**University of Petroleum and Energy Studies**

Python Programming Lab

2023-24

Submitted to: Submitted by:

Dr. Sugandha Sharma Name: Harsh Nadkarni

Enrolment no:R2142231560

SAP ID: 500122803

Batch:46

**EXPERIMENT – 1**

#2. Write Python programs to print strings in the given manner:

#a # to print "Hello Everyone!!!"

print("a. Hello Everyone!!!")

#b # to print "Hello

              #world"

print("b. Hello \nworld")

#c # to print "Hello

                  #world"

print("c. Hello\n\tworld")

#d # to print Rohit's DOB is 12/05/1999

print("d. Rohit's Date of birth is 12/05/1999")

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

            Print out the value of x'''

x="Hello"

print(x)

'''4. take different types of datatypes and print them using print() function'''

integer=10

float=10.5

list=['Hello i am learning python']

string= "Hello"

boolean= True

print(integer)

print(float)

print(list)

print(string)

print(boolean)

'''5. take two variables a and b '''

a="Harsh"

b="Nadkarni"

print(a,b)

'''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. '''

a="Harsh"

b="Nadkarni"

c="(Leftie)"

print(a,c,b)

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

name="NIKUNJ BANSAL"

sap="500069944"

dob="13 OCT 1999"

adr="UPES\n\t Bidholi Campus\n\t Pincode: 248007"

prog="AI & ML"

sem="2"

print("Name:",name)

print("SAP ID:",sap)

print("Date of Birth:",dob)

print("Address:",adr)

print("Programme:",prog)

print("Semester:",sem)

**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.

x=9

y=7

z=x+y

print(z)

z=x-y

print(z)

z=x\*y

print(z)

z=x/y

print(z)

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

def ar\_circle(r):

    return 3.14\*r\*r

r=int(input("Enter the radius: "))

print(ar\_circle(r))

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

def solve(x,y):

    return (x+y)\*(x-y)

x=int(input("Enter the value of x: "))

y=int(input("Enter the value of y: "))

print(solve(x,y))

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

def hypotenuse(a,b):

    return(a\*\*2+b\*\*2)

a=int(input("Enter the value of a: "))

b=int(input("Enter the value of b:"))

print("The length of the right Triangle is:", hypotenuse(a,b))

#5.Write a program to find simple interest.

def simple\_interest(p,r,t):

    return(p\*r\*t)/100

p=int(input("Enter the value of p: "))

r=int(input("Enter the value of r: "))

t=int(input("Enter the value of t: "))

print("The simple interest is:", simple\_interest(p,r,t))

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

def ar\_triangle(a,b,c):

    s=(a+b+c)/2

    return (s\*(s-a)\*(s-b)\*(s-c))\*\*0.5

a=int(input("Enter the value of a: "))

b=int(input("Enter the value of b: "))

c=int(input("Enter the value of c: "))

print("The area of the triangle is:", ar\_triangle(a,b,c))

**EXPERIMENT – 3**

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

def check(n):

    if n%3==0 and n%5==0:

        return True

    else:

        return False

n=int(input("Enter the number: "))

print(check(n))

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

def check(n):

    if n%5==0:

        return True

    else:

        return False

n=int(input("Enter the number: "))

print(check(n))

#3. Find the greatest among two numbers. If numbers are equal than print “numbers are equal”.

def greatest(a,b):

    if a>b:

        return a

    elif a<b:

        return b

    else:

        return "Numbers are equal"

a=int(input("Enter the value of a: "))

b=int(input("Enter the value of b: "))

print(greatest(a,b))

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

def greatest(a,b,c):

    if a>b and a>c:

        return a

    elif b>a and b>c:

        return b

    else:

        return c

a=int(input("Enter the value of a: "))

b=int(input("Enter the value of b: "))

c=int(input("Enter the value of c: "))

print(greatest(a,b,c))

