Submitted to:

Mrs. Sugandha Sharma

Submitted by:

Rohan Singh

Batch No.43

Roll-R2142231439

SAP ID:500124818

EXPERIMENT – 1

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’

Program:

```
print("hello word")
 x="Hello Everyone
!!!" print(x)

x="Hello"
b="World"
print(x)
print(b)
a="Hello"
b="
World"
print(a)
print(b)

date='12/05/1999' s="chintu date
of birth is " + date print(s)
```

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

Print out the value of x

Program:

```
x="Hello"
print(x)
```

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

Program:

```
x=10 y=90
print(x)
print(y)
```

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.

Program:

```
first_name='Rohan' last_name='
Singh' joined_name= first_name +
last_name print(joined_name)
```

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.

Program:

```
first_name=" Rohan " last_name=Singh"
nickname="Jonu"
```

5 7. Declare and assign values to suitable variables and print in the following way :

NAME : ROHAN SINGH

SAP ID : 5000124818

DATE OF BIRTH : 12 Dec 2004

ADDRESS : UPES

Bidholi Campus

Pincode : 248007

Programme : Python

Program:

```
Name="ROHAN SINGH" sap_id='500124818' date_of_birth="12 Dec
2004\n" address="UPES\n                    Bidholi  campus\n
Pincode:248007"          program="Python"          semester='2'
print("NAME:"+Name)  print("SAP  ID:"+sap_id)  print("DATE  OF
BIRTH:"+date_of_birth)               print("ADDRESS:"+address)
print("PROGRAM:"+program) print("SEMESTER:"+semester)
```

EXPERIMENT – 2:

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.

Program:

```
x=int(input("Enter the first number:")) y=int(input("Enter
the second number:")) addition=x+y subtraction=x-y
multiplication=x*y division=x/y print(addition)
print(subtraction) print(multiplication)
print(division)
```

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

Program:

```
radius=int(input("Enter the radius of circle:"))
area=3.14*radius*radius
print(area)
```

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

Test data : x = 4 , y = 3

Expected output: 49

Program:

```
x=int(input("Enter the first number:")) y=int(input("Enter the
second number:")) solution=(x+y)*(x+y)
print(solution)
```

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

Program:

```
import math a=float(input("Enter the length of
side a:")) b=float(input("Enter the length of
side b:")) c=math.sqrt(a*a+b*b)
print("The length of the hypotenuse is:",c)
```

5. Write a program to find simple interest.

Program:

```
rate=float(input("Enter the rate:"))
principal=float(input("Enter the principal:"))
time=float(input("Enter the time period:"))
interest=(principal*rate*time)/100
print(interest)
```

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

Program:

```
import math
a=int(input("Enter the first side:")) b=int(input("Enter the
second side:")) c=int(input("Enter the third side:"))
s=(a+b+c)/2 area=math.sqrt(s*(s-a)(s-b)(s-c)) print(area)
```

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

Program:

```
seconds=int(input("Enter the number of seconds:"))
hours=seconds//3600 minutes=(seconds%3600)//60
remain_seconds=(seconds%3600)%60 print("The time is", hours,
"hours", minutes, "minutes and", remain_seconds,
"seconds")
```

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

Program:

```
a=int(input("Enter the first number:"))
b=int(input("Enter the second number:"))
a=a+b b=a-b a=a-b
print("After swapping:", a,b)
```

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

```
n=int(input("Enter the number:")) sum=(n*(n+1))//2
print("The sum of first", n, "natural numbers is:", sum)
```

10. Write a program to print truth table for bitwise operators(& , | and ^ operators)

Program:

```
print("Truth Table for Bitwise AND (&), OR (|), and XOR (^) operators:")
print("x\t y\t x & y\t x | y\t x ^ y")
 for x in range(2):
for y in range(2):
        print(f"{x}\t {y}\t {x & y}\t {x | y}\t {x ^ y}")
```

11. Write a program to find left shift and right shift values of a given number.

Program:

```
left_shift_result = num << shift_value right_shift_result
= num >> shift_value
 print(f"Left shift result: {left_shift_result}")
print(f"Right shift result: {right_shift_result}")
```

12. Using membership operator find whether a given number is in sequence

(10,20,56,78,89)

Program:

```
given_number = int(input("Enter a number: "))
sequence = [10, 20, 56, 78, 89]
 if given_number in
sequence:
    print(f"{given_number} is in the sequence.") else:
    print(f"{given_number} is not in the sequence.")
```

13. Using membership operator find whether a given character is in a string.

Program:

```
given_char = input("Enter a character: ")
given_string = "Hello, World!"
 if given_char in
given_string:
    print(f"{given_char} is present in the string.")
else:
    print(f"{given_char} is not present in the string.")
```

EXPERIMENT – 3

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

Program:

```
number = int(input("Enter a number: ")) if
(number % 3 == 0) and (number % 5 == 0):
    print(f"{number} is divisible by both 3 and 5.")
else:
    print(f"{number} is not divisible by both 3 and 5.")
```

Output:

```
Enter a number: 15
15 is divisible by both 3 and 5.
```

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

Program:

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

Output:

```
Enter a number: 90
90 is divisible by both 3 and 5.
```

3. Find the greatest among two numbers. If numbers are equal than print "numbers are equal".

Program:

```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
if num1 > num2:
    print(f"The greatest number is: {num1}")
elif num2 > num1:
    print(f"The greatest number is: {num2}")
else:     print("Numbers are equal.")
```

Output:

```
Enter the first number: 23
Enter the second number: 56
The greatest number is: 56
```

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

Program:

```
num1 = int(input("Enter the first number: ")) num2 =
int(input("Enter the second number: ")) num3 =
int(input("Enter the third number: ")) greatest_number =
max(num1, num2, num3)
print(f"The greatest number is: {greatest_number}")
```

Output:

```
Enter the first number: 23
Enter the second number: 56
The greatest number is: 56
```

5. Check whether the quadratic equation has real roots or imaginary roots. Display the roots.

Program:

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

Program:

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

Output:

```
Enter a year: 2024
2024 is a leap year.
```

7. Write a program which takes any date as input and display next date of the calendar e.g.

   I/P: day=20 month=9 year=2005

   O/P: day=21 month=9 year 2005

   Program:

```
days_in_month = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
 if (year % 4 == 0 and year % 100 != 0) or (year % 400 ==
0):
    days_in_month[2] = 29
if 1 <= day <=
days_in_month[month]:
    day += 1     if day >
days_in_month[month]:
        day = 1
month += 1         if
month > 12:
month = 1
year += 1
 print(f"The next date is: day={day} month={month}
year={year}")
```

Output:

```
Enter the day: 20
Enter the month: 9
Enter the year: 2005
Next date:
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: Rohan Singh

Roll Number: R2142231439 SAPID: 500124818

Sem: 2 Course: B.Tech. Python

Subject name: Marks

PDS: 70 Python:80

Chemistry: 90

English: 60 Physics:

50

Percentage: 70%

CGPA:7.0 Grade:

Program:

```
name = input("Enter student name: ")
roll_number = input("Enter roll number:
") sapid = input("Enter SAPID: ")
semester = input("Enter semester: ")
course = input("Enter course: ")
 pds = float(input("Enter marks for PDS: ")) python =
float(input("Enter marks for Python: ")) chemistry =
float(input("Enter marks for Chemistry: ")) english =
float(input("Enter marks for English: ")) physics =
float(input("Enter marks for Physics: "))
 total_marks = pds + python + chemistry + english +
physics percentage = (total_marks / 500) * 100
 cgpa = percentage /
10
 if 0 <= cgpa <=
3.4:
    grade = 'F' elif
3.5 <= cgpa <= 5.0:
    grade = 'C+' elif
5.1 <= cgpa <= 6.0:
    grade = 'B' elif
6.1 <= cgpa <= 7.0:
    grade = 'B+' elif
7.1 <= cgpa <= 8.0:
    grade = 'A' elif
8.1 <= cgpa <= 9.0:
    grade = 'A+' elif
9.1 <= cgpa <= 10.0:
    grade = 'O (Outstanding)'
else:
    grade = 'Invalid CGPA'


print("\nGradesheet") print(f"Name: {name}") print(f"Roll Number:
{roll_number}\tSAPID: {sapid}") print(f"Sem: {semester}\tCourse:
{course}") print("Marks") print(f"PDS: {pds} Python: {python}")
print(f"Chemistry: {chemistry} English: {english} Physics:
{physics}") print(f"Percentage: {percentage}%") print(f"CGPA:
{cgpa:.1f} Grade: {grade}")
```

Output:

```
Enter student's name: Rohan Singh
Enter student's roll number: 2142231439
Enter student's SAPID: 500124818
Enter student's semester: 2
Enter student's course: B Tech. Python
Enter marks of five subjects:
Enter marks of subject 1: 78
Enter marks of subject 2: 67
Enter marks of subject 3: 56
Enter marks of subject 4: 98
Enter marks of subject 5:
```

Experiment 4: Loops

1. Find a factorial of given number.

Program:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)
number = int(input("Enter a number: "))
print("Factorial of", number, "is", factorial(number))
```

Output:

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

2. Find whether the given number is Armstrong number.

Program:

```
def is_armstrong(num):
    order = len(str(num))
    sum = 0
    temp = num
    while temp > 0:
        digit = temp % 10
        sum += digit ** order
        temp //= 10
    return num == sum

number = int(input("Enter a number: "))
if is_armstrong(number):
    print(number, "is an Armstrong number")
else:
    print(number, "is not an Armstrong number")
```

Output:


Enter a number: 153
153 is an Armstrong number
PS C:\Users\Varun>

3. Print Fibonacci series up to given term.

Program:

```
def fibonacci(n):
    a, b = 0, 1
    count = 0
    while count < n:
        print(a, end=" ")
        nth = a + b
        a = b
        b = nth
        count += 1
terms = int(input("Enter the number of terms: "))
print("Fibonacci series:")
fibonacci(terms)
```

Output:

```
Enter the number of terms: 8
Fibonacci series:
0 1 1 2 3 5 8 13
```

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

Program:

```
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True


number = int(input("Enter a number: "))
if is_prime(number):
    print(number, "is a prime number")
else:
    print(number, "is not a prime number")
```

Output:

```
Enter a number: 46
46 is not a prime number
```

5. Check whether given number is palindrome or not.

Program:
```
def is_palindrome(num):
    temp = num
    reverse = 0
    while temp > 0:
        digit = temp % 10
        reverse = reverse * 10 + digit
        temp //= 10
    return num == reverse


number = int(input("Enter a number: "))
if is_palindrome(number):
    print(number, "is a palindrome")
else:
    print(number, "is not a palindrome")
```

Output:

6. Write a program to print sum of digits.

Program:
```
def sum_of_digits(num):
    total = 0
    while num > 0:
        digit = num % 10
        total += digit
        num //= 10
    return total


number = int(input("Enter a number: "))
print("Sum of digits in", number, "is", sum_of_digits(number))
```

Output:

```
Enter a number: 12345
Sum of digits in 12345 is 15
```

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

Program:

```
def divisible_by_5_or_7():
    count = 0
    for i in range(1, 101):
        if i % 5 == 0 or i % 7 == 0:
            count += 1
            print(i, end=" ")
    print("\nTotal:", count)


print("Numbers divisible by 5 or 7 between 1 to 100:")
divisible_by_5_or_7()
```

Output:

```
Numbers divisible by 5 or 7 between 1 to 100:
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
Total: 32
```

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

Program:

```
def convert_to_uppercase(string):
    return string.upper()
text = input("Enter a string: ")
print("Uppercase:", convert_to_uppercase(text))
```

