mathematical Constants:
    - Utilizes constants like Planck's constant (h) and speed of light (c) in calculations.

11. Modularity and Reusability:
    - Functions are modularized for code reusability and maintainability.

12. Documentation:
    - Comments document the purpose and functionality of each part of the code.

Source Code:-

```
import math

calculation_history=[] #global list variable to store calculation history

def save_to_history(calculation_name, inputs, result):
    calculation_history.append({
        "name": calculation_name,
        "inputs": inputs,
        "result": result
    })

def calculate_numerical_aperture(core_radius, refractive_index_core,
refractive_index_cladding):
    # Calculate the critical angle
    critical_angle = math.asin(refractive_index_cladding / refractive_index_core)
    # Calculate the numerical aperture
    numerical_aperture = math.sqrt(refractive_index_core**2 -
refractive_index_cladding**2)
    return numerical_aperture

def calculate_photon_energy(wavelength):
    # Calculate the energy of a photon using Planck's equation
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
    h = 6.62607015e-34  # Planck's constant in J·s
    c = 3e8  # Speed of light in m/s
    energy = h * c / wavelength
    return energy

def calculate_pulse_dispersion(material_dispersion, waveguide_dispersion,
pulse_width):
    # Calculate the total dispersion
    total_dispersion = material_dispersion + waveguide_dispersion
    # Calculate the pulse spread
    pulse_spread = total_dispersion * pulse_width
    return pulse_spread

def calculate_acceptance_angle(refractive_index_core,
refractive_index_cladding):
    # Calculate the acceptance angle in degrees
    acceptance_angle = math.degrees(math.asin(refractive_index_cladding /
refractive_index_core))
    return acceptance_angle

def calculate_population_ratio(energy_state_1, energy_state_2, temperature):
    # Calculate the ratio of population in two states using the Boltzmann
distribution
    k = 8.6173e-5  # Boltzmann constant in eV/K
    population_ratio = math.exp(-(energy_state_2 - energy_state_1) / (k *
temperature))
    return population_ratio

def optical_fiber_calculation():
    print("Select an option for Optical Fiber Calculation:")
    print("1. Numerical Aperture (NA)")
    print("2. Pulse Spread Due to Dispersion")
    print("3. Acceptance Angle")
    option = input("Enter your choice: ")
    if option == "1":
        print("Numerical Aperture (NA) Calculation:")
        print("The numerical aperture (NA) of an optical fiber is a measure of its
ability to gather light.")
        print("It is calculated using the formula:")
        print("NA = √(core^2 - clad^2)")
        core_radius = float(input("Enter the core radius of the optical fiber (in
micrometers): "))
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

