Avi Khandelwal

SPML, Roll no. 204102301

1. **Print 'Hello World!'**

Code:

```
print('Hello World!')
```

2. **User input two numbers a and b. Perform the following algebraic operations c = a + b, d = a − b, e = a ∗ b, f = a/b and g = a%b and print their results.**

Code:

```
a = int(input())
b = int(input())
c = a+b
print("c =",c)
d = a-b
print("d =",d)
e = a*b
print("e =",e)
f = a/b
print("f =",f)
g = a%b
print("g =",g)
```

3. **Find the factorial of a number 'num' and print the result.**

Code:

```
def fact(num):
    if num != 1:
        return num*fact(num-1)
    else:
        return 1

num = int(input("Enter number: "))
f = fact(num)
print("Factorial is",f)
```

4. **Take two user inputs a and b. Write a program to print all the prime numbers in the interval [a,b].**

Code:

```
def check_prime(num):
    flag = True
    if num == 1:
        flag = False
    for i in range(2,num):
        if num%i == 0:
            flag = False
            break
    return flag

a = int(input("Enter a: "))
b = int(input("Enter b: "))

for i in range(a,b+1):
    if check_prime(i) == True:
        print(i)
```

5. **Take two user inputs a and b and find their Lowest Common Multiple(LCM).**

Code:

```
a = int(input("Enter a: "))
b = int(input("Enter b: "))
if a>b:
    max = a
else:
    max = b
lst = []
```

```
div = 2
while div <= max:
    if a%div==0 and b%div==0:
        a = a/div
        b = b/div
        lst.append(div)
        continue
    elif a%div==0 and b%div!=0:
        a = a/div
        lst.append(div)
        continue
    elif a%div!=0 and b%div==0:
        b = b/div
        lst.append(div)
        continue
    else:
        div += 1

prod = 1
for el in lst:
    prod *= el
print("LCM =",prod)
```

6. Create a list of length n = 15. Sort the array in descending order and print the sorted List as well as the sorted indices. Use the bubble sort algorithm.

Code:

```
print("Enter 15 numbers:")
lst = []
lsti = list(range(15))
for i in range(15):
```

```
    num = int(input())
    lst.append(num)

for i in range(14):
    for j in range(14):
        if lst[j] < lst[j+1]:
            temp = lst[j]
            lst[j] = lst[j+1]
            lst[j+1] = temp
            tempi = lsti[j]
            lsti[j] = lsti[j+1]
            lsti[j+1] = tempi
print("Sorted numbers in descending order:",lst)
print("Indices of sorted numbers in descending order: ",lsti)
```

7. Repeat the previous program for sorting in ascending order. Use numpy array instead of list.

Code:

```
from numpy import *

print("Enter 15 numbers:")
arr = array([],dtype = int)
arri = array(range(15))
for i in range(15):
    num = int(input())
    arr = append(arr,num)

for i in range(14):
    for j in range(14):
        if arr[j] < arr[j+1]:
```

```
        temp = arr[j]
        arr[j] = arr[j+1]
        arr[j+1] = temp
        tempi = arri[j]
        arri[j] = arri[j+1]
        arri[j+1] = tempi
print("Sorted numbers in descending order:",arr)
print("Indices of sorted numbers in descending order: ",arri)
```

8. Print a matrix M Rmxn having random values in the given range [-2, 5]. m and n are to be given as userinput.

Code:

```
import numpy as np

m = int(input())
n = int(input())


mat1 = np.random.uniform(-2,5,m*n)
mat1 = mat1.reshape(m,n)
print(mat1,end="\n\n")
```

9. Write a program to multiply two random matrices M1Rmxn, M2Rnxp(Don't use built-in functions). Compare the result obtained with the built-in function.