Output:

```
Enter a string: a B c d j
Uppercase: A B C D J
```

9. Print all prime numbers between 1 and 100.

Program:

```
def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True
def print_primes():
    print("Prime numbers between 1 and 100:")
    for num in range(1, 101):
        if is_prime(num):
            print(num, end=" ")
print_primes()
```

Output:

```
Prime numbers between 1 and 100:
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……..

Program:

```
def print_table(num):
    for i in range(1, 11):
        print(num, "*", i, "=", num*i)


number = int(input("Enter a number to print its table: "))
print("Table for", number, ":")
print_table(number)
```

Output:

```
Enter a number to print its table: 6
Table for 6 :
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
6 * 10 = 60
```

Experiment 5: String and Sets

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

   Program:

```
def count_capital_letters(string):
    count = 0
    for char in string:
        if char.isupper():
            count += 1
    return count


text = input("Enter a string: ")
print("Number of capital letters:", count_capital_letters(text))
```

   Output:

```
Enter a string: A b c U L k T
Number of capital letters: 4
```

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

   Program:

```
def count_vowels(string):
    vowels = "aeiouAEIOU"
    count = 0
    for  char  in  string:
        if char in vowels:
            count += 1
    return count


text = input("Enter a string: ")
print("Number of vowels:", count_vowels(text))
```

   Output:

```
Enter a string: hey my name is varun what's yours
Number of vowels: 9
```

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

   Program:

```
sentence = input("Enter a sentence: ")
```

```
words = sentence.split()
print("Words in separate lines:")
for word in words:
    print(word)
```

Output:

```
Enter a sentence: Today i am writing the 5th experiment of python
Words in separate lines:
Today
i
am
writing
the
5th
experiment
of
python
```

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

Program:

```
def count_substring(string, sub):
    count = 0
    start = 0
    while True:
        start = string.find(sub, start)
        if start == -1:
            break
        count += 1
        start += 1
    return count
```

```
main_string = input("Enter a string: ")
sub_string = input("Enter a substring: ")
print("Number of times substring occurs:",
      count_substring(main_string, sub_string))
```

Output:

```
Enter a string: Hello there
Enter a substring: Hi
Number of times substring occurs: 0
```

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

2

A

3

B

2C

1G

Program:

```
def count_alphabets(string):
    counts = {}
    for char in string:
        if char.isalpha():
            char = char.upper()
            counts[char] = counts.get(char, 0) + 1
    return counts


text = input("Enter a string: ")
result = count_alphabets(text)
print("Occurrences of each alphabet:")
for char, count in result.items():
    print(char + ":", count)
```

Output:

```
Enter a string: Hello World
Occurrences of each alphabet:
H: 1
E: 1
L: 3
O: 2
W: 1
R: 1
D: 1
```

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

Program:

```
sentence = input("Enter a sentence: ")
words = sentence.split()
unique_words = set(words)
print("Number of unique words:", len(unique_words))
```

Output:

```
Enter a sentence: I study in university of petroleum and energy studies
Number of unique words: 9
```

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

Program:

```
n = int(input("Enter the number of fruits: "))

s1 = set()
s2 = set()

print("Enter fruits for set s1:")
for _ in range(n):
    fruit = input()
    s1.add(fruit)

print("Enter fruits for set s2:")
for _ in range(n):
```

```
    fruit = input()
    s2.add(fruit)

print("Fruits which are in both sets s1 and s2:", s1.intersection(s2))
print("Fruits only in s1 but not in s2:", s1.difference(s2))
print("Count of all fruits from s1 and s2:", len(s1.union(s2)))
```

Output:

```
Enter the number of fruits: 3
Enter fruits for set s1:
Apple
banana
mango
Enter fruits for set s2:
Mango
papaya
Apple
Fruits which are in both sets s1 and s2: {'Apple'}
Fruits only in s1 but not in s2: {'mango', 'banana'}
Count of all fruits from s1 and s2: 5
```

8. Take two sets and apply various set operations on them :

S1 = {Red ,yellow, orange , blue }

S2 = {violet, blue , purple}

Program:

```
S1 = {"Red", "Yellow", "Orange", "Blue"}
S2 = {"Violet", "Blue", "Purple"}

print("Union of S1 and S2:", S1.union(S2))
print("Intersection of S1 and S2:", S1.intersection(S2))
print("Difference between S1 and S2:", S1.difference(S2))
print("Symmetric difference between S1 and S2:", S1.symmetric_difference(S2))
```

Output:

```
Union of S1 and S2: {'Red', 'Purple', 'Yellow', 'Violet', 'Blue', 'Orange'}
Intersection of S1 and S2: {'Blue'}
Difference between S1 and S2: {'Red', 'Yellow', 'Orange'}
Symmetric difference between S1 and S2: {'Red', 'Purple', 'Yellow', 'Violet', 'Orange'}
```

Experiment 6: Lists, tuples, dictionary

1. Scan n values in range 0-3 and print the number of times each value has occurred.

   Program:

```
n = int(input("Enter the number of values: "))
count = [0] * 4

for _ in range(n):
    value = int(input("Enter a value (0-3): "))
    count[value] += 1

for i in range(4):
    print("Number of times", i, "has occurred:", count[i])
```

   Output:

```
Enter the number of values: 2
Enter a value (0-3): 1
Enter a value (0-3): 2
Number of times 0 has occurred: 0
Number of times 1 has occurred: 1
Number of times 2 has occurred: 1
Number of times 3 has occurred: 0
```

2. Create a tuple to store n numeric values and find average of all values.

   Program:

```
n = int(input("Enter the number of values: "))
values = tuple(
    map(int, input("Enter the values separated by space: ").split()))

average = sum(values) / n
print("Average of the values:", average)
```

   Output:

```
Enter the number of values: 4
Enter the values separated by space: 6 8 7 4
Average of the values: 6.25
```

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.

Program:

```
N = int(input("Enter the number of students: "))
scores = list(map(int, input("Enter the scores separated by space:
").split()))

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

print("Runner-up score:", unique_scores[1])
```

Output:

```
Enter the number of students: 3
Enter the scores separated by space: 87 54 69
Runner-up score: 69
```

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.

   Program:

```
n = int(input("Enter the number of persons: "))
person_dict = {}

for _ in range(n):
    name = input("Enter name: ")
    city = input("Enter city: ")
    person_dict[name] = city


print("Names:", list(person_dict.keys()))
```

```
print("Cities:", list(person_dict.values()))


print("Student name and city:")
for name, city in person_dict.items():
    print(name + ":", city)


city_count = {}
for city in person_dict.values():
    city_count[city] = city_count.get(city, 0) + 1

print("Number of students in each city:")
for city, count in city_count.items():
    print(city + ":", count)
```

Output:

```
Enter the number of persons: 2
Enter name: Varun
Enter city: Patna
Enter name: Priyanshu
Enter city: Varanasi
Names: ['Varun', 'Priyanshu']
Cities: ['Patna', 'Varanasi']
Student name and city:
Varun: Patna
Priyanshu: Varanasi
Number of students in each city:
Patna: 1
Varanasi: 1
```

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.

print movies directed by a particular director.

Program:

```python
n = int(input("Enter the number of movies: "))

movies = []
for i in range(n):
    name = input("Enter movie name: ")
    year = int(input("Enter year of release: "))
    director = input("Enter director name: ")
    cost = float(input("Enter production cost: "))
    collection = float(input("Enter collection made: "))
    movies.append({"Name": name, "Year": year, "Director": director,
                   "Cost": cost, "Collection": collection})

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

# b) Display name of movies released before 2015
print("Movies released before 2015:")
for movie in movies:
    if movie["Year"] < 2015:
        print(movie["Name"])

# c) Print movies that made a profit
print("Movies that made a profit:")
for movie in movies:
    if movie["Collection"] > movie["Cost"]:
        print(movie["Name"])

# d) Print movies directed by a particular director
director_name = input("Enter director's name: ")
print("Movies directed by", director_name + ":")
for movie in movies:
    if movie["Director"] == director_name:
        print(movie["Name"])
```

Output:

```
Enter the number of movies: 1
Enter movie name: Sholay
Enter year of release: 1975
Enter director name: Ramesh Sippy
Enter production cost: 30000000
Enter collection made: 35000000
All movie details:
{'Name': 'Sholay', 'Year': 1975, 'Director': 'Ramesh Sippy', 'Cost': 30000000.0, 'Collection': 35000000.0}
Movies released before 2015:
Sholay
Movies that made a profit:
Sholay
Enter director's name: Ramesh Sippy
Movies directed by Ramesh Sippy:
Sholay
```

Experiment 7: Functions

1. Write a Python function to find the maximum and minimum numbers from a sequence of numbers. (Note: Do not use built-in functions.)

   Program:

```
def find_max_min(numbers):
    if not numbers:
        return None, None
    max_num = min_num = numbers[0]
    for num in numbers:
        if num > max_num:
            max_num = num
        if num < min_num:
            min_num = num
    return max_num, min_num
, 8, 90, 12, 56]
max_num, min_num = find_max_min(numbers)
print("Maximum number:", max_num)
print("Minimum number:", min_num)
```

   Output:

```
Maximum number: 90
Minimum number: 6
```

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.

   Program:

```
def sum_of_cubes(n):
    if n <= 0:
        return 0
    return sum(i**3 for i in range(1, n))
num = int(input("Enter a positive integer: "))
print("Sum of cubes of positive integers smaller than",
      num, "is:", sum_of_cubes(num))
```

   Output:

```
Enter a positive integer: 34
Sum of cubes of positive integers smaller than 34 is: 314721
```

3. Write a Python function to print 1 to n using recursion. (Note: Do not use loop)

   Program:

```
def print_numbers(n):
    if n > 0:
        print_numbers(n - 1)
        print(n)
n = int(input("Enter a number: "))
print("Numbers from 1 to", n, "using recursion:")
print_numbers(n)
```

   Output:

```
Enter a number: 10
Numbers from 1 to 10 using recursion:
1
2
3
4
5
6
7
8
9
10
```

4. Write a recursive function to print Fibonacci series upto n terms.

   Program:

```
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)
terms = int(input("Enter the number of terms: "))
print("Fibonacci series up to", terms, "terms:")
for i in range(terms):
    print(fibonacci(i))
```

   Output:

```
Enter the number of terms: 12
Fibonacci series up to 12 terms:
0
1
1
2
3
5
8
13
21
34
55
89
```

5. Write a lambda function to find volume of cone.

   Program:

```
def cone_volume(r, h): return (1/3) * 3.14159 * r**2 * h
# Example usage:
radius = float(input("Enter the radius of the cone: "))
height = float(input("Enter the height of the cone: "))
print("Volume of the cone:", cone_volume(radius, height))
```

   Output:

```
Enter the radius of the cone: 5
Enter the height of the cone: 8
Volume of the cone: 209.4393333333333
```

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)

   Program:

```
def max_min_tuple(lst): return (max(lst), min(lst))


# Example usage:
input_list = [10, 6, 8, 90, 12, 56]
result = max_min_tuple(input_list)
print("Max and Min from the list:", result)
```

   Output:

```
Max and Min from the list: (90, 6)
```

7. Write functions to explain mentioned concepts:

   a. Keyword argument

   Program:

```
def example_keyword_arg(name, age):
    print("Name:", name)
    print("Age:", age)

example_keyword_arg(age=25, name="Alice")
```

Output:

```
Name: Alice
Age: 25
```

   b. Default argument

   Program:

```
def example_default_arg(name, age=30):
    print("Name:", name)
    print("Age:", age)

example_default_arg("Bob")
```

Output:

```
Name: Bob
Age: 30
```

   c. Variable length argument

   Program:

```
def example_var_length_arg(*args):
    print("Arguments:")
    for arg in args:
        print(arg)

example_var_length_arg(1, 2, "Hello", [4, 5, 6])
```

Output:

```
Arguments:
1
2
Hello
[4, 5, 6]
```

Experiment 8: File Handling and Exception Handling

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

Program:

```
with open("names.txt", "w") as file:
    file.write("Varun\n")
    file.write("Aryan\n")
    file.write("Priyanshu\n")
    file.write("Rishi\n")
    file.write("Jay\n")

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

count_vowel_starting_names = sum(
    1 for name in names if name[0].lower() in 'aeiou')
print("Number of names starting with a vowel:", count_vowel_starting_names)
longest_name = max(names, key=len).strip()
print("Longest name:", longest_name)
```

Output:

```
Number of names: 5
Number of names starting with a vowel: 1
Longest name: Priyanshu
```

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

Program:

```
with open("numbers.txt", "w") as file:
    file.write("10\n")
    file.write("20\n")
    file.write("30\n")
    file.write("150\n")
    file.write("200\n")

with open("numbers.txt", "r") as file:
```

```
    numbers = [int(line) for line in file.readlines()]
    max_number = max(numbers)
    print("Max number:", max_number)

average_number = sum(numbers) / len(numbers)
print("Average:", average_number)

count_greater_than_100 = sum(1 for num in numbers if num > 100)
print("Number of numbers greater than 100:", count_greater_than_100)
```

Output:

```
Max number: 200
Average: 82.0
Number of numbers greater than 100: 2
```

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

Program:

```
with open("city.txt", "r") as file:
    print("City Details:")
    for line in file:
        city, population, area = line.strip().split()
        print("City:", city)
        print("Population (in lakhs):", population)
        print("Area (in sq KM):", area)
        print()

with open("city.txt", "r") as file:
    print("City names with population more than 10 Lakhs:")
```

```
    for line in file:
        city, population, _ = line.strip().split()
        if float(population) > 10:
            print(city)

with open("city.txt", "r") as file:
    total_area = sum(float(line.strip().split()[2]) for line in file)
    print("Sum of areas of all cities:", total_area, "sq KM")
```

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

   Program:

```
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, b)

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

   Output:

```
Enter the number of test cases: 2
10 2
5
24 4
6
```

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

Program:

```python
try:
    with open("demofile.txt", "r") as file:
        data = file.read()
except FileNotFoundError:
    print("Error: File not found")

try:
    my_list = [1, 2, 3]
    print(my_list[4])
except IndexError:
    print("Error: Index out of range")

try:
    result = 10 / 0
except ZeroDivisionError:
    print("Error: Division by zero")

try:
    num = int("hello")
except ValueError:
    print("Error: Invalid literal for int() with base 10")
```

Output:

```
Error: Index out of range
Error: Division by zero
Error: Invalid literal for int() with base 10
```

Experiment 9: Classes and objects

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.

Program:

```
class Student:
    def __init__(self, name, sap_id, phy_marks, chem_marks, maths_marks):
        self.name = name
        self.sap_id = sap_id
        self.marks = {'Physics': phy_marks, 'Chemistry': chem_marks, 'Maths':
maths_marks}

    def display_details(self):
        print("Name:", self.name)
        print("SAP ID:", self.sap_id)
        print("Marks:")
        for subject, marks in self.marks.items():
            print(subject + ":", marks)

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

print("\nDetails of all students:")
for student in students:
    student.display_details()
    print()
```

Output:

```
Enter student's name: Varun
Enter SAP ID: 500123408
Enter Physics marks: 84
Enter Chemistry marks: 95
Enter Maths marks: 67
Enter student's name: Aryan
Enter SAP ID: 500121478
Enter Physics marks: 65
Enter Chemistry marks: 84
Enter Maths marks: 75
Enter student's name: Jay
Enter SAP ID: 500147895
Enter Physics marks: 45
Enter Chemistry marks: 48
Enter Maths marks: 61

Details of all students:
Name: Varun
SAP ID: 500123408
Marks:
Physics: 84.0
Chemistry: 95.0
Maths: 67.0

Name: Aryan
SAP ID: 500121478
Marks:
Physics: 65.0
Chemistry: 84.0
Maths: 75.0

Name: Jay
SAP ID: 500147895
Marks:
Physics: 45.0
Chemistry: 48.0
Maths: 61.0
```

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.

Program:

```
class Student:
    def __init__(self, name, sap_id, phy_marks, chem_marks, maths_marks):
        self.name = name
        self.sap_id = sap_id
        self.marks = {'Physics': phy_marks, 'Chemistry': chem_marks, 'Maths':
maths_marks}

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

```
        print("Marks:")
        for subject, marks in self.marks.items():
            print(subject + ":", marks)

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

    def display_result(self):
        percentage = self.marks_percentage()
        result = "Pass" if all(marks >= 40 for marks in self.marks.values())
else "Fail"
        print("Result:", result)
        print("Percentage:", round(percentage, 2), "%")

class StudentsList:
    def __init__(self):
        self.students = []

    def add_student(self, name, sap_id, phy_marks, chem_marks, maths_marks):
        student = Student(name, sap_id, phy_marks, chem_marks, maths_marks)
        self.students.append(student)

    def display_all_students(self):
        for student in self.students:
            student.display_details()
            student.display_result()
            print()

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

students_list = StudentsList()
for _ in range(3):
    name = input("Enter student's name: ")
    sap_id = input("Enter SAP ID: ")
    phy_marks = float(input("Enter Physics marks: "))
    chem_marks = float(input("Enter Chemistry marks: "))
    maths_marks = float(input("Enter Maths marks: "))
    students_list.add_student(name, sap_id, phy_marks, chem_marks,
maths_marks)

print("\nDetails of all students:")
students_list.display_all_students()
```

```
print("Class Average Percentage:", round(students_list.class_average(), 2),
"%")
```

Output:

```
Enter Physics marks: 74
Enter Chemistry marks: 85
Enter Maths marks: 95
Enter student's name: Aryan
Enter SAP ID: 500148932
Enter Physics marks: 50
Enter Chemistry marks: 46
Enter Maths marks: 74
Enter student's name: Jay
Enter SAP ID: 500487596
Enter Physics marks: 98
Enter Chemistry marks: 95
Enter Maths marks: 93