```
        refractive_index_core = float(input("Enter the refractive index of the core: "))
        refractive_index_cladding = float(input("Enter the refractive index of the cladding: "))
        if refractive_index_core > refractive_index_cladding:
            numerical_aperture = calculate_numerical_aperture(core_radius, refractive_index_core, refractive_index_cladding)
            print("Numerical Aperture:", numerical_aperture)
            save_to_history("Numerical Aperture", {"Core Radius": core_radius, "Refractive Index Core": refractive_index_core, "Refractive Index Cladding": refractive_index_cladding}, numerical_aperture)
        else:
            print("Refractive Index of Core Must be Greater Than Cladding")
    elif option == "2":
        print("Pulse Spread Due to Dispersion Calculation:")
        print("Pulse spread due to dispersion refers to the broadening of an optical pulse as it propagates through a fiber.")
        print("It is calculated using the formula:")
        print("Pulse Spread = (Material Dispersion + Waveguide Dispersion) * Pulse Width")
        # Material dispersion validation
        material_dispersion = 0
        while material_dispersion <= 0:
            material_dispersion = float(input("Enter the material dispersion (in ps/nm-km): "))
            if material_dispersion <= 0:
                print("Material dispersion must be a positive value.")
        # Waveguide dispersion validation
        waveguide_dispersion = 0
        while waveguide_dispersion <= 0:
            waveguide_dispersion = float(input("Enter the waveguide dispersion (in ps/nm-km): "))
            if waveguide_dispersion <= 0:
                print("Waveguide dispersion must be a positive value.")
        # Pulse width validation
        pulse_width = 0
        while pulse_width <= 0:
            pulse_width = float(input("Enter the pulse width (in picoseconds): "))
            if pulse_width <= 0:
                print("Pulse width must be a positive value.")
        pulse_spread = calculate_pulse_dispersion(material_dispersion, waveguide_dispersion, pulse_width)
        print("Pulse Spread (in micrometers):", pulse_spread)
```

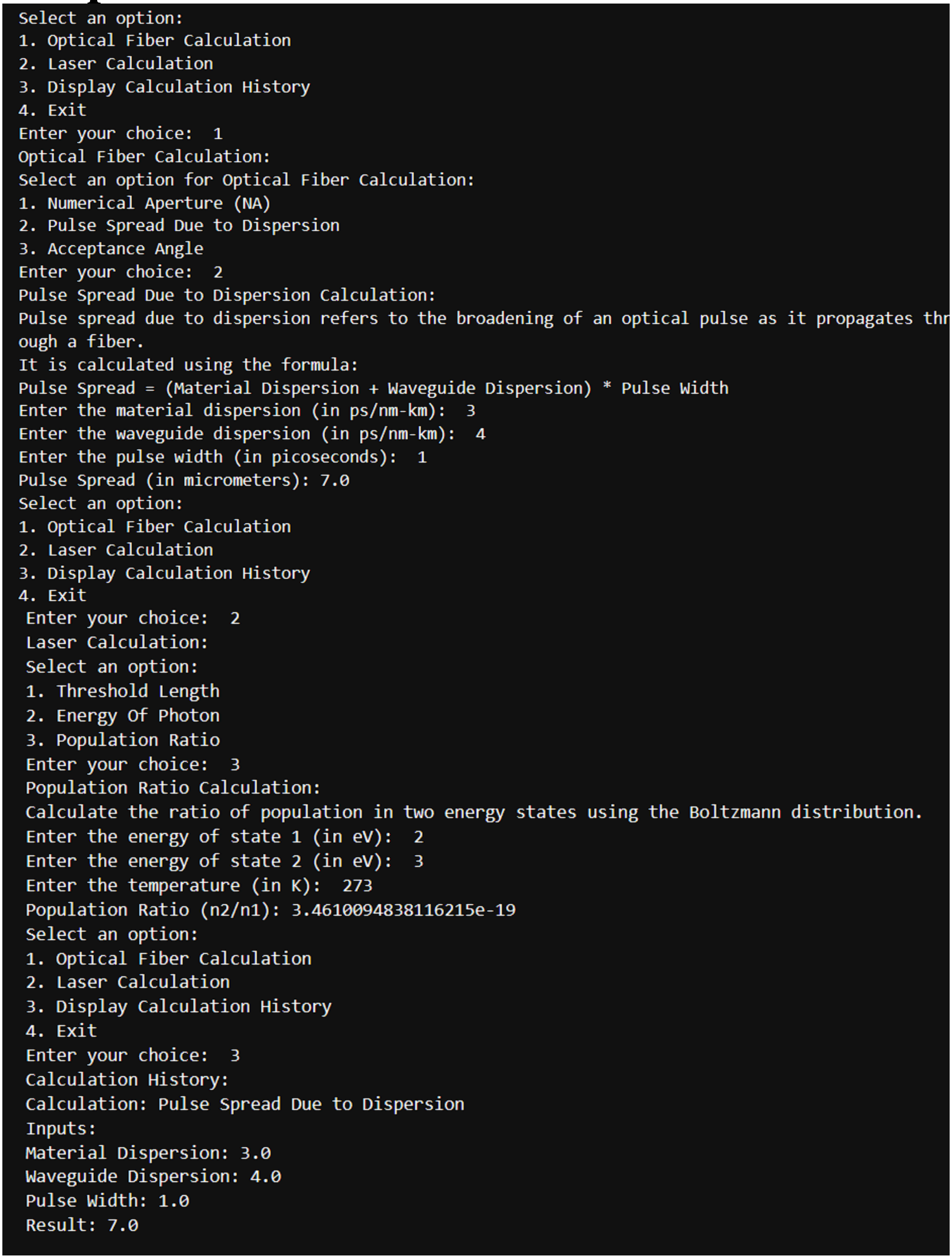
```
        save_to_history("Pulse Spread Due to Dispersion", {"Material
Dispersion": material_dispersion, "Waveguide Dispersion":
waveguide_dispersion, "Pulse Width": pulse_width}, pulse_spread)
    elif option == "3":
        print("Acceptance Angle Calculation:")
        print("The acceptance angle of an optical fiber is the maximum angle at
which light can enter the fiber and still be guided along the core.")
        print("It is calculated using the formula:")
        print("Acceptance Angle (θ) = arcsin(n_clad / n_core)")
        # Refractive index validation
        refractive_index_core = float(input("Enter the refractive index of the core:
"))
        refractive_index_cladding = float(input("Enter the refractive index of the
cladding: "))
        if refractive_index_core > refractive_index_cladding:
            acceptance_angle = calculate_acceptance_angle(refractive_index_core,
refractive_index_cladding)
            print("Acceptance Angle (in degrees):", acceptance_angle)
            save_to_history("Acceptance Angle", {"Refractive Index Core":
refractive_index_core, "Refractive Index Cladding":
refractive_index_cladding}, acceptance_angle)
        else:
            print("Error: Refractive Index of Core must be greater than Refractive
Index of Cladding")
    else:
        print("Invalid option. Please select either '1', '2' or '3'.")

