**DATA SCEINCE & MACHINE LEARNING**

**LAB**

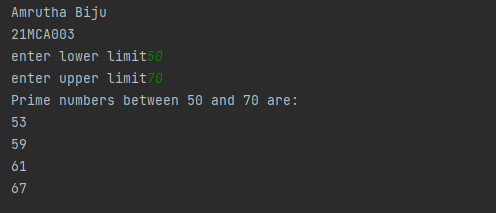
**LAB CYCLE-1**

1. Program to Print all non-Prime Numbers in an Interval

print("Amrutha Biju")  
print("21MCA003")

lower = int(input("enter lower limit"))  
upper = int(input("enter upper limit"))  
  
print("Prime numbers between", lower, "and", upper, "are:")  
  
for num in range(lower, upper + 1):  
  
 if num > 1:  
 for i in range(2, num):  
 if (num % i) == 0:  
 break  
 else:  
 print(num)

**output:**

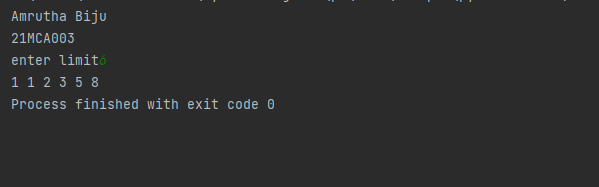


2. Program to print the first N Fibonacci numbers.

print("Amrutha Biju")  
print("21MCA003")

n=int(input("enter limit"))  
f1 = 0  
f2 = 1  
for x in range(0, n):  
 print(f2, end=" ")  
 next = f1 + f2  
 f1 = f2  
 f2 = next

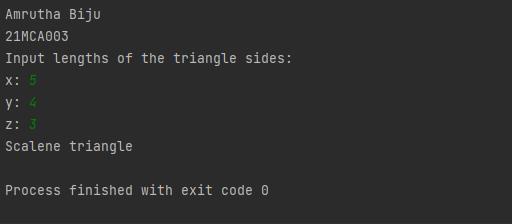
**output:**

****

3. Given sides of a triangle, write a program to check whether given triangle is an isosceles, equilateral or scalene.

print("Amrutha Biju")  
print("21MCA003")  
print("Input lengths of the triangle sides: ")  
x = int(input("x: "))  
y = int(input("y: "))  
z = int(input("z: "))  
  
if x == y == z:  
 print("Equilateral triangle")  
elif x==y or y==z or z==x:  
 print("isosceles triangle")  
else:  
 print("Scalene triangle")

**output:**

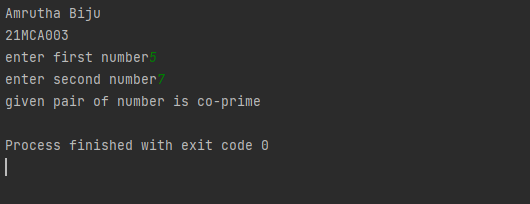


4. Program to check whether given pair of number is coprime

print("Amrutha Biju")  
print("21MCA003")

num1=int(input("enter first number"))  
num2=int(input("enter second number"))  
for i in range(1,num1):  
 if num1%i ==0 and num2%i==0:  
 hcf=i  
if hcf==1:  
 print("given pair of number is co-prime")  
else:  
 print("given pair of number is not co-prime")

**Output:**

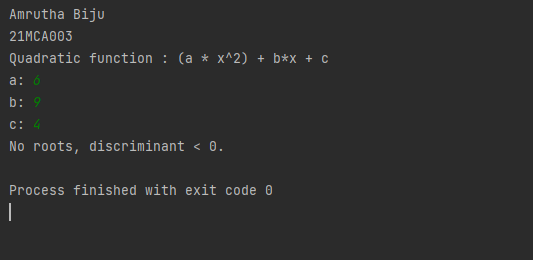
****

5. Program to find the roots of a quadratic equation(rounded to 2 decimal places)

print("Amrutha Biju")  
print("21MCA003")

from math import sqrt  
  
print("Quadratic function : (a \* x^2) + b\*x + c")  
a = float(input("a: "))  
b = float(input("b: "))  
c = float(input("c: "))  
  
r = b \*\* 2 - 4 \* a \* c  
  
if r > 0:  
 num\_roots = 2  
 x1 = (((-b) + sqrt(r)) / (2 \* a))  
 x2 = (((-b) - sqrt(r)) / (2 \* a))  
 print("There are 2 roots: %f and %f" % (x1, x2))  
elif r == 0:  
 num\_roots = 1  
 x = (-b) / 2 \* a  
 print("There is one root: ", x)  
else:  
 num\_roots = 0  
 print("No roots, discriminant < 0.")  
 exit()

**output:**

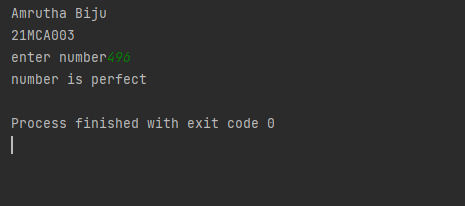
****

6. Program to check whether a given number is perfect number or not(sum of factors=number)

print("Amrutha Biju")  
print("21MCA003")

n=int(input("enter number"))  
sum=0  
for i in range(1,n):  
 if n%i==0:  
 sum=sum+i  
if sum==n:  
 print("number is perfect")  
else:  
 print("number is not perfect")

**output:**

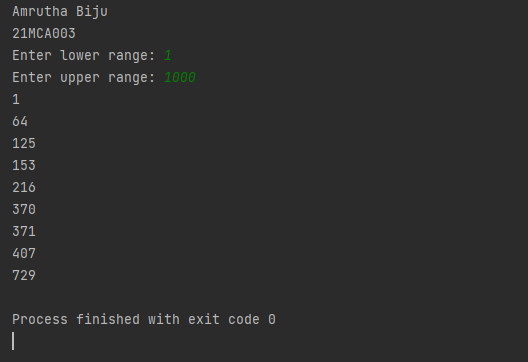
****

7. Program to display amstrong numbers upto 1000

print("Amrutha Biju")  
print("21MCA003")

lower = int(input("Enter lower range: "))  
upper = int(input("Enter upper range: "))  
  
for num in range(lower, upper + 1):  
 sum = 0  
 temp = num  
 while temp > 0:  
 digit = temp % 10  
 sum += digit \*\* 3  
 temp //= 10  
 if num == sum:  
 print(num)

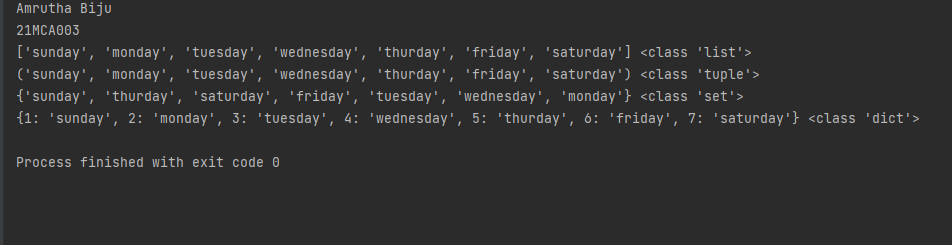
**output:**

****

8. Store and display the days of a week as a List, Tuple, Dictionary, Set. Also demonstrate different ways to store values in each of them. Display its type also.

