1. Write a Program to implement Inheritance

class student():

def \_\_init\_\_(self, sno, name):

self.sno = sno

self.name = name

class student1(student):

def \_\_init\_\_(self,sno, name, course,fees):

super().\_\_init\_\_(sno, name)

self.course = course

self.fees=fees

obj = student1(103, "Kiran gupta", "MSC" , 15000)

print('The Student number is:', obj.sno)

print('The Name is:',obj.name)

print('The Course is:', obj.course)

print('The Fees is:', obj.fees)

2. Write a python program to implement Polymorphism

class Parrot:

def fly(self):

print("Parrot can fly")

def swim(self):

print("Parrot can't swim")

class Penguin:

def fly(self):

print("Penguin can't fly")

def swim(self):

print("Penguin can swim")

# common interface

def flying\_test(bird):

bird.fly()

bird.swim()

#instantiate objects

blu = Parrot()

peggy = Penguin()

# passing the object

flying\_test(blu)

flying\_test(peggy)

**3. Write a python to implement Data Encapsulation.**

class Computer:

def \_\_init\_\_(self):

self.\_\_maxprice = 900

def sell(self):

print("Selling Price: {}".format(self.\_\_maxprice))

def setMaxPrice(self, price):

self.\_\_maxprice = price

c = Computer()

c.sell()

# change the price

c.\_\_maxprice = 1000

c.sell()

# using setter function

c.setMaxPrice(1000)

c.sell()

4. Write a programs for String handling functions

import numpy as np

# converting to lowercase

print(np.char.lower(['KCPG', 'COLLEGE']))

# converting to lowercase

print(np.char.lower('KCPG'))

# splitting a string

print(np.char.split('kcpg for students'))

# splitting a string

print(np.char.split('kcpg, for, students', sep = ','))

print(np.char.join(['-', ':'], ['kcpg', 'college']))

array1 = np.array(['iPhone: ', 'price: '])

array2 = np.array(['15', '$900'])

# perform element-wise array string concatenation

result = np.char.add(array1, array2)

print(result)

# define array with three string elements

array1 = np.array(['A', 'B', 'C'])

# repeat each element in array1 two times

result = np.char.multiply(array1, 2)

print(result)

# define an array with three string elements

array1 = np.array(['eric', 'paul', 'sean'])

# capitalize the first letter of each string in array1

result = np.char.capitalize(array1)

print(result)

array1 = np.array(['nEpalI', 'AmeriCAN', 'CaNadIan'])

# convert all string elements to uppercase

result1 = np.char.upper(array1)

# create two arrays of strings

array1 = np.array(['C', 'Python', 'Swift'])

array2 = np.array(['C++', 'Python', 'Java'])

# check if each element of the arrays is equal

result = np.char.equal(array1, array2)

print(result)

5.Write a program for creating Data Frame

import pandas as pd

student = {

"sno": [1,2,3,4,5],

"name": ['raju','ravi','rani','radha','rakhi'],

"marks": [543,587,458,963,478],

"address":['nellore','kavali','gudur','nellore','nellore']

}

df = pd.DataFrame(student)

print(df)

print(df["name"])

6. Write a program for Line chart

import matplotlib.pyplot as plt

import numpy as np

# Define X and Y variable data

x = np.array([1,8])

y = np.array([3,10])

plt.plot(x, y,color="red")

plt.xlabel("X-axis") # add X-axis label

plt.ylabel("Y-axis") # add Y-axis label

plt.title("Linee chart") # add title

plt.show()

7. Write a program for creating Bar chart(vertical & Horizontal)

Vertical Bar Graph

import matplotlib.pyplot as plt

import numpy as np

categories = ['Category A', 'Category B', 'Category C']

values = [15, 24, 30]

plt.bar(categories, values, color='skyblue')

plt.xlabel('Categories')

plt.ylabel('Values')

plt.title('Basic Vertical Bar Graph')

plt.show()

Horizontal Bar Graph

import matplotlib.pyplot as plt

import numpy as np

categories = ['Category A', 'Category B', 'Category C']

values = [15, 24, 30]

plt.barh(categories, values, color='red')

plt.xlabel('Categories')

plt.ylabel('Values')

plt.title('Basic horizontal Bar Graph')

plt.show()

8. Write a program for creating Scatter plot chart

import matplotlib.pyplot as plt

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]

y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

plt.scatter(x, y)

plt.show()