def laser_calculation():
    print("Select an option:")
    print("1. Threshold Length")
    print("2. Energy Of Photon")
    print("3. Population Ratio")
    laser_option = input("Enter your choice: ")
    if laser_option == "1":
        print("Threshold Length Calculation:")
        print("The threshold length of a laser is the length of the laser cavity
required to reach the threshold gain.")
        print("It is calculated using the formula:")
        print("Threshold Length (Lth) = Threshold Gain (Gth) / Material Gain
Coefficient (α)")
        # Material gain coefficient validation
        material_gain = 0
        while material_gain <= 0:
```

```
            material_gain = float(input("Enter the material gain coefficient (in cm^-
1): "))
            if material_gain <= 0:
                print("Material gain coefficient must be a positive value.")
        # Threshold gain validation
        threshold_gain = float(input("Enter the threshold gain of the laser (in
dB/cm): "))
        threshold_length = threshold_gain / material_gain
        print("Threshold Length:", threshold_length)
        save_to_history("Threshold Length", {"Material Gain Coefficient":
material_gain, "Threshold Gain": threshold_gain}, threshold_length)
    elif laser_option == '2':
        print("Photon Energy Calculation:")
        print("Calculate the energy of a photon based on its wavelength using
Planck's equation.")
        # Wavelength validation
        wavelength = 0
        while wavelength <= 0:
            wavelength = float(input("Enter the wavelength of the photon (in
meters): "))
            if wavelength <= 0:
                print("Wavelength must be a positive value.")
        energy = calculate_photon_energy(wavelength)
        print("Photon Energy (in Joules):", energy)
        save_to_history("Photon Energy", {"Wavelength": wavelength}, energy)
    elif laser_option == "3":
        print("Population Ratio Calculation:")
        print("Calculate the ratio of population in two energy states using the
Boltzmann distribution.")
        energy_state_1 = float(input("Enter the energy of state 1 (in eV): "))
        energy_state_2 = float(input("Enter the energy of state 2 (in eV): "))
        temperature = float(input("Enter the temperature (in K): "))
        population_ratio = calculate_population_ratio(energy_state_1,
energy_state_2, temperature)
        print("Population Ratio (n2/n1):", population_ratio)
        save_to_history("Population Ratio", {"Energy State 1": energy_state_1,
"Energy State 2": energy_state_2, "Temperature": temperature},
population_ratio)
    else:
        print("Invalid option. Please select a valid option (1, 2, or 3).")