print("Amrutha Biju")  
print("21MCA003")  
  
list1=["sunday","monday","tuesday","wednesday","thurday","friday","saturday"]  
tuple=("sunday","monday","tuesday","wednesday","thurday","friday","saturday")  
set={"sunday","monday","tuesday","wednesday","thurday","friday","saturday"}  
dict={1:"sunday",2:"monday",3:"tuesday",4:"wednesday",5:"thurday",6:"friday",7:"saturday"}  
print(list1,type(list1))  
print(tuple,type(tuple))  
print(set,type(set))  
print(dict,type(dict))

**output:**

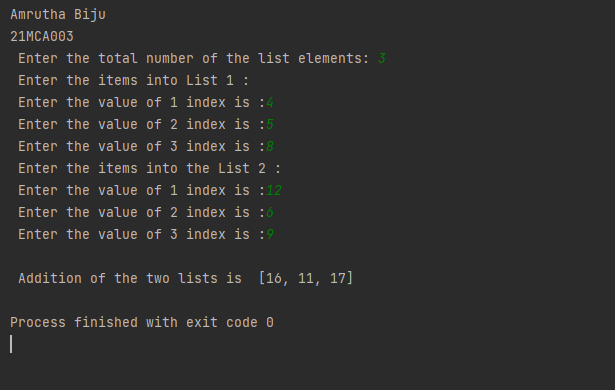


9. Write a program to add elements of given 2 lists

print("Amrutha Biju")  
print("21MCA003")

lt1 = []  
lt2 = []  
lt3 = []  
  
items = int(input(" Enter the total number of the list elements: "))  
  
print(" Enter the items into List 1 : ")  
for i in range(1, items + 1):  
 num = int(input(" Enter the value of %d index is :" % i))  
 lt1.append(num)  
  
print(" Enter the items into the List 2 : ")  
for i in range(1, items + 1):  
 num = int(input(" Enter the value of %d index is :" % i))  
 lt2.append(num)  
  
for j in range(items):  
 lt3.append(lt1[j] + lt2[j])  
print("\n Addition of the two lists is ", lt3)

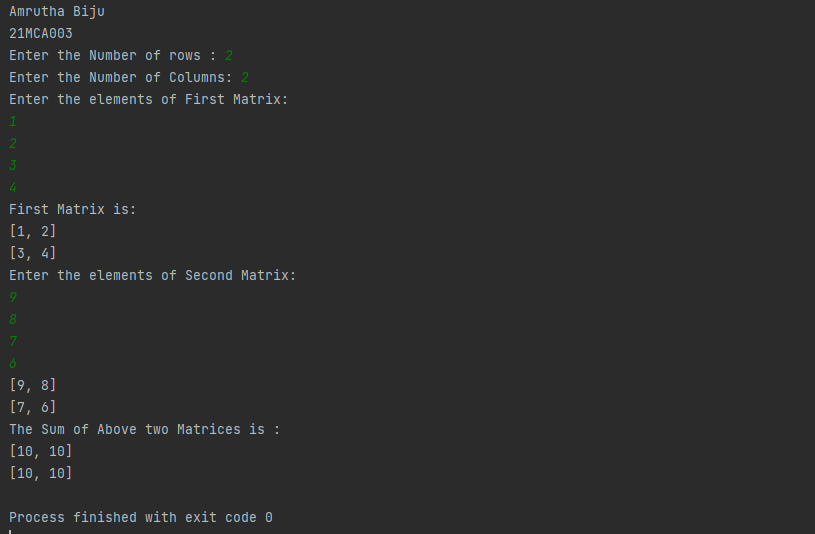
**output:**



10. Write a program to find the sum of 2 matrices using nested List.

print("Amrutha Biju")  
print("21MCA003")  
  
rows = int(input("Enter the Number of rows : "))  
column = int(input("Enter the Number of Columns: "))  
  
print("Enter the elements of First Matrix:")  
matrix\_a = [[int(input()) for i in range(column)] for i in range(rows)]  
print("First Matrix is: ")  
for n in matrix\_a:  
 print(n)  
  
print("Enter the elements of Second Matrix:")  
matrix\_b = [[int(input()) for i in range(column)] for i in range(rows)]  
for n in matrix\_b:  
 print(n)  
  
result = [[0 for i in range(column)] for i in range(rows)]  
  
for i in range(rows):  
 for j in range(column):  
 result[i][j] = matrix\_a[i][j] + matrix\_b[i][j]  
  
print("The Sum of Above two Matrices is : ")  
for r in result:  
 print(r)

**output:**

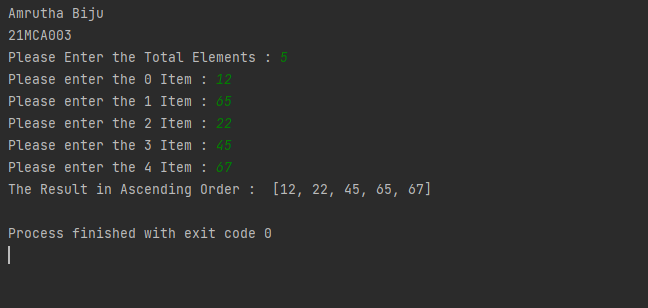


11. Write a program to perform bubble sort on a given set of elements.

print("Amrutha Biju")  
print("21MCA003")

a = []  
number = int(input("Please Enter the Total Elements : "))  
for i in range(number):  
 value = int(input("Please enter the %d Item : " %i))  
 a.append(value)  
  
for i in range(number -1):  
 for j in range(number - i - 1):  
 if(a[j] > a[j + 1]):  
 temp = a[j]  
 a[j] = a[j + 1]  
 a[j + 1] = temp  
  
print("The Result in Ascending Order : ", a)

**output:**

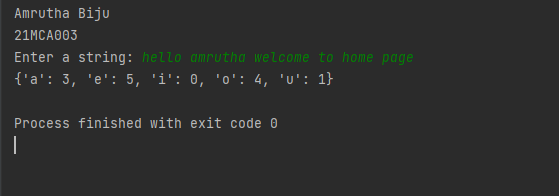


12. Program to find the count of each vowel in a string(use dictionary)

print("Amrutha Biju")  
print("21MCA003")

a = input("Enter a string: ")  
a = a.casefold()  
  
count = {x:sum([1 for char in a if char == x]) for x in 'aeiou'}  
  
print(count)

**output:**



13. Write a Python program that accept a positive number and subtract from this number the sum of its digits and so on. Continues this operation until the number is

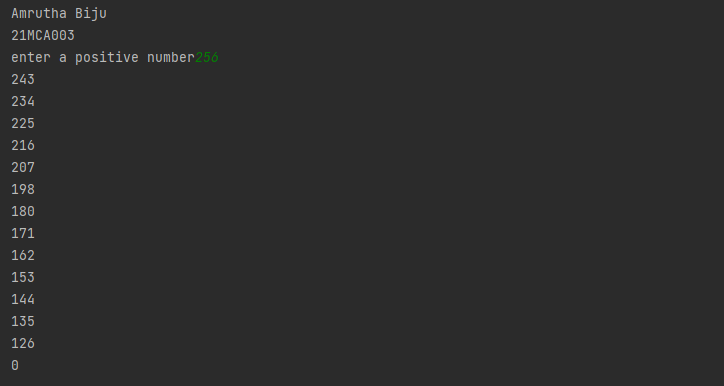
positive(eg: 256-&gt;2+5+6=13

256-13=243

243-9=232……..

print("Amrutha Biju")  
print("21MCA003")  
  
num=int(input("enter a positive number"))  
digsum=0  
new\_num=num  
while new\_num >= digsum:  
 list1 = [int(x) for x in str(new\_num)]  
 for i in list1:  
 digsum=digsum+i  
 new\_num=num-digsum  
  