Code:

```
import numpy as np

m = int(input())
n = int(input())
p = int(input())

mat1 = np.random.uniform(-2,5,m*n)
mat1 = mat1.reshape(m,n)
print(mat1,end="\n\n")

mat2 = np.random.uniform(-2,5,n*p)
mat2 = mat2.reshape(n,p)
print(mat2,end="\n\n")
```

```
mat = np.zeros(m*p)
mat = mat.reshape(m,p)
row = 0
col = 0

while (row < m):
   while (col < p):
      for i in range(n):
         mat[row][col] += mat1[row][i]*mat2[i][col]
      col += 1
   col = 0
   row += 1

print(mat,end="\n\n")

mat1 = np.matrix(mat1)
mat2 = np.matrix(mat2)

print(mat1*mat2)
```

10. **Write File operations :**
    a. **Generate a set of n= 100 random points X = xi , i = 1, . . . n, xiR10.**
    b. **Write the points to a CSV file**

Code:

```
import pandas as pd

import numpy as np

num = np.random.randint(0,10,100)

df = pd.DataFrame(num)

df.to_csv("Desktop/SPML_Sem1/ML/new.csv",index=False)
```

11. **11. Read File operations: • Read the CSV file generated in the previous program to a matrix. Each column of matrix should represent a vector. • Compute the following : C = 1 n Pn i=1(Xi − μ)(Xi − μ) T where μ = 1 n Pn i=1 Xi where i = 1,2,3 ... n. Xi = [xi1,xi2, ... xi10] is a column vector.**

Code:

```
import pandas as pd

import numpy as np
```

```
num = np.random.randint(0,10,100)
df = pd.DataFrame(num)
df.to_csv("Desktop/SPML_Sem1/ML/new.csv",index=False) # Writing into csv file
df1 = pd.read_csv("Desktop/SPML_Sem1/ML/new.csv") #Reading from CSV file
mat = df1.values
mat = np.matrix(mat.reshape(10,10))
sm = np.matrix(np.zeros(100))
sm = sm.reshape(10,10)
for i in range(10):
    col = mat[:,i]
    colT = col.transpose()
    mean = col.sum()/10
    col = col-mean
    colT = colT-mean
    m = col*colT
    sm += m
C= sm/10
print(C)
```

Output:

```
[[ 2.722  0.252 -1.128 -2.158 -1.328  0.302  0.882  0.732 -1.148  0.872]
 [ 0.252  7.682 -0.798 -2.028 -2.598 -1.368 -0.088  3.462 -4.418 -0.098]
 [-1.128 -0.798  6.222 -1.908 -0.678  0.252 -1.668 -2.518  0.502  1.722]
 [-2.158 -2.028 -1.908  7.462  4.992 -0.178 -2.398 -2.948  1.672 -2.508]
 [-1.328 -2.598 -0.678  4.992  7.622  1.252 -6.468 -2.818  1.502 -1.478]
 [ 0.302 -1.368  0.252 -0.178  1.252  4.982 -0.138 -1.088 -3.368 -0.648]
 [ 0.882 -0.088 -1.668 -2.398 -6.468 -0.138  9.042  1.792 -0.388 -0.568]
 [ 0.732  3.462 -2.518 -2.948 -2.818 -1.088  1.792  5.842 -2.438 -0.018]
 [-1.148 -4.418  0.502  1.672  1.502 -3.368 -0.388 -2.438  8.782 -0.698]
 [ 0.872 -0.098  1.722 -2.508 -1.478 -0.648 -0.568 -0.018 -0.698  3.422]]
```

12. **Define a class for a complex number a + jb. Define memeber functions to do basic operations conjugate, absolute value, addition, subtraction, multiplication, division and angle. Define two complex numbers c1, c2 and print the results of the following operations c1 + c2, c1c2, c1c2, c1 c2 , |c1|, |c2|,6 c1, 6 c2.**

Code:

```
from math import *
from numpy import *

class cplex:
   def __init__(self,a,b):
      self.a = a
      self.b = b
   def display(self):
      print(str(self.a)+"+j"+'('+str(self.b)+')')
   def __add__(self,other):
      c = cplex(0,0)
      c.a = self.a+other.a
      c.b = self.b+other.b
      return c
   def __sub__(self,other):
      c = cplex(0,0)
      c.a = self.a-other.a
      c.b = self.b-other.b
      return c
   def __mul__(self,other):
      c = cplex(0,0)
      c.a = self.a*other.a - self.b*other.b
      c.b = self.a*other.b + self.b*other.a
      return c
   def absolute(self):
      result = sqrt(self.a**2 + self.b**2)
      return round(result,2)
   def __truediv__(self,other):
      c = cplex(0,0)
      c = self*other
      mag = other.absolute()
      c.a = round(c.a/(mag**2),2)
      c.b = round(c.b/(mag**2),2)
      return c
   def angle(self):
      theta = arctan(self.b/self.a)
      return round(degrees(theta),2)
   def conj(self):
      self.b = -self.b

c1 = cplex(2,3)
c2 = cplex(4,5)
print("C1 =",end = " ")
c1.display()
print("C2 =",end = " ")
```