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

def roots(a,b,c):

    d=b\*\*2-4\*a\*c

    if d>0:

        return "Real Roots"

    elif d==0:

        return "Real and Equal Roots"

    else:

        return "Imaginary Roots"

a=int(input("Enter the value of a: "))

b=int(input("Enter the value of b: "))

c=int(input("Enter the value of c: "))

print(roots(a,b,c))

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

def leap\_year(year):

    if year%4==0 and year%100!=0 or year%400==0:

        return "Leap Year"

    else:

        return "Not a Leap Year"

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

print(leap\_year(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'''

def next\_date(day,month,year):

    if year%4==0 and year%100!=0 or year%400==0:

        leap=True

    else:

        leap=False

    if month==1 or month==3 or month==5 or month==7 or month==8 or month==10 or month==12:

        max\_days=31

    elif month==4 or month==6 or month==9 or month==11:

        max\_days=30

    else:

        if leap:

            max\_days=29

        else:

            max\_days=28

    if day<max\_days:

        day+=1

    else:

        day=1

        if month==12:

            month=1

            year+=1

        else:

            month+=1

    return day,month,year

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

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

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

print(next\_date(day,month,year))

#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

Sem: 1

Subject name: Marks

PDS:

70

Python:

80

Chemistry:  90

English:

60

Physics:

50

Percentage: 70%

CGPA:7.0

Grade: '''

def grade\_sheet():

    name=input("Enter the name: ")

    roll\_no=input("Enter the roll number: ")

    sem=input("Enter the semester: ")

    sap=int(input("Enter the SAP ID: "))

    course=input("Enter the course: ")

    pds=int(input("Enter the marks of PDS: "))

    python=int(input("Enter the marks of Python: "))

    chemistry=int(input("Enter the marks of Chemistry: "))

    english=int(input("Enter the marks of English: "))

    physics=int(input("Enter the marks of Physics: "))

    percentage=(pds+python+chemistry+english+physics)/5

    cgpa=percentage/10

    if cgpa>=0 and cgpa<=3.4:

        grade="F"

    elif cgpa>=3.5 and cgpa<=5.0:

        grade="C+"

    elif cgpa>=5.1 and cgpa<=6:

        grade="B"

    elif cgpa>=6.1 and cgpa<=7:

        grade="B+"

    elif cgpa>=7.1 and cgpa<=8:

        grade="A"

    elif cgpa>=8.1 and cgpa<=9:

        grade="A+"

    else:

        grade="O (Outstanding)"

    return name,roll\_no,sem,pds,python,chemistry,english,physics,percentage,cgpa,grade

**EXPERIMENT – 4**

import math

# 1. Find the factorial of a given number.

def fact(n):

    if n==0:

        return 1

    else:

        return n\*fact(n-1)

n=int(input("Enter the number: "))

print(fact(n))

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

def armstrong(n):

    sum=0

    temp=n

    while temp>0:

        digit=temp%10

        sum+=digit\*\*3

        temp//=10

    if n==sum:

        return "It is an Armstrong number"

    else:

        return "It is not an Armstrong number"

n=int(input("Enter the number: "))

print(armstrong(n))

# 3. Print Fibonacci series up to given term.

def fibonacci(n):

    a=0

    b=1

    if n==1:

        print(a)

    else:

        print(a)

        print(b)

        for i in range(2,n):

            c=a+b

            a=b

            b=c

            print(c)

n=int(input("Enter the number: "))

fibonacci(n)

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

def prime(n):

    if n>1:

        for i in range(2, int(math.sqrt(n))+1):

            if n%i==0:

                return "It is not a prime number"

        else:

            return "It is a prime number"

    else:

        return "It is not a prime number"

n=int(input("Enter the number: "))

print(prime(n))

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

def palindrome(n):

    temp=n

    rev=0

    while n>0:

        digit=n%10

        rev=rev\*10+digit

        n=n//10

    if temp==rev:

        return "It is a palindrome number"

    else:

        return "It is not a palindrome number"

n=int(input("Enter the number: "))

print(palindrome(n))