  
  
 print(new\_num)  
print(new\_num-new\_num)

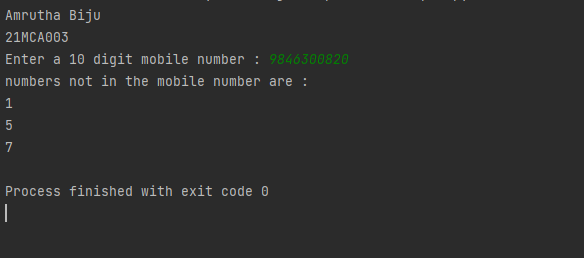
**output:**

****

14. Write a Python program that accepts a 10 digit mobile number, and find the digits which are absent in a given mobile number

num = int(input("Enter a 10 digit mobile number : "))  
nums = []  
for i in range(0, 10):  
 n = num % 10  
 nums.append(n)  
 num = num // 10  
print("numbers not in the mobile number are : ")  
for i in range(0, 10):  
 if i not in nums:  
 print(i)

**Output:**

****

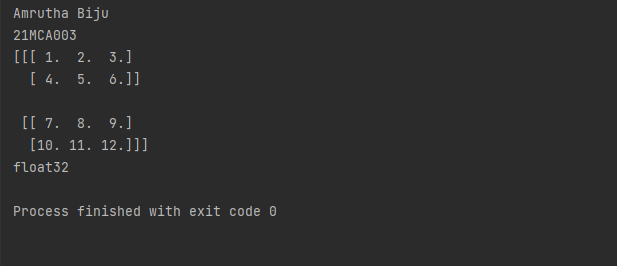
**LAB CYCLE2**

**PART A**

1. Create a three dimensional array specifying float data type and print it.

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
arr=np.array([[[1,2,3],[4,5,6]],[[7,8,9],[10,11,12]]],dtype='f')  
print(arr)  
print(arr.dtype)

**output:**



2. Create a 2 dimensional array (2X3) with elements belonging to complex datatype and print it. Also display

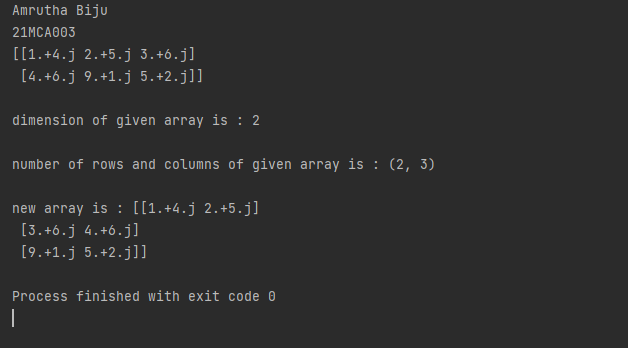
a. the no: of rows and columns

b. dimension of an array

c. reshape the same array to 3X2

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
  
  
arr = np.array([  
  
 [1+4j,2+5j,3+6j],  
  
 [4+6j,9+1j,5+2j],  
  
 ],  
  
 dtype=complex)  
  
print(arr)  
  
print("\ndimension of given array is :",arr.ndim)  
  
print("\nnumber of rows and columns of given array is :",arr.shape)  
  
  
newarr = arr.reshape(3,2)  
  
print("\nnew array is :",newarr)

**output:**

****

3. Familiarize with the functions to create

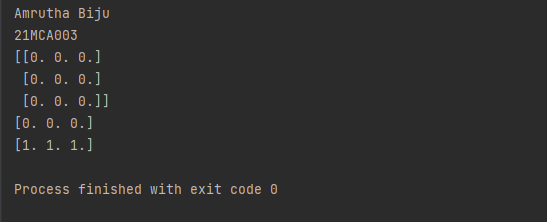
a) an uninitialized array

b) array with all elements as 1,

c) all elements as 0

print("Amrutha Biju")  
print("21MCA003")  
  
  
import numpy as np  
myarr=np.empty((3, 3))  
print(myarr)  
print(np.zeros(3))  
print(np.ones(3))

**output:**



4. Create an one dimensional array using arange function containing 10 elements.

Display

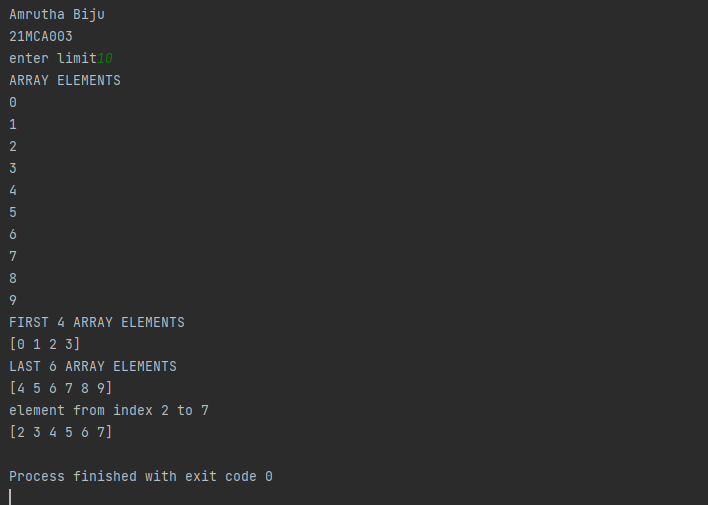
a. First 4 elements

b. Last 6 elements

c. Elements from index 2 to 7

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
n=int(input("enter limit"))  
print("ARRAY ELEMENTS")  
for i in range(n):  
 x=np.arange(n)  
 print(x[i])  
print("FIRST 4 ARRAY ELEMENTS")  
l=np.arange(stop=4)  
print(l)  
print("LAST 6 ARRAY ELEMENTS")  
f=np.arange(start=4,stop=n)  
print(f)  
print("element from index 2 to 7")  
k=x[2:8]  
print(k)

**output:**

****

5.Create an 1D array with arange containing first 15 even numbers as elements

a. Elements from index 2 to 8 with step 2(also demonstrate the same

using slice function)

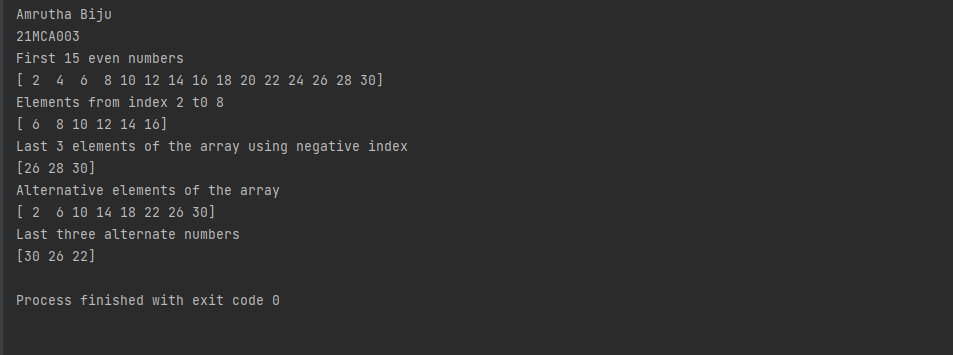
b. Last 3 elements of the array using negative index

c. Alternate elements of the array

d. Display the last 3 alternate elements

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
num=np.arange(2,32,2)  
print("First 15 even numbers")  
print(num)  
s=slice(2,8)  
print("Elements from index 2 t0 8")  
print (num[s])  
index=num[-3:]  
print("Last 3 elements of the array using negative index")  
print(index)  
alternate=num[::2]  
print("Alternative elements of the array")  
print(alternate)  
lalternate=num[15:9:-2]  
print("Last three alternate numbers")  
print(lalternate)