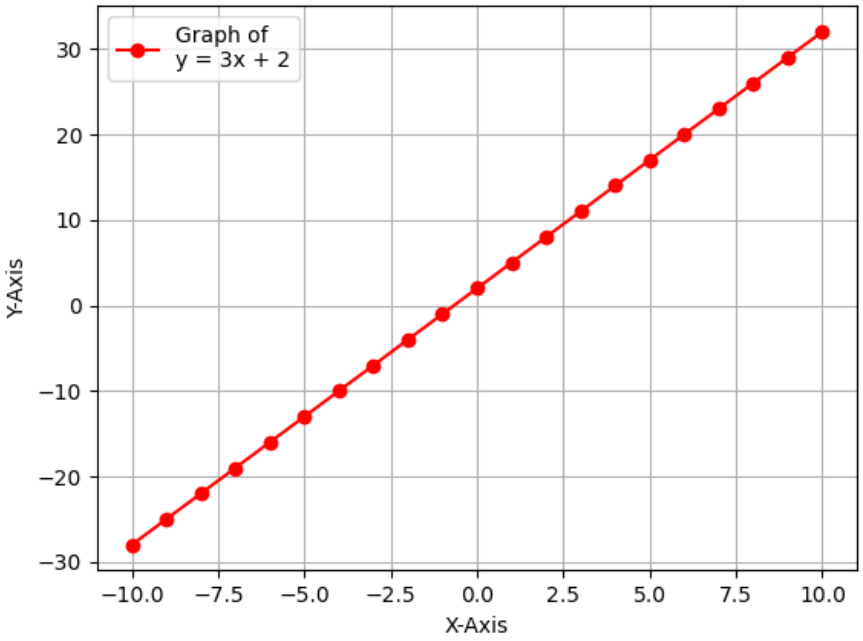
```
c2.display()

c3 = c1+c2
print("Sum is:",end = " ")
c3.display()

c3 = c1-c2
print("Difference is:",end = " ")
c3.display()

c3 = c1*c2
print("Product is:",end = " ")
c3.display()

print("Absolute values of c1 and c2: ",end=" ")
print(c1.absolute(),end=",")
print(c2.absolute())

c3 = c1/c2
print("Division is:",end = " ")
c3.display()

print("Angles of c1 and c2 in degrees: ",end=" ")
print(c1.angle(),end=",")
print(c2.angle())

c1.conj()
c2.conj()
print("Conjugate of c1: ",end = "")
c1.display()
print("Conjugate of c2: ",end = "")
c2.display()
```

13. Plot the function y = 3x + 2 with x [−10, 10]. Use Matplotlib for the same.

Code:

```
import matplotlib.pyplot as plt

import numpy as np

x = np.arange(-10,11)

plt.plot(x,3*x + 2,'o-',label="Graph of\ny = 3x + 2",color='r')

plt.xlabel("X-Axis")

plt.ylabel("Y-Axis")

plt.legend()
```

```
plt.grid(True)

plt.show()
```



14. Scatter plot all the points.

• Generate a set of n = 100 points, X = xi , i = 1,2, ... n, xiR2 within an ellipse centred at μx = 5 and μy = -5 with major axis as 10 and minor axis as 5.

• Plot all the points using Matplotlib

Code:

```
import matplotlib.pyplot as plt
import numpy as np
from numpy import cos,sin,radians,pi


theta = np.linspace(0,360,90)

for i in range(90):
    x = 5*cos(radians(theta[i])) + 5
    y = 2.5*sin(radians(theta[i])) - 5
    plt.scatter(x,y,color='k',marker='*',s=20)
plt.legend(['Boundary points'],loc='upper right')

for i in range(100):
    a = np.random.uniform(0,5)
    b = np.random.uniform(0,2.5)
```

```
    theta = np.random.uniform(0,2*pi)
    x = a*cos(theta) + 5
    y = b*sin(theta) - 5
    plt.scatter(x,y,color='g',marker='.')

plt.xlabel("X-Axis")
plt.ylabel("Y-Axis")
plt.title("100 Random generated\npoints inside ellipse")

plt.show()
```