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

def sum\_of\_digit(n):

    sum=0

    while n>0:

        digit=n%10

        sum+=digit

        n=n//10

    return sum

n=int(input("Enter the number: "))

print(sum\_of\_digit(n))

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

def divisible(n):

    for i in range(1,n+1):

        if i%5==0 or i%7==0:

            print(i)

n=100

divisible(n)

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

def strings(s):

    return s.upper()

s=input("Enter the string: ")

print(strings(s))

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

def prime(n):

    for i in range(2,n+1):

        if i>1:

            for j in range(2, int(math.sqrt(i))+1):

                if i%j==0:

                    break

            else:

                print(i)

n=100

prime(n)

#10. Print the table for a given number:

#   5 \* 1 = 5

#   5 \* 2 = 10………..

def table(n):

    for i in range(1,11):

        print(n,"x",i,"=",n\*i)

n=int(input("Enter the number: "))

table(n)

**EXPERIMENT-5**

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

def capital(s):

    count=0

    for i in s:

        if i.isupper():

            count+=1

    return count

s=input("Enter the string: ")

print(capital(s))

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

def vowels(s):

    count=0

    for i in s:

        if i in "aeiou,AEIOU":

            count=count+1

    return count

s=input("Enter the string: ")

print(vowels(s))

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

def words(s):

    return s.split()

s=input("Enter the sentence: ")

print(words(s))

#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

def substring(s,sub):

    count=0

    for i in range(len(s)):

        if s[i:i+len(sub)]==sub:

            count+=1

    return count

s=input("Enter the string: ")

sub=input("Enter the substring: ")

print(substring(s,sub))

#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

def occurence(s):

    d={}

    for i in s:

        if i in d:

            d[i]+=1

        else:

            d[i]=1

    return d

s=input("Enter the string: ")

print(occurence(s))

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

def unique(s):

    return len(set(s.split()))

s=input("Enter the sentence: ")

print(unique(s))

# 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

def fruits(s1,s2):

    print("Fruits in both sets: ",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)))

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

s1=set()

s2=set()

for i in range(n):

    s1.add(input("Enter the fruit: "))

for i in range(n):

    s2.add(input("Enter the fruit: "))

fruits(s1,s2)

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

#S1 = {Red ,yellow, orange , blue }

#S2 = {violet, blue , purple}

def set\_operations(s1,s2):

    print("Union: ",s1.union(s2))

    print("Intersection: ",s1.intersection(s2))

    print("Difference: ",s1.difference(s2))

    print("Symmetric Difference: ",s1.symmetric\_difference(s2))

s1={"Red","yellow","orange","blue"}

s2={"violet","blue","purple"}

set\_operations(s1,s2)

**EXPERIMENT-6**

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

def occurence(n):

    d={}

    for i in range(n):

        if i in d:

            d[i]+=1

        else:

            d[i]=1

    return d

n=int(input("Enter the range: "))

print(occurence(n))

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

def average(n):

    sum=0

    for i in range(n):

        sum+=int(input("Enter the value: "))

    return sum/n

n=int(input("Enter the range: "))

print(average(n))

# 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.

def runner\_up(n):

    scores=[]

    for i in range(n):

        scores.append(int(input("Enter the score:")))

    scores.sort()

    print("The List of Score is:",scores)

    return scores[-2]

n=int(input("Enter the range:"))

print("The Runner up is:",runner\_up(n))

# 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.

def persons(n):

    d={}

    occurence={}

    for i in range(n):

        name=input("Enter the name:")

        city=input("Enter the city:")

        d[name]=city

    print("The names are:",d.keys())

    print("The city names are:",d.values())

    print("The student name and city are:",d)

    print("The number of students in each city are:",occurence(n))

n=int(input("Enter the range:"))

persons(n)

'''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.'''