**output:**

****

6. Create a 2 Dimensional array with 4 rows and 4 columns.

a. Display all elements excluding the first row

b. Display all elements excluding the last column

c. Display the elements of 1 st and 2 nd column in 2 nd and 3 rd row

d. Display the elements of 2 nd and 3 rd column

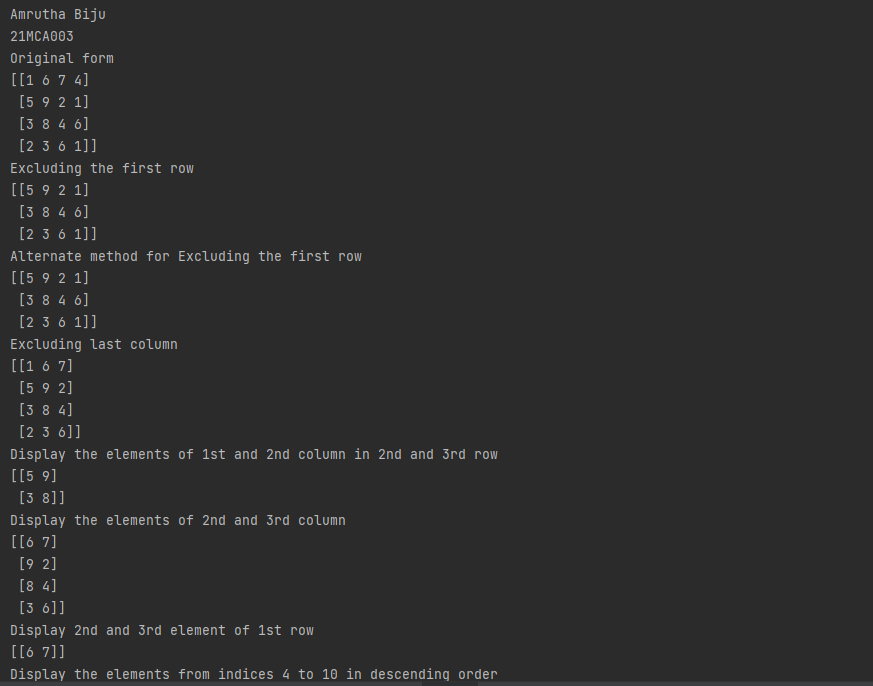
e. Display 2 nd and 3 rd element of 1 st row

f. Display the elements from indices 4 to 10 in descending order(use

–values)

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
X = np.array( [ [ 1, 6, 7, 4],  
[ 5, 9, 2, 1],  
[ 3, 8, 4, 6],  
[ 2, 3, 6, 1] ] )  
print("Original form")  
print(X)  
print("Excluding the first row")  
print(X[1:,])  
print("Alternate method for Excluding the first row")  
num=np.delete(X,0,axis=0)  
print(num)  
print("Excluding last column")  
print(X[:, :-1])  
print("Display the elements of 1st and 2nd column in 2nd and 3rd row")  
print(X[1:3,0:2])  
print("Display the elements of 2nd and 3rd column")  
print(X[:,[1,2]])  
print("Display 2nd and 3rd element of 1st row")  
print(X[0:1,1:3])  
print("Display the elements from indices 4 to 10 in descending order")  
flat\_array=X.flatten()  
print(flat\_array)  
new=sorted(flat\_array[-3:-10])  
index=flat\_array[11:4:-1]  
print(index)

**output**

****

****

7. Create two 2D arrays using array object and

a. Add the 2 matrices and print it

b. Subtract 2 matrices

c. Multiply the individual elements of matrix

d. Divide the elements of the matrices

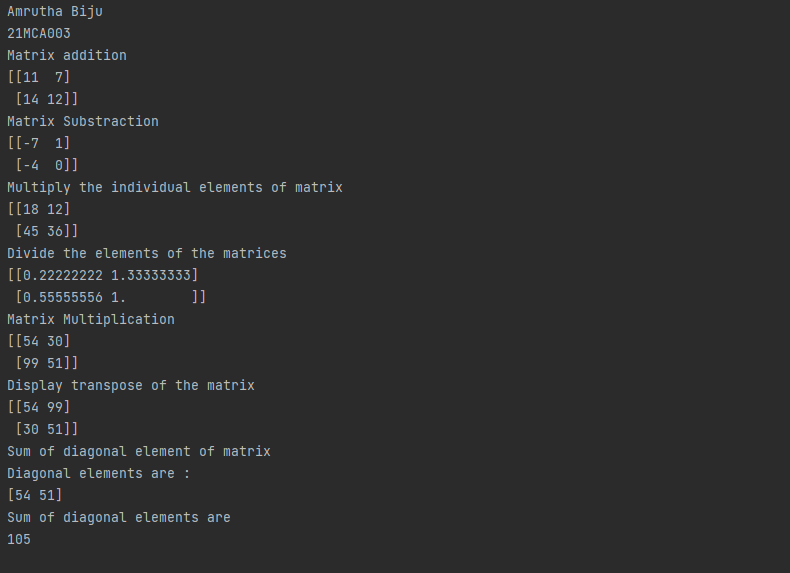
e. Perform matrix multiplication

f. Display transpose of the matrix

g. Sum of diagonal elements of a matrix

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
A=np.array([ [2, 4],[5, 6] ])  
B=np.array([ [9, 3],[9, 6] ])  
print("Matrix addition ")  
C=A+B  
print(C)  
print("Matrix Substraction ")  
C=A-B  
print(C)  
print("Multiply the individual elements of matrix ")  
C=np.multiply(A,B)  
print(C)  
print("Divide the elements of the matrices ")  
C=np.divide(A,B)  
print(C)  
print("Matrix Multiplication" )  
C=np.matmul(A,B)  
print(C)  
print("Display transpose of the matrix ")  
C=np.transpose(C)  
print(C)  
print("Sum of diagonal element of matrix ")  
C=np.diagonal(C)  
print("Diagonal elements are :")  
print(C)  
print("Sum of diagonal elements are ")  
print(sum(C))

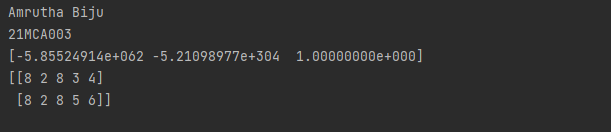
**output**

****

8. Demonstrate the use of insert() function in 1D and 2D array

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
empt\_aray=np.empty([2])  
empt\_aray1=np.array([[2,3,4],[2,5,6]])  
print(np.insert(empt\_aray,2,[1]))  
  
print(np.insert(empt\_aray1,[0,1],8,axis=1))

**output**

****

9. Demonstrate the use of diag() function in 1D and 2D array.

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
arr1=np.array([1,2,3,4,5,6])  
  
arr2=np.array([[1,2,3,4,5,6],[1,2,3,4,5,6],[1,2,3,4,5,6],[1,2,3,4,5,6],[1,2,3,4,5,6]])  
print(arr2)  
print(np.diag(arr2,k=1))