Details of all students:
Name: Varun
SAP ID: 500123408
Marks:
Physics: 74.0
Chemistry: 85.0
Maths: 95.0
Result: Pass
Percentage: 84.67 %

Name: Aryan
SAP ID: 500148932
Marks:
Physics: 50.0
Chemistry: 46.0
Maths: 74.0
Result: Pass
Percentage: 56.67 %

Name: Jay
SAP ID: 500487596
Marks:
Physics: 98.0
Chemistry: 95.0
Maths: 93.0
Result: Pass
Percentage: 95.33 %

Class Average Percentage: 78.89 %
```

3. Create programs to implement different types of inheritances.

   Program:

```
# Single Inheritance
class Person:
    def __init__(self, name, age):
```

```
        self.name = name
        self.age = age

class Student(Person):
    def __init__(self, name, age, roll_no):
        super().__init__(name, age)
        self.roll_no = roll_no
# Multiple Inheritance
class A:
    def method_A(self):
        print("Method A")
class B:
    def method_B(self):
        print("Method B")
class C(A, B):
    def method_C(self):
        print("Method C")
# Hierarchical Inheritance
class Animal:
    def speak(self):
        pass
class Dog(Animal):
    def speak(self):
        print("Bark")
class Cat(Animal):
    def speak(self):
        print("Meow")
# Hybrid Inheritance
class Vehicle:
    pass
class LandVehicle(Vehicle):
    pass
class AirVehicle(Vehicle):
    pass
class Car(LandVehicle):
    pass
class Plane(AirVehicle):
    pass
```

4. Create a class to implement method Overriding.

Program:

```
class Animal:
    def speak(self):
        print("Animal speaks")
class Dog(Animal):
    def speak(self):
        print("Dog barks")
class Cat(Animal):
```

```
    def       speak(self):
        print("Cat meows")
dog = Dog()
cat = Cat()
dog.speak()
cat.speak()
```

Output:

```
Dog barks
Cat meows
```

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

e.g. if

P1(x=10,y=20)

P2(x=12,y=15)

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

Program:

```
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(6, 25)
P2 = Point(12, 19)
P3 = P1 + P2
print("P3 =", P3)
```

Output:

```
P3 = (18, 44)
```

Experiment 10: Data Analysis and Visualization

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

   Program:

```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6]])
print("Array:")
print(arr)
print("Sum of all elements:", np.sum(arr))
```

   Output:

```
Array:
[[1 2 3]
 [4 5 6]]
Sum of all elements: 21
```

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

   Program:

```
import numpy as np
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Array:")
print(arr)
print("Sum of rows:", np.sum(arr, axis=1))
print("Sum of columns:", np.sum(arr, axis=0))
print("Second maximum element:", np.max(np.unique(arr)[-2]))
```

   Output:

```
Array:
[[1 2 3]
 [4 5 6]
 [7 8 9]]
Sum of rows: [ 6 15 24]
Sum of columns: [12 15 18]
Second maximum element: 8
```

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

   Program:

```
import numpy as np
```

```
matrix1 = np.array([[1, 2], [3, 4]])
matrix2 = np.array([[5, 6], [7, 8]])
result = np.dot(matrix1, matrix2)
print("Matrix Multiplication Result:")
print(result)
```

Output:

```
Matrix Multiplication Result:
[[19 22]
 [43 50]]
```

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

5 86 86 83

Program:

```
import pandas as pd

data = {'X': [2, 3, 4, 5, 6], 'Y': [
    7, 8, 9, 10, 11], 'Z': [12, 13, 14, 15, 16]}
df = pd.DataFrame(data)
powers = pd.DataFrame()
for column in df.columns:
    powers[column] = df[column] ** 2
print("Powers of array values:")
print(powers)
```

Output:

```
Powers of array values:
    X    Y    Z
0   4   49  144
1   9   64  169
2  16   81  196
3  25  100  225
4  36  121  256
```

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

Program:

```
import pandas as pd
exam_data = {
    'name': ['Varun', 'Aryan', 'Priyanshu', 'Jay', 'Rishi', 'Vedant',
'Kartikey', 'Abhinav', 'Yashu', 'Happy'],
    'score': [12.5, 9, 16.5, 11.5, 9, 20, 14.5, 10, 8, 19],
    'attempts': [1, 3, 2, 3, 2, 3, 1, 1, 2, 1],
    'qualify': ['yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no',
'yes']
}
labels = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']
df = pd.DataFrame(exam_data, index=labels)
print("First three rows of the data frame:")
print(df.head(3))
```

Output:

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

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

Program:

```
import numpy as np
import pandas as pd

data = {'A': [1, 2, np.nan, 4, 5],
        'B': [np.nan, np.nan, np.nan, np.nan, np.nan],
        'C': [10, np.nan, 30, np.nan, 50]}
df = pd.DataFrame(data)
print("DataFrame with missing values:")
print(df)

df.replace(to_replace=np.nan, value="No Information", inplace=True)

print("\nDataFrame with missing values replaced:")
print(df)
```

Output:

```
DataFrame with missing values:
     A   B     C
0  1.0 NaN  10.0
1  2.0 NaN   NaN
2  NaN NaN  30.0
3  4.0 NaN   NaN
4  5.0 NaN  50.0

DataFrame with missing values replaced:
                A              B              C
0             1.0  No Information           10.0
1             2.0  No Information  No Information
2  No Information  No Information           30.0
3             4.0  No Information  No Information
4             5.0  No Information           50.0
```

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

Program:

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0, 10)
y = np.random.randint(0, 20, size=10)
plt.plot(x, y)
plt.title("Line Plot")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

x = np.random.randn(100)
y = np.random.randn(100)
plt.scatter(x, y)
plt.title("Scatter Plot")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

data = np.random.randn(1000)
plt.hist(data, bins=30)
plt.title("Histogram")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.show()

x = ['A', 'B', 'C', 'D', 'E']
y = [10, 20, 15, 25, 30]
plt.bar(x, y)
plt.title("Bar Plot")
plt.xlabel("Category")
plt.ylabel("Value")
plt.show()
```