def movies(n):

    d={}

    for i in range(n):

        name=input("Enter the name of movie:")

        year=int(input("Enter the year of movie:"))

        director=input("Enter the director of movie:")

        production\_cost=int(input("Enter the production cost of movie:"))

        collection=int(input("Enter the collection of movie:"))

        d[name]=[year,director,production\_cost,collection]

    print("The movie details are:",d)

    print("The name of movies released before 2015 are:",[i for i in d if d[i][0]<2015])

    print("The movies that made profit are:",[i for i in d if d[i][3]>d[i][2]])

    director=input("Enter the director name:")

    print("The movies directed by director are:",[i for i in d if d[i][1]==director])

n=int(input("Enter the range:"))

movies(n)

**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.)

def max\_min(n):

    max=n[0]

    min=n[0]

    for i in n:

        if i>max:

            max=i

        elif i<min:

            min=i

    return max,min

n=[1,2,3,4,5,6,7,8,9]

print(max\_min(n))

#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.

def cube(n):

    sum=0

    for i in range(1,n):

        sum+=i\*\*3

    return sum

n=int(input("Enter the number: "))

print(cube(n))

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

def print\_n(n):

    if n>0:

        print\_n(n-1)

        print(n)

n=int(input("Enter the number: "))

print\_n(n)

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

def fibonacci(n):

    if n<=1:

        return n

    else:

        return fibonacci(n-1)+fibonacci(n-2)

n=int(input("Enter the number: "))

for i in range(n):

    print(fibonacci(i))

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

volume=lambda r,h:3.14\*r\*r\*h/3

r=int(input("Enter the radius: "))

h=int(input("Enter the height: "))

print("The volume of cone is:",volume(r,h))

#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)

max\_min=lambda n:(max(n),min(n))

n=list(map(int,input("Enter the numbers: ").split()))

print(max\_min(n))

#7. Write functions to explain mentioned concepts:

#a. Keyword argument

#b. Default argument

#c. Variable length argument

#a. Keyword argument

def keyword(name,age):

    print("Name:",name)

    print("Age:",age)

keyword(name="Ram",age=20)

#b. Default argument

def default(name,age=20):

    print("Name:",name)

    print("Age:",age)

default("Ram")

#c. Variable length argument

def variable(\*args):

    for i in args:

        print(i)

variable(1,2,3,4,5)

**EXPERIMENT-8**

'''1. Add few names, one name in each row, in “name.txt file”.

a. Count no of names

b. Count all names starting with vowel

c. Find longest name'''

def count\_names():

    with open('name.txt', 'r') as file:

        names = file.readlines()

        print(f'Number of names: {len(names)}')

        print(f'Number of names starting with vowel: {len([name for name in names if name[0].lower() in "aeiou"])}')

        print(f'Longest name: {max(names, key=len)}')

def main():

    count\_names()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''2. Store integers in a file.

a. Find the max number

b. Find average of all numbers

c. Count number of numbers greater than 100'''

def count\_numbers():

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

        numbers = [int(number) for number in file.readlines()]

        print(f'Max number: {max(numbers)}')

        print(f'Average of all numbers: {sum(numbers) / len(numbers)}')

        print(f'Number of numbers greater than 100: {len([number for number in numbers if number > 100])}')

def main():

    count\_numbers()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''3. Assume a file city.txt with details of 5 cities in given format (cityname population(in

lakhs) area(in sq KM) ):

Example:

Dehradun 5.78 308.20

Delhi 190 1484

……………

Open file city.txt and read to:

a. Display details of all cities

b. Display city names with population more than 10Lakhs

c. Display sum of areas of all cities'''

def city\_details():

    with open('city.txt', 'r') as file:

        cities = [city.strip().split() for city in file.readlines()]

        print('Details of all cities:')

        for city in cities:

            print(f'City: {city[0]}, Population: {city[1]} Lakhs, Area: {city[2]} sq KM')

        print('City names with population more than 10 Lakhs:', [city[0] for city in cities if float(city[1]) > 10])

        print('Sum of areas of all cities:', sum([float(city[2]) for city in cities]))

def main():

    city\_details()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''4.  Input two values from user where the first line contains N, the number of test

cases. The next N lines contain the space separated values of a and b. Perform

integer division and print a/b. Handle exception in case of ZeroDivisionError or

ValueError.