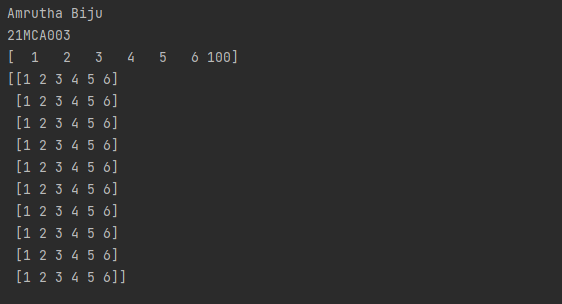
**output**

****

10. Demonstarte the use of append() function in 1D and 2D array.

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
arr1=np.array([1,2,3,4,5,6])  
  
arr2=np.array([[1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6]])  
  
print(np.append(arr1,100))  
  
lst=[20,30,40,50,60,70]  
#print(np.append(arr2,lst))  
print(np.append(arr2,arr2,axis=0))

**output**

****

11. Demonstarte the use of sum() function in 1D and 2D array.

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
arr1=np.array([1,2,3,4,5,6])  
  
arr2=np.array([[1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6],  
 [1,2,3,4,5,6]])  
print("SUM")  
  
print(np.sum(arr2))

**output**

****

**PART B**

1. Create a square matrix with random integer values(use randint()) and use

appropriate functions to find:

i) inverse

ii) rank of matrix

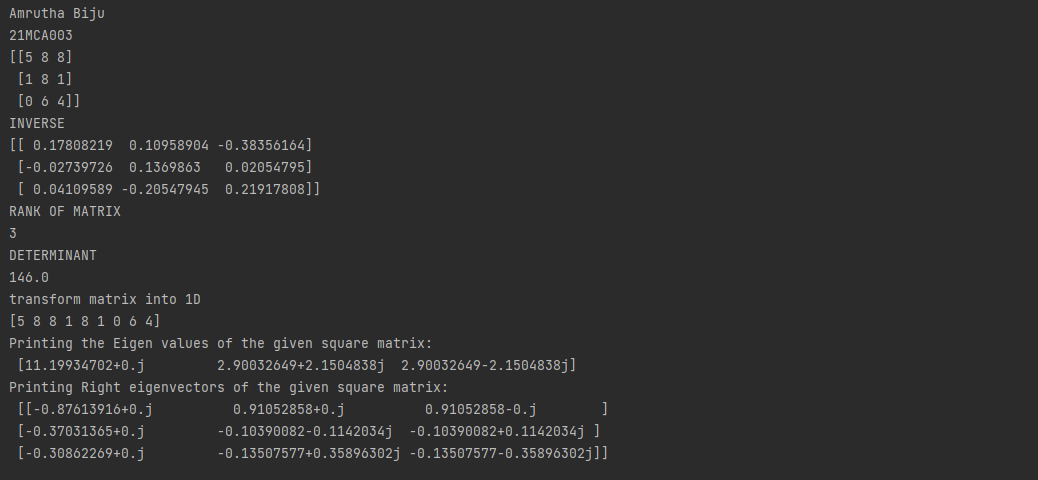
iii) Determinant

iv) transform matrix into 1D array

v) eigen values and vectors

print("Amrutha Biju")  
print("21MCA003")  
  
import numpy as np  
m = np.random.randint(10, size=(3, 3))  
print(m)  
  
print("INVERSE")  
inverse=np.linalg.inv(m)  
print(inverse)  
  
print("RANK OF MATRIX")  
rank = np.linalg.matrix\_rank(m)  
print(rank)  
  
print("DETERMINANT")  
det=np.linalg.det(m)  
print(det)  
  
print("transform matrix into 1D")  
tmatrix = np.ravel(m)  
print(tmatrix)  
  
w, v = np.linalg.eig(m)  
  
  
print("Printing the Eigen values of the given square matrix:\n",w)  
  
  
print("Printing Right eigenvectors of the given square matrix:\n",v)

**output**



2. Create a matrix X with suitable rows and columns

i) Display the cube of each element of the matrix using different methods

(use multiply(), \*, power(),\*\*)

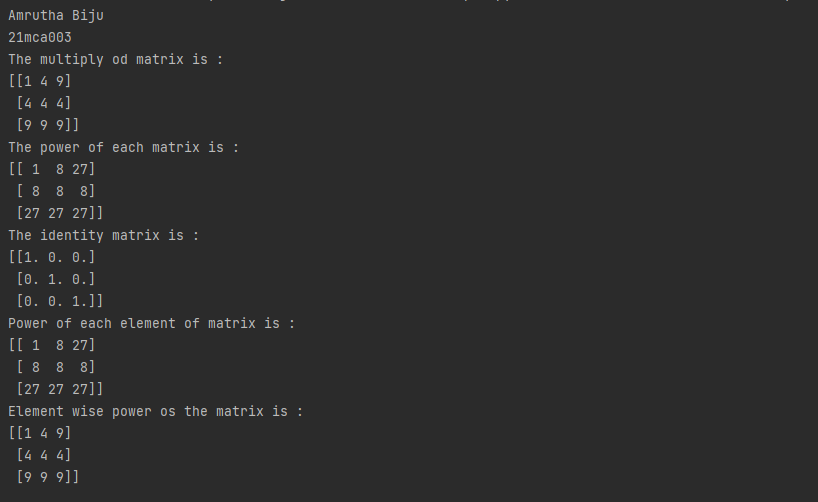
ii) Display identity matrix of the given square matrix.

iii) Display each element of the matrix to different powers.

iv) Create a matrix Y with same dimension as X and perform the operation X 2 +2Y

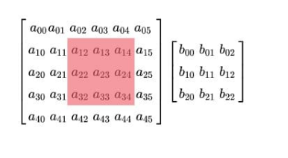
print("Amrutha Biju")  
print("21mca003")  
  
import numpy as np  
  
A = np.array([ [1, 2, 3], [2, 2, 2], [3, 3, 3] ])  
#B = np.array([ [3, 2, 1], [1, 2, 3], [1, 2, 3] ])  
  
arrA = np.multiply(A,A)  
print("The multiply od matrix is :")  
print(arrA)  
  
arrB = np.power(A, 3)  
print("The power of each matrix is :")  
print(arrB)  
  
arrC = np.identity(3)  
print("The identity matrix is :")  
print(arrC)  
  
arrD = np.power(A,3)  
print("Power of each element of matrix is : ")  
print(arrD)  
  
arrE=np.power(A,2)  
print("Element wise power os the matrix is :")  
print(arrE)

**output**



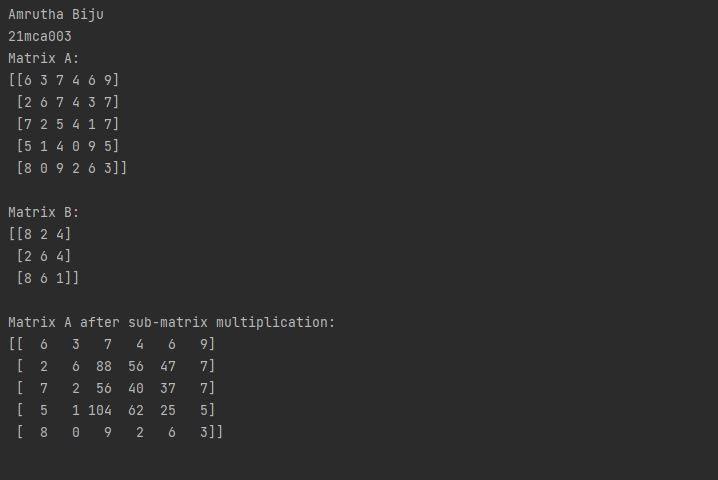
3. Multiply a matrix with a submatrix of another matrix and replace the same in larger

matrix.