Output:

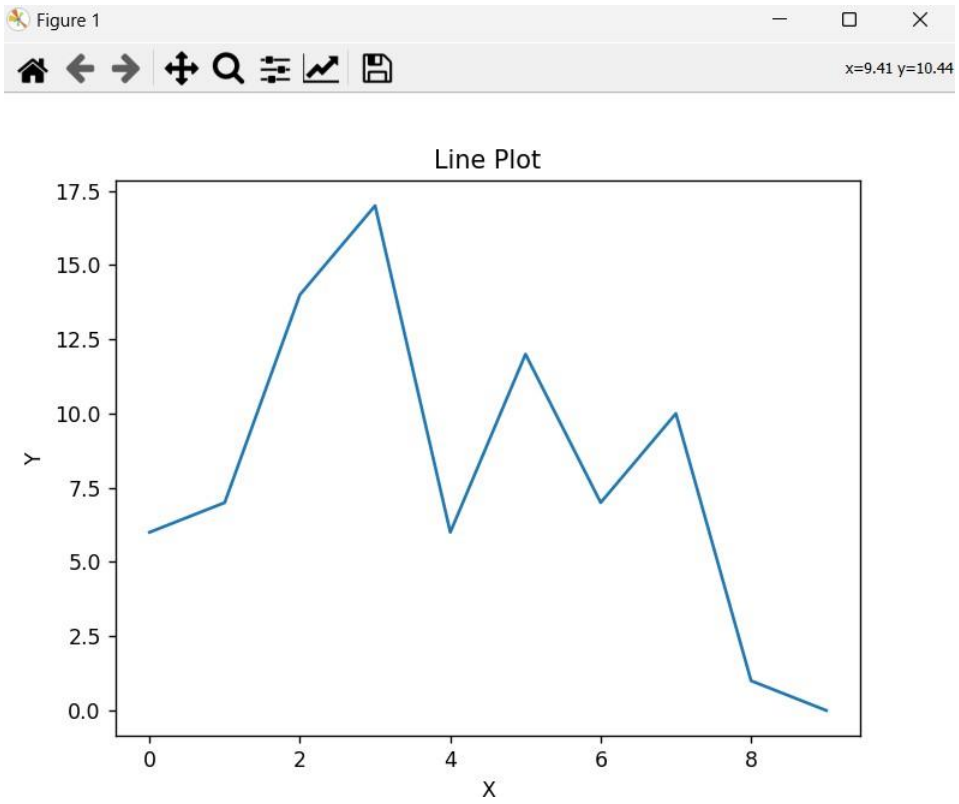
Figure 1
x=9.41 y=10.44
Line Plot
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Y
0
2
4
6
8
X

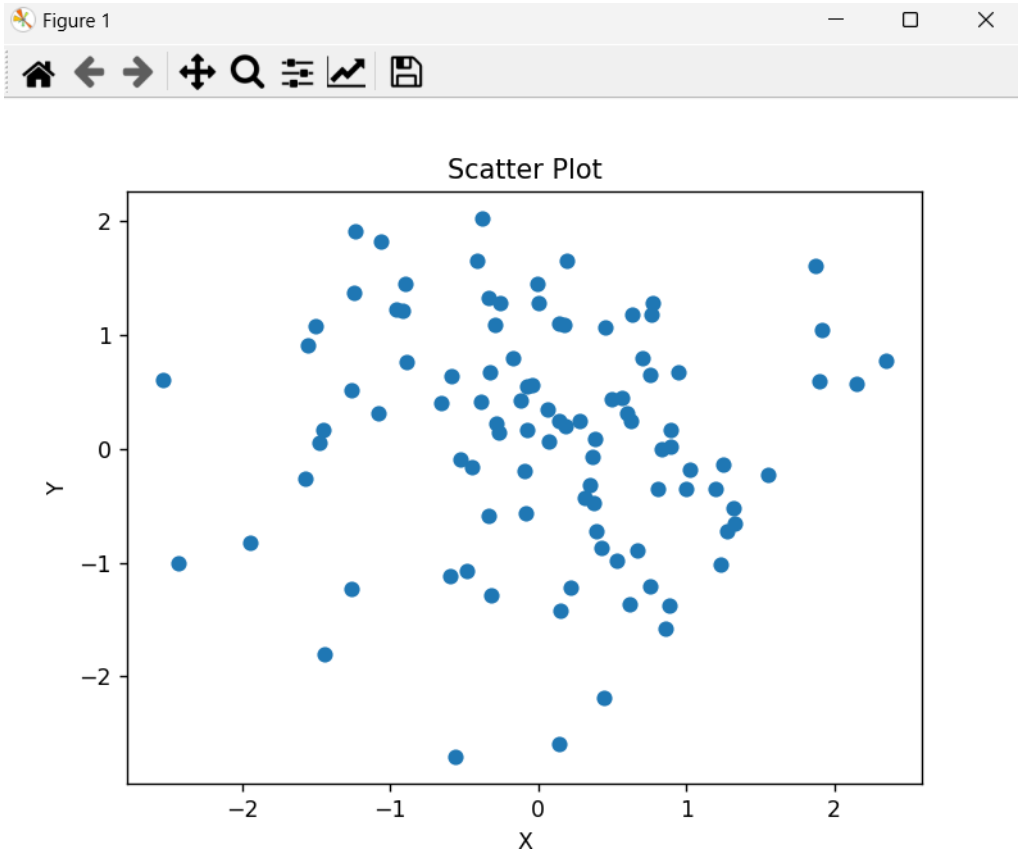
Figure 1
Scatter Plot
2
1
0
−1
−2
Y
−2
−1
0
1
2
X