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'''

def integer\_division():

    n = int(input('Enter number of test cases: '))

    for \_ in range(n):

        try:

            a, b = map(int, input().split())

            print(a // b)

        except ZeroDivisionError:

            print('Error Code: integer division or modulo by zero')

        except ValueError:

            print('Error Code: invalid literal for int() with base 10')

def main():

    integer\_division()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

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

def file\_handling():

    try:

        with open('file.txt', 'r') as file:

            print(file.read())

    except FileNotFoundError:

        print('File not found')

    except PermissionError:

        print('Permission denied')

    except Exception as e:

        print(f'Error: {e}')

def main():

    file\_handling()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**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.'''

class Student:

    def \_\_init\_\_(self, name, sap\_id, marks):

        self.name = name

        self.sap\_id = sap\_id

        self.marks = marks

    def display(self):

        print(f'Name: {self.name}')

        print(f'SAP ID: {self.sap\_id}')

        print(f'Physics: {self.marks[0]}')

        print(f'Chemistry: {self.marks[1]}')

        print(f'Maths: {self.marks[2]}')

def main():

    students = []

    for \_ in range(3):

        name = input('Enter name: ')

        sap\_id = input('Enter SAP ID: ')

        marks = list(map(int, input('Enter marks in Physics, Chemistry, Maths: ').split()))

        students.append(Student(name, sap\_id, marks))

    for student in students:

        student.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''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. '''

class Student:

    def \_\_init\_\_(self, name, sap\_id, marks):

        self.name = name

        self.sap\_id = sap\_id

        self.marks = marks

    def display(self):

        print(f'Name: {self.name}')

        print(f'SAP ID: {self.sap\_id}')

        print(f'Physics: {self.marks[0]}')

        print(f'Chemistry: {self.marks[1]}')

        print(f'Maths: {self.marks[2]}')

    def marks\_percentage(self):

        return sum(self.marks) / 3

    def result(self):

        if all(mark > 40 for mark in self.marks):

            return 'Pass'

        else:

            return 'Fail'

def average(students):

    return sum(student.marks\_percentage() for student in students) / len(students)

def main():

    students = []

    n = int(input('Enter number of students: '))

    for \_ in range(n):

        name = input('Enter name: ')

        sap\_id = input('Enter SAP ID: ')

        marks = list(map(int, input('Enter marks in Physics, Chemistry, Maths: ').split()))

        students.append(Student(name, sap\_id, marks))

    for student in students:

        student.display()

        print(f'Marks percentage: {student.marks\_percentage()}')

        print(f'Result: {student.result()}')

    print(f'Average marks percentage of the class: {average(students)}')

if \_\_name\_\_ == '\_\_main\_\_':

    main()

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

# Single inheritance

class A:

    def display(self):

        print('Class A')

class B(A):

    def display(self):

        print('Class B')

def main():

    obj = B()

    obj.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Multiple inheritance

class A:

    def display(self):

        print('Class A')

class B:

    def display(self):

        print('Class B')

class C(A, B):

    pass

def main():

    obj = C()

    obj.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Multilevel inheritance

class A:

    def display(self):

        print('Class A')

class B(A):

    def display(self):

        print('Class B')

class C(B):

    pass

def main():

    obj = C()

    obj.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Hierarchical inheritance

class A:

    def display(self):

        print('Class A')

class B(A):

    pass

class C(A):

    pass

def main():

    obj1 = B()

    obj1.display()

    obj2 = C()

    obj2.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

# Hybrid inheritance

class A:

    def display(self):

        print('Class A')

class B(A):

    pass

class C(A):

    pass

class D(B, C):

    pass

def main():

    obj = D()

    obj.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''4. Create a class to implement method Overriding.'''