print("Amrutha Biju")  
print("21mca003")  
import numpy as np  
np.random.seed(42)  
A = np.random.randint(0, 10, size=(5, 6))  
B = np.random.randint(0, 10, size=(3, 3))  
print("Matrix A:\n{}\n".format(A))  
print("Matrix B:\n{}\n".format(B))  
C = A[1:4, 2:5] @ B  
A[1:4, 2:5] = C  
print("Matrix A after sub-matrix multiplication:\n{}\n".format(A))

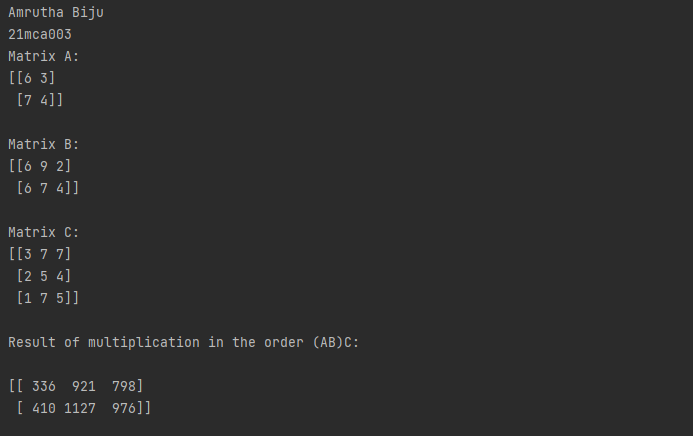
**output**

****

4. Given 3 Matrices A, B and C. Write a program to perform matrix multiplication of the 3 matrices.

print("Amrutha Biju")  
print("21mca003")  
  
import numpy as np  
np.random.seed(42)  
A = np.random.randint(0, 10, size=(2, 2))  
B = np.random.randint(0, 10, size=(2, 3))  
C = np.random.randint(0, 10, size=(3, 3))  
print("Matrix A:\n{}\n".format(A))  
print("Matrix B:\n{}\n".format(B))  
print("Matrix C:\n{}\n".format(C))  
D = np.matmul(np.matmul(A, B), C)  
print("Result of multiplication in the order (AB)C:\n\n{}\n".format(D))

**output**

****

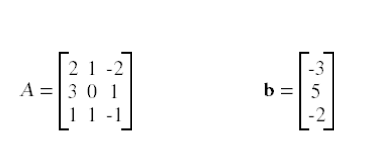
5. Write a program to check whether given matrix is symmetric or Skew Symmetric.

**Solving systems of equations with numpy**

One of the more common problems in linear algebra is solving a matrix-vector equation.

Here is an example. We seek the vector x that solves the equation

A X = b



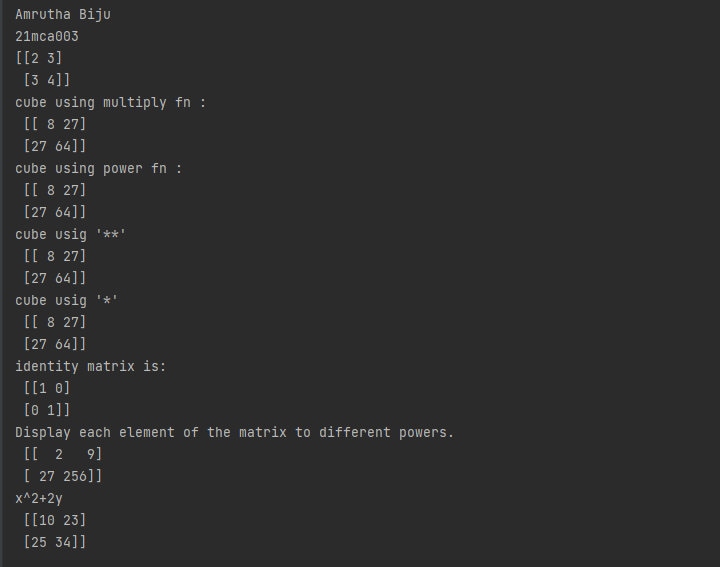
Where

And X=A -1 b.

Numpy provides a function called solve for solving such eauations.

print("Amrutha Biju")  
print("21mca003")  
  
import numpy as np  
x=np.array([[2,3],  
 [3,4]])  
print(x)  
print("cube using multiply fn :\n",np.multiply(x,np.multiply(x,x)))  
print("cube using power fn :\n",np.power(x,3))  
print("cube usig '\*\*'\n",x\*\*3)  
print("cube usig '\*'\n",x\*x\*x)  
print('identity matrix is:\n',np.identity(2,dtype=int))  
print("Display each element of the matrix to different powers.\n",np.power(x,[[1,2],[3,4]]))  
y=np.array([[3,7],  
 [8,9]])  
print("x^2+2y\n",(x\*\*2)+(2\*y))

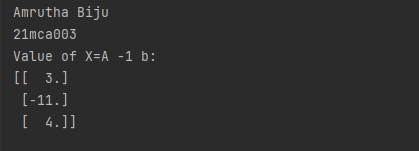
**output**

****

6. Write a program to find out the value of X using solve(), given A and b as above

print("Amrutha Biju")  
print("21mca003")  
import numpy as np  
A = np.array([[2, 1, -2],  
 [3, 0, 1],  
 [1, 1, -1]])  
  
b = np.array([[-3],  
 [5],  
 [-2]])  
a=np.linalg.inv(A)  
x= np.linalg.solve(a, b)  
print("Value of X=A -1 b: ")  
print(x)

**output**

****

7. Write a program to perform the SVD of a given matrix. Also reconstruct the given matrix from the 3 matrices obtained after performing SVD.

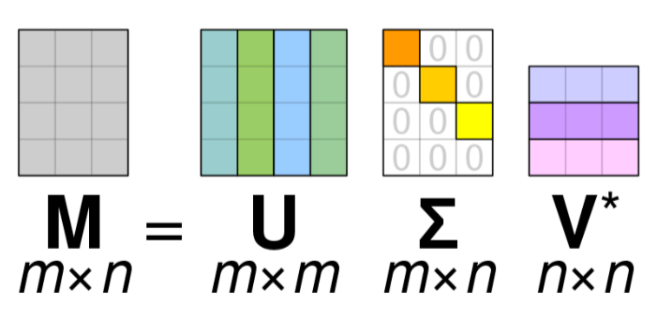
**Singular value Decomposition**

Matrix decomposition, also known as matrix factorization, involves describing a given

matrix using its constituent elements.

The Singular-Value Decomposition, or SVD for short, is a matrix decomposition method for reducing a matrix to its constituent parts in order to make certain subsequent matrix calculations simpler. This approach is commonly used in reducing the no: of attributes in the given data set.

