INSTITUTE OF SPACE TECHNOLOGY
KICSIT, Kahuta Campus

ARTIFICIAL INTELLIGENCE LAB

Lab Report 02

SUBMITTED BY: HAFIZ MOIZ ALI

SUBMITTED TO: ENGR. ZIA-UR-REHMAN

REGISTRATION NO: 202101009

SEMESTER: BSCE-VII

DEPARTMENT OF COMPUTER ENGINEERING

INSTITUTE OF SPACE TECHNOLOGY

KICSIT, KAHUTA CAMPUS

1. **Lab Task 01**: NumPy is a fundamental tool for AI because it provides the foundation for efficient data handling, mathematical operations, and integration with AI frameworks. Its ability to perform fast, vectorized operations on large datasets makes it indispensable for AI practitioners and researchers.

   Practice the all examples of numpy library mentioned in lab 02 link (**Click here**).

[CLO-01, PLO-02, P-3(Guided Response), Rubric (Coding)]

Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it

NUMPY EXAMPLE

Example 01

Take 2 lists and multiply both you'll see that error occurs repeat the process but by coverting them toarray by numpy.array()

```
l1 = [1,2,3]
l2 = [4,5,6]

l1*l2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Untitled-1.ipynb Cell 4 line 4
      <a href='vscode-notebook-cell:Untitled-1.ipynb?jupyter-notebook#Y233sdW50aXRsZWQ%3D?line=0'>1</a> l1 = [1,2,3]
      <a href='vscode-notebook-cell:Untitled-1.ipynb?jupyter-notebook#Y233sdW50aXRsZWQ%3D?line=1'>2</a> l2 = [4,5,6]
----> <a href='vscode-notebook-cell:Untitled-1.ipynb?jupyter-notebook#Y233sdW50aXRsZWQ%3D?line=3'>4</a> l1*l2

TypeError: can't multiply sequence by non-int of type 'list'
```

```
import numpy as np

l1 = [1,2,3]
l2 = [4,5,6]

A1 = np.array(l1)
A2 = np.array(l2)

print(f"{A1} * {A2} = {A1*A2}")
```

```
[1 2 3] * [4 5 6] = [ 4 10 18]
```

Example 02

Demonstrate the use of numpy.dtype and numpy.shape() functions

```
import numpy as np

l1 = [1,2,3]
l2 = [4,5,6]

A1 = np.array(l1)
A2 = np.array(l2)

A = A1*A2

print(f"{A1} * {A2} = {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")
```

```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (3,)
```

Example 03

The size of an array created with numpy.array() is int32 convert it to int 8

```
import numpy as np

l1 = [1,2,3]
l2 = [4,5,6]

A1 = np.array(l1, np.int8)
A2 = np.array(l2, np.int8)

A = A1*A2

print(f"{A1} * {A2} = {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")
```

```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int8
The dimension of an array: (3,)
```

Example 04

Demonstrate the use of numpy.size() functions

```
import numpy as np

l1 = [1,2,3]
l2 = [4,5,6]

A1 = np.array(l1, np.int8)
A2 = np.array(l2, np.int8)

A = A1*A2

print(f"{A1} * {A2} = {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int8
The dimension of an array: (3,)
```

Example 05

Create a 2D array using numpy.array()

```
import numpy as np

l1 = [1,2,3]
l2 = [4,5,6]

A = np.array((l1, l2))
print(f" The 2D array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The 2D array is :
 [[1 2 3]
 [4 5 6]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (2, 3)
The size of an array: 6
```

Example 06

Create a 1 D array by passing a list

```
A = np.array(([1,2,3,4,5]))
print(f" The 1D array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The 1D array is :
 [1 2 3 4 5]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (5,)
The size of an array: 5
```

Example 07

Create a 2 D array by passing lists

```
import numpy as np

A = np.array(([1,2,3,4,5], [2,3,4,5,6]))
print(f" The 2D array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The 2D array is :
 [[1 2 3 4 5]
 [2 3 4 5 6]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (2, 5)
The size of an array: 10
```

Example 08 Create 4 x 4 Matrix

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [[1 2 3 4]
 [3 6 3 4]
 [1 2 9 4]
 [1 4 5 4]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (4, 4)
The size of an array: 16
```

Example 09 Replace 2nd row 3rd element of above 4x4 matrix with 10

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The original array is : \n {A}")

A[1,2] = 10
print(f" The array after replacing : \n {A}")
```

```
The original array is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The array after replacing :
[[ 1  2  3  4]
[ 3  6 10  4]
[ 1  2  9  4]
[ 1  4  5  4]]
```

Example 10

Create a 5 x 5 matrix of all zeros by setting values of both rows and column

```
import numpy as np

A = np.zeros([5,5])
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (5, 5)
The size of an array: 25
```

Example 11

Create a 5 x 5 matrix of all zeros by passing only 1 argument

```
import numpy as np

A = np.zeros([5])
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [0. 0. 0. 0. 0.]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (5,)
The size of an array: 5
```