def display_history():
    print("Calculation History:")
```

```
    if not calculation_history:
        print("No calculations performed yet.")
    else:
        for entry in calculation_history:
            print(f"Calculation: {entry['name']}")
            print("Inputs:")
            for key, value in entry['inputs'].items():
                print(f"{key}: {value}")
            print(f"Result: {entry['result']}")
            print("\n")

def main():
    while True:
        print("Select an option:")
        print("1. Optical Fiber Calculation")
        print("2. Laser Calculation")
        print("3. Display Calculation History")
        print("4. Exit")
        option = input("Enter your choice: ")
        if option == "1":
            print("Optical Fiber Calculation:")
            optical_fiber_calculation()
        elif option == "2":
            print("Laser Calculation:")
            laser_calculation()
        elif option == "3":
            display_history()
        elif option == "4":
            print("Bye Bye 🤫 🤫 🧏 🧏 🗣 🗣 ")
            break
        else:
            print("Invalid option. Please select either '1', '2', '3', or '4'.")

main()
```

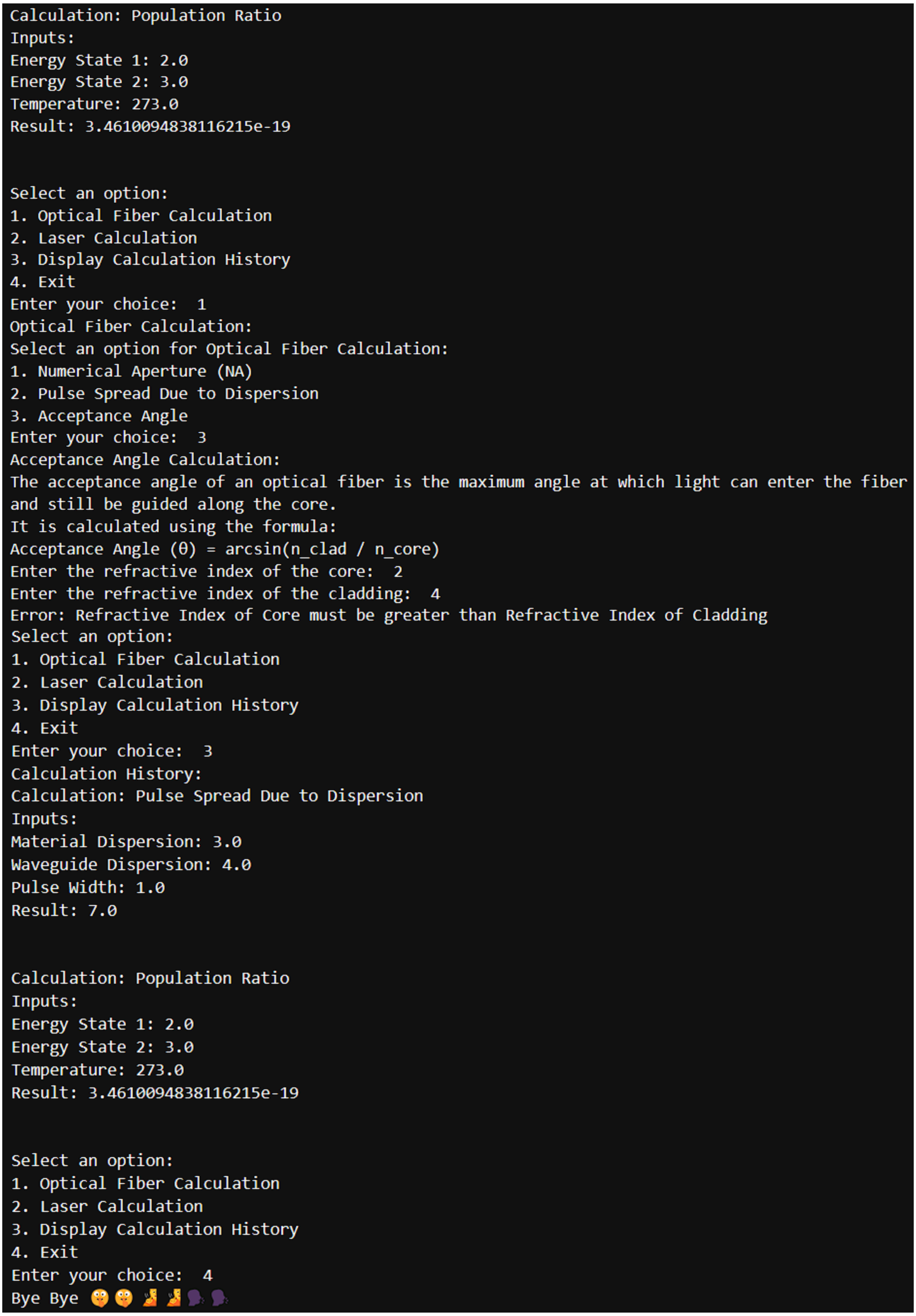
Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086

Output:-