**M= U ∑V^T**

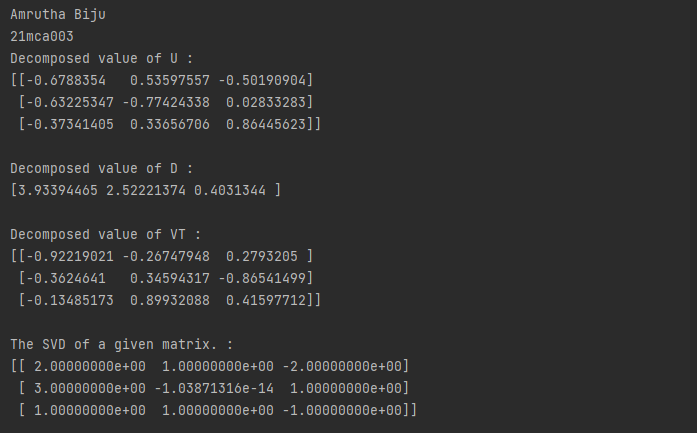


* M-is original matrix we want to decompose
* U-is left singular matrix (columns are left singular vectors). U columns contain eigenvectors of matrix MMᵗ
* Σ-is a diagonal matrix containing singular (eigen) values.
* V-is right singular matrix (columns are right singular vectors). V columns containeigenvectors of matrix MᵗM

Numpy provides a function for performing svd, which decomposes the given matrix into 3 matrices.

print("Amrutha Biju")  
print("21mca003")  
import numpy as np  
A = np.array([[2, 1, -2],  
 [3, 0, 1],  
 [1, 1, -1]])  
  
U, D, VT = np.linalg.svd(A)  
print("Decomposed value of U :")  
print(U)  
print()  
print("Decomposed value of D :")  
print(D)  
print()  
print("Decomposed value of VT :")  
print(VT)  
print()  
  
A\_remake = (U @ np.diag(D) @ VT)  
print("The SVD of a given matrix. :")  
print(A\_remake)

**output**

****

**LAB CYCLE-3**

**Mathplotlib**

**Demonstrate creating various types of charts and plots using functions in mathplotlib library**

1. Sarah bought a new car in 2001 for $24,000. The dollar value of her car changed each year as shown in the table below.

**Value of Sarah's Car**

**Year Value**

**2001 $24,000**

**2002 $22,500**

**2003 $19,700**

**2004 $17,500**

**2005 $14,500**

**2006 $10,000**

**2007 $ 5,800**

Represent the following information using a line graph with following style properties

* **X- axis - Year**

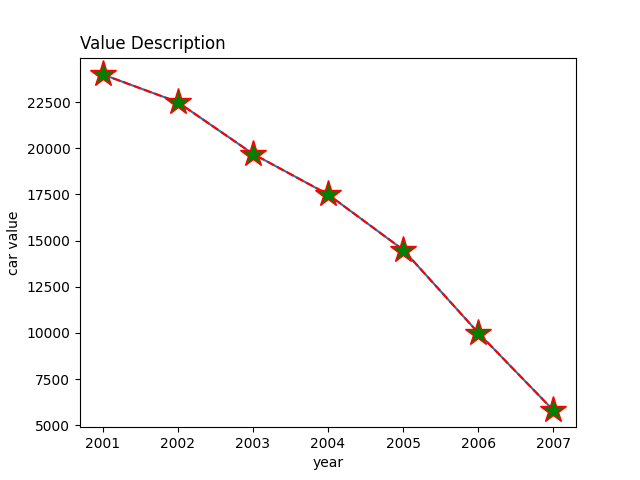
**Y –axis - Car Value**

* **title –Value Depreciation (left Aligned)**
* **Line Style dashdot and Line-color should be red**
* **point using \* symbol with green color and size 20**

Subplot() provides multiple plots in one figure.

print("Amrutha Biju")  
print("21mca003")  
from matplotlib import pyplot as plt  
import numpy as np  
x = np.array([2001,2002,2003,2004,2005,2006,2007])  
y = np.array([24000,22500,19700,17500,14500,10000,5800])  
plt.plot(x,y)  
plt.xlabel("year")  
plt.ylabel("car value")  
plt.title("Value Description",loc='left')  
plt.plot(x, y, linestyle='dashed',color='r',marker='\*',markersize='20',markerfacecolor='green')  
plt.show()

**output**

****

2. Following table gives the daily sales of the following items in a shop

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Day | Mon | Tues | Wed | Thurs | Fri |
| Drinks | 300 | 450 | 150 | 400 | 650 |
| Food | 400 | 500 | 350 | 300 | 500 |

Use subplot function to draw the line graphs with **grids(color as blue**  and line style dotted) for the above information as 2 separate graphs in two rows

1. Properties for the Graph 1:

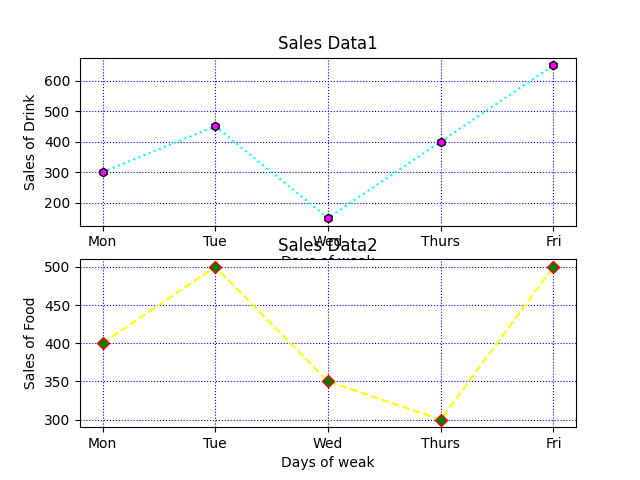
* X label- Days of week
* Y label-Sale of Drinks
* Title-Sales Data1 (right aligned)
* Line –dotted with cyan color
* Points- hexagon shape with color magenta and outline black

1. Properties for the Graph 2:

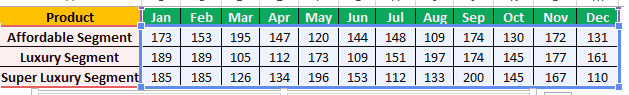
* X label- Days of Week
* Y label-Sale of Food
* Title-Sales Data2 ( center aligned)
* Line –dashed with yellow color
* Points- diamond shape with color green and outline red

print("Amrutha Biju")  
print("21mca003")  
import matplotlib.pyplot as plt  
x = ['Mon','Tue','Wed','Thurs','Fri']  
y = [300,450,150,400,650]  
plt.subplot(2,1,1)  
plt.plot(x,y,linestyle='dotted',color='cyan',marker='h',markerfacecolor='magenta',markeredgecolor='black')  
plt.xlabel('Days of weak')  
plt.ylabel('Sales of Drink')  
plt.title("Sales Data1")  
plt.grid(color='blue',linestyle=':')  
x = ['Mon','Tue','Wed','Thurs','Fri']  
y = [400,500,350,300,500]  
plt.subplot(2,1,2)  
plt.plot(x,y,linestyle='dashed',color='yellow',marker='D',markerfacecolor='green',markeredgecolor='red')  
plt.xlabel('Days of weak')  
plt.ylabel('Sales of Food')  
plt.title("Sales Data2")  
plt.grid(color='blue',linestyle=':')  
plt.show()

**output**

****

3. Create scatter plot for the below data:(use Scatter function)

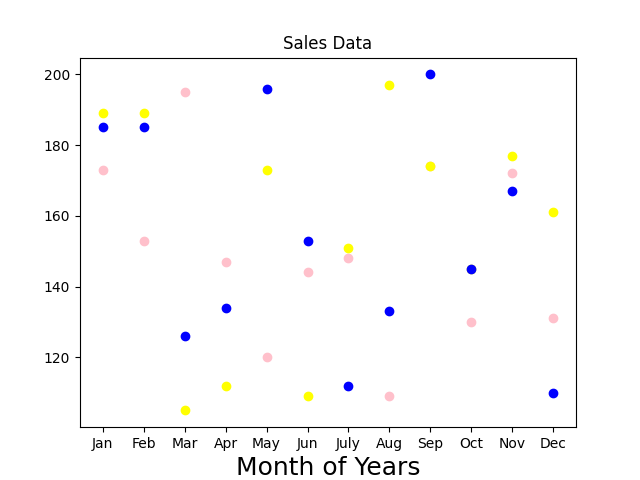


Create scatter plot for each Segment with following properties within one graph

* X Label- Months of Year with font size 18
* Y-Label- Sales of Segments
* Title –Sales Data
* Color for Affordable segment- pink
* Color for Luxury Segment- Yellow
* Color for Super luxury segment-blue

print("Amrutha Biju")  
print("21mca003")  
  
import matplotlib.pyplot as plt  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [173,153,195,147,120,144,148,109,174,130,172,131]  
plt.title('Sales Data')  
plt.xlabel('Month of Years',fontsize=18)  
plt.scatter(x,y,color='pink')  
  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [189,189,105,112,173,109,151,197,174,145,177,161]  
plt.scatter(x,y,color='yellow')  
  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [185,185,126,134,196,153,112,133,200,145,167,110]  
plt.scatter(x,y,color='blue')  
plt.show()