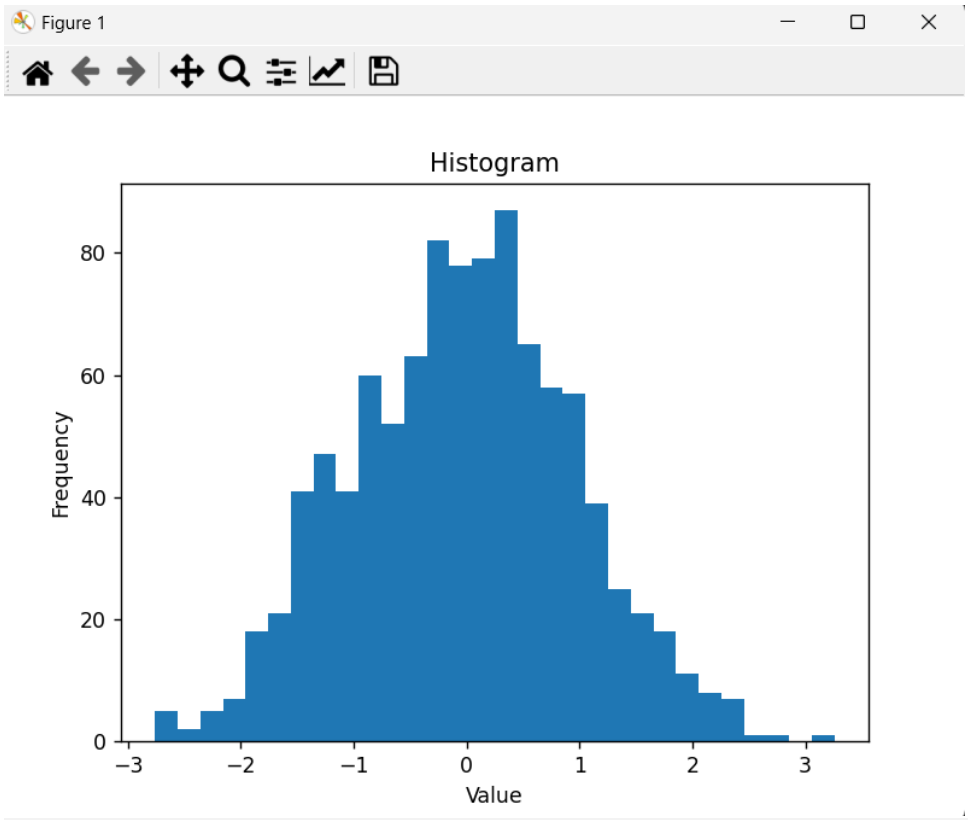
Figure 1
Histogram
Frequency
Value
0
20
40
60
80
−3
−2
−1
0
1
2
3

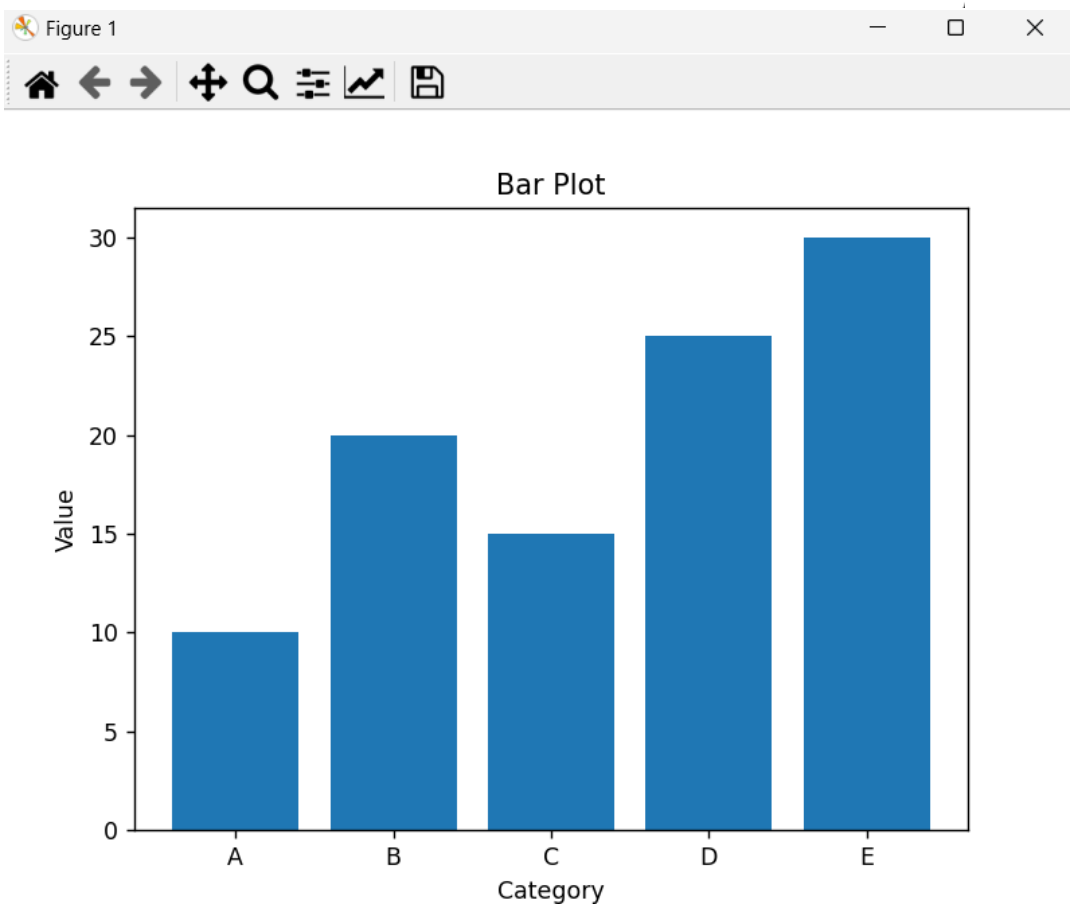
Figure 1
Bar Plot
Value
Category
0
5
10
15
20
25
30
A
B
C
D
E