```
Select an option:
1. Optical Fiber Calculation
2. Laser Calculation
3. Display Calculation History
4. Exit
Enter your choice:  1
Optical Fiber Calculation:
Select an option for Optical Fiber Calculation:
1. Numerical Aperture (NA)
2. Pulse Spread Due to Dispersion
3. Acceptance Angle
Enter your choice:  2
Pulse Spread Due to Dispersion Calculation:
Pulse spread due to dispersion refers to the broadening of an optical pulse as it propagates thr
ough a fiber.
It is calculated using the formula:
Pulse Spread = (Material Dispersion + Waveguide Dispersion) * Pulse Width
Enter the material dispersion (in ps/nm-km):  3
Enter the waveguide dispersion (in ps/nm-km):  4
Enter the pulse width (in picoseconds):  1
Pulse Spread (in micrometers): 7.0
Select an option:
1. Optical Fiber Calculation
2. Laser Calculation
3. Display Calculation History
4. Exit
 Enter your choice:  2
 Laser Calculation:
 Select an option:
 1. Threshold Length
 2. Energy Of Photon
 3. Population Ratio
 Enter your choice:  3
 Population Ratio Calculation:
 Calculate the ratio of population in two energy states using the Boltzmann distribution.
 Enter the energy of state 1 (in eV):  2
 Enter the energy of state 2 (in eV):  3
 Enter the temperature (in K):  273
 Population Ratio (n2/n1): 3.4610094838116215e-19
 Select an option:
 1. Optical Fiber Calculation
 2. Laser Calculation
 3. Display Calculation History
 4. Exit
 Enter your choice:  3
 Calculation History:
 Calculation: Pulse Spread Due to Dispersion
 Inputs:
 Material Dispersion: 3.0
 Waveguide Dispersion: 4.0
 Pulse Width: 1.0
 Result: 7.0
```

Name :- Anadi Sharma
SAP ID :- 500126798
Roll No. - R2142232086



```
Calculation: Population Ratio
Inputs:
Energy State 1: 2.0
Energy State 2: 3.0
Temperature: 273.0
Result: 3.4610094838116215e-19


Select an option:
1. Optical Fiber Calculation
2. Laser Calculation
3. Display Calculation History
4. Exit
Enter your choice:  1
Optical Fiber Calculation:
Select an option for Optical Fiber Calculation:
1. Numerical Aperture (NA)
2. Pulse Spread Due to Dispersion
3. Acceptance Angle
Enter your choice:  3
Acceptance Angle Calculation:
The acceptance angle of an optical fiber is the maximum angle at which light can enter the fiber
and still be guided along the core.
It is calculated using the formula:
Acceptance Angle (θ) = arcsin(n_clad / n_core)
Enter the refractive index of the core:  2
Enter the refractive index of the cladding:  4
Error: Refractive Index of Core must be greater than Refractive Index of Cladding
Select an option:
1. Optical Fiber Calculation
2. Laser Calculation
3. Display Calculation History
4. Exit
Enter your choice:  3
Calculation History:
Calculation: Pulse Spread Due to Dispersion
Inputs:
Material Dispersion: 3.0
Waveguide Dispersion: 4.0
Pulse Width: 1.0
Result: 7.0


Calculation: Population Ratio
Inputs:
Energy State 1: 2.0
Energy State 2: 3.0
Temperature: 273.0
Result: 3.4610094838116215e-19


Select an option:
1. Optical Fiber Calculation
2. Laser Calculation
3. Display Calculation History
4. Exit
Enter your choice:  4
Bye Bye 🤫🤫🧖‍♀️🧖‍♀️🗣🗣
```