**output**

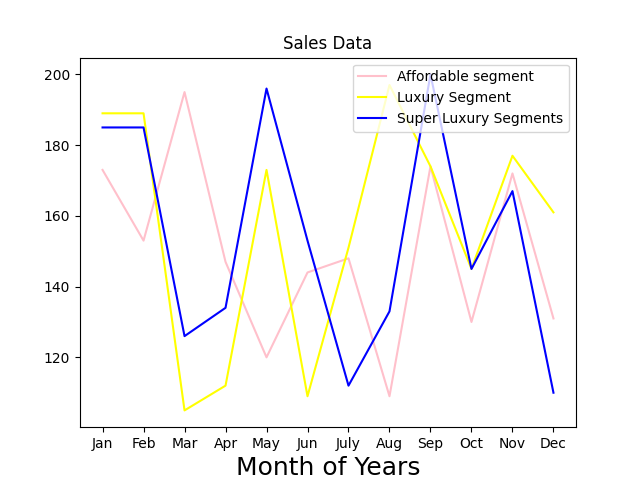
****

4. Display the above data using multiline plot( 3 different lines in same graph)

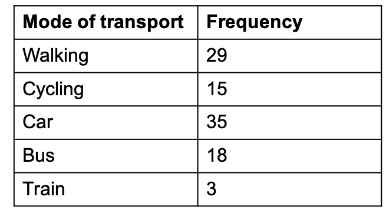
* Display the description of the graph in upper right corner(**use legend())**
* Use different colors and line styles for 3 different lines

print("Amrutha Biju")  
print("21mca003")  
import matplotlib.pyplot as plt  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [173,153,195,147,120,144,148,109,174,130,172,131]  
plt.title('Sales Data')  
plt.xlabel('Month of Years',fontsize=18)  
plt.plot(x,y,color='pink')  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [189,189,105,112,173,109,151,197,174,145,177,161]  
plt.plot(x,y,color='yellow')  
x = ['Jan','Feb','Mar','Apr','May','Jun','July','Aug','Sep','Oct','Nov','Dec']  
y = [185,185,126,134,196,153,112,133,200,145,167,110]  
plt.plot(x,y,color='blue')  
plt.legend(["Affordable segment","Luxury Segment","Super Luxury Segments"],loc='upper right')  
plt.show()

**output**



5. 100 students were asked what their primary mode of transport for getting to school was. The results of this survey are recorded in the table below. Construct a bar graph representing this information.

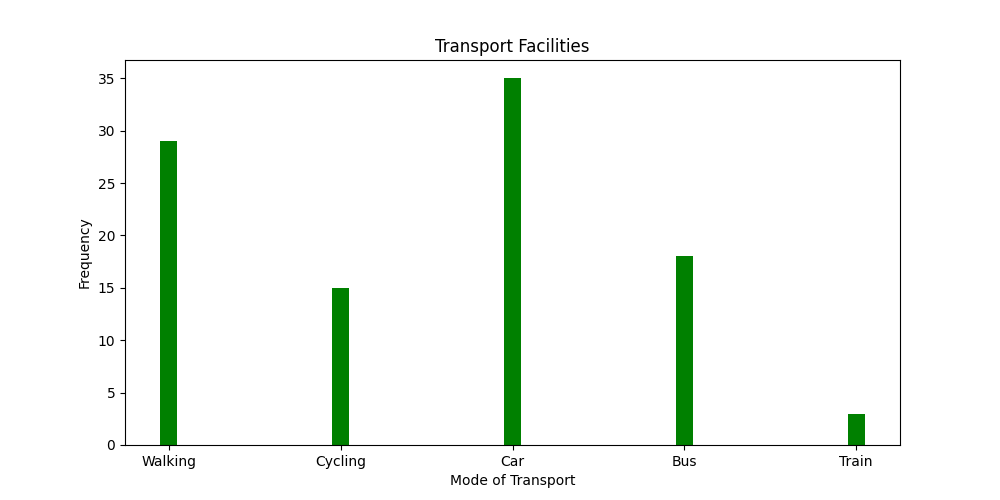


Create a bar graph with

* X axis -mode of Transport and Y axis ‘frequency’
* Provide appropriate labels and title
* Width .1, color  green

print("Amrutha Biju")  
print("21mca003")  
import matplotlib.pyplot as plt  
import numpy as np  
data={'Walking': 29,'Cycling': 15,'Car':35,'Bus':18,'Train':3}  
transport=list(data.keys())  
frequency = list(data.values())  
  
fig = plt.figure(figsize = (10, 5))  
plt.bar(transport, frequency, color ='green', width = 0.1)  
plt.xlabel("Mode of Transport")  
plt.ylabel("Frequency")  
plt.title("Transport Facilities")  
plt.show()

**output**

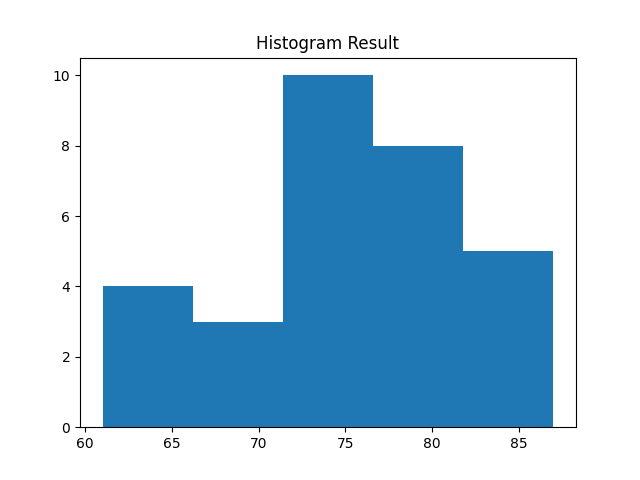
****

6. We are provided with the height of 30 cherry trees.

 The height of the trees (in inches): 61, 63, 64, 66, 68, 69, 71, 71.5, 72, 72.5, 73, 73.5, 74, 74.5, 76, 76.2, 76.5, 77, 77.5, 78, 78.5, 79, 79.2, 80, 81, 82, 83, 84, 85, 87.Create a histogram with a bin size of 5

print("Amrutha Biju")  
print("21mca003")  
import matplotlib.pyplot as plt  
import numpy as np  
fig,ax = plt.subplots(1,1)  
a=np.array([61, 63, 64, 66, 68, 69, 71, 71.5, 72, 72.5, 73, 73.5, 74, 74.5, 76, 76.2,  
76.5, 77, 77.5, 78, 78.5, 79, 79.2, 80, 81, 82, 83, 84, 85, 87.])  
plt.hist(a, bins =5)  
plt.title("Histogram Result")  
plt.show()

**output**

****