class A:

    def display(self):

        print('Class A')

class B(A):

    def display(self):

        print('Class B')

def main():

    obj = B()

    obj.display()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''5. Create a class for operator overloading which adds two Point Objects where Point

has x & y values

e.g. if

P1(x=10,y=20)

P2(x=12,y=15)

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

class Point:

    def \_\_init\_\_(self, x, y):

        self.x = x

        self.y = y

    def \_\_add\_\_(self, other):

        return Point(self.x + other.x, self.y + other.y)

    def \_\_str\_\_(self):

        return f'Point(x={self.x}, y={self.y})'

def main():

    p1 = Point(10, 20)

    p2 = Point(12, 15)

    p3 = p1 + p2

    print(p3)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**EXPERIMENT-10**

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

import numpy as np

def sum\_of\_elements():

    arr = np.array([1, 2, 3, 4, 5])

    print(f'Sum of all elements in the array: {arr.sum()}')

def main():

    sum\_of\_elements()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''2.  Create numpy array of (3,3) dimension. Now find sum of all rows & columns

individually. Also find 2nd maximum element in the array.  '''

import numpy as np

def sum\_of\_rows\_columns():

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

    print(f'Array:\n{arr}')

    print(f'Sum of all rows: {arr.sum(axis=1)}')

    print(f'Sum of all columns: {arr.sum(axis=0)}')

    print(f'Second maximum element in the array: {np.partition(arr.flatten(), -2)[-2]}')

def main():

    sum\_of\_rows\_columns()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

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

import numpy as np

def matrix\_multiplication():

    arr1 = np.array([[1, 2], [3, 4]])

    arr2 = np.array([[5, 6], [7, 8]])

    print(f'Matrix 1:\n{arr1}')

    print(f'Matrix 2:\n{arr2}')

    print(f'Matrix multiplication:\n{np.dot(arr1, arr2)}')

def main():

    matrix\_multiplication()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''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'''

import pandas as pd

def powers\_of\_array():

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

    df = pd.DataFrame(data)

    print(df)

def main():

    powers\_of\_array()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''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 '''

import numpy as np

import pandas as pd

def first\_3\_rows():

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

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

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

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

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

    df = pd.DataFrame(exam\_data, index=labels)

    print('First three rows of the data frame:')

    print(df.head(3))

def main():

    first\_3\_rows()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

'''6. Write a Pandas program to find and replace the missing values in a given

DataFrame which do not have any valuable information.'''

import numpy as np

import pandas as pd

def replace\_missing\_values():

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

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

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

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

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

    df = pd.DataFrame(exam\_data, index=labels)

    print('Original DataFrame:')

    print(df)

    df.replace(np.nan, 0, inplace=True)

    print('DataFrame after replacing missing values with 0:')

    print(df)

def main():

    replace\_missing\_values()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

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

import matplotlib.pyplot as plt

def visual\_forms():

    x = [1, 2, 3, 4, 5]

    y = [10, 20, 30, 40, 50]

    plt.plot(x, y)

    plt.xlabel('X-axis')

    plt.ylabel('Y-axis')

    plt.title('Line plot')

    plt.show()

    plt.bar(x, y)

    plt.xlabel('X-axis')

    plt.ylabel('Y-axis')

    plt.title('Bar plot')

    plt.show()

    plt.scatter(x, y)

    plt.xlabel('X-axis')

    plt.ylabel('Y-axis')

    plt.title('Scatter plot')

    plt.show()

    plt.hist(y)

    plt.xlabel('X-axis')

    plt.ylabel('Y-axis')

    plt.title('Histogram')

    plt.show()

def main():

    visual\_forms()

if \_\_name\_\_ == '\_\_main\_\_':

    main()