Example 12

Create an array from 1 to 100 by numpy.arrange()

```
import numpy as np

A = np.arange(1,100)
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96
 97 98 99]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (99,)
The size of an array: 99
```

Example 13

Create an array from 1 to 100 by numpy.arrange() with a stepsize of 10

```
import numpy as np

A = np.arange(1,100,10)
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [ 1 11 21 31 41 51 61 71 81 91]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (10,)
The size of an array: 10
```

Example 14

Create an array of 100 elements ranging from 2 to 3

```
import numpy as np

A = np.linspace(2,3,100)
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [2.         2.01010101 2.02020202 2.03030303 2.04040404 2.05050505
 2.06060606 2.07070707 2.08080808 2.09090909 2.1010101  2.11111111
 2.12121212 2.13131313 2.14141414 2.15151515 2.16161616 2.17171717
 2.18181818 2.19191919 2.2020202  2.21212121 2.22222222 2.23232323
 2.24242424 2.25252525 2.26262626 2.27272727 2.28282828 2.29292929
 2.3030303  2.31313131 2.32323232 2.33333333 2.34343434 2.35353535
 2.36363636 2.37373737 2.38383838 2.39393939 2.4040404  2.41414141
 2.42424242 2.43434343 2.44444444 2.45454545 2.46464646 2.47474747
 2.48484848 2.49494949 2.50505051 2.51515152 2.52525253 2.53535354
 2.54545455 2.55555556 2.56565657 2.57575758 2.58585859 2.5959596
 2.60606061 2.61616162 2.62626263 2.63636364 2.64646465 2.65656566
 2.66666667 2.67676768 2.68686869 2.6969697  2.70707071 2.71717172
 2.72727273 2.73737374 2.74747475 2.75757576 2.76767677 2.77777778
 2.78787879 2.7979798  2.80808081 2.81818182 2.82828283 2.83838384
 2.84848485 2.85858586 2.86868687 2.87878788 2.88888889 2.8989899
 2.90909091 2.91919192 2.92929293 2.93939394 2.94949495 2.95959596
 2.96969697 2.97979798 2.98989899 3.        ]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (100,)
The size of an array: 100
```

Example 15

Create identity matrix

```python
import numpy as np

A = np.identity(5)
print(f" The array is : \n {A}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The array is :
 [[1. 0. 0. 0. 0.]
 [0. 1. 0. 0. 0.]
 [0. 0. 1. 0. 0.]
 [0. 0. 0. 1. 0.]
 [0. 0. 0. 0. 1.]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (5, 5)
The size of an array: 25
```

Example 16

Create a 4 x 4 matrix and find the sum of all columns

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The Matrix is : \n {A}")

print(f" The row wise sum is : {A.sum(axis=1)}")
print(f" The column wise sum is : {A.sum(axis=0)}")

print(f"The type of array using type: {type(A)}")
print(f"The type of array using dtype: {A.dtype}") # no () with dtype because it is an attribute of A not a function

print(f"The dimension of an array: {A.shape}")

print(f"The size of an array: {A.size}") # The size attrubute counts the total element in the array
```

```
 The Matrix is :
 [[1 2 3 4]
 [3 6 3 4]
 [1 2 9 4]
 [1 4 5 4]]
 The row wise sum is : [10 16 16 14]
 The column wise sum is : [ 6 14 20 16]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (4, 4)
The size of an array: 16
```

Example 17

Find the transpose of a Matrix

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The Matrix is : \n {A}")

print(f" The transpose is : \n {A.T}")
```

```
The Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The transpose is :
[[1 3 1 1]
[2 6 2 4]
[3 3 9 5]
[4 4 4 4]]
```

Example 18

Use reshape command to convrt 4 x 4 matrix to 8 x 2

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The 4x4 Matrix is : \n {A}")

print(f" The 8x2 matrix: \n {A.reshape(8,2)}")
```

```
The 4x4 Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The 8x2 matrix:
[[1 2]
[3 4]
[3 6]
[3 4]
[1 2]
[9 4]
[1 4]
[5 4]]
```

Example 19

Demonstrate the use of numpy.ravel()

```
import numpy as np

r1 = [1,2,3,4]
r2 = [3,6,3,4]
r3 = [1,2,9,4]
r4 = [1,4,5,4]

A = np.array((r1,r2,r3,r4))
print(f" The 4x4 Matrix is : \n {A}")

print(f" The 1D array from above matrix using ravel: \n {A.ravel()}")
```

```
The 4x4 Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The 1D array from above matrix using ravel:
[1 2 3 4 3 6 3 4 1 2 9 4 1 4 5 4]
```

Example 20

Demonstrate the use of argmax, argmin,argsort

```
import numpy as np

a = [1, 16, 31, 4]

A = np.array(a)
print(f"The original array: {A}")

print(f"The index of maximum value in array is: {A.argmax()}")
print(f"The index of minimum value in array is: {A.argmin()}")
print(f"Sorted Indexes: {A.argsort()}")
```

```
The original array: [ 1 16 31  4]
The index of maximum value in array is: 2
The index of minimum value in array is: 0
Sorted Indexes: [0 3 1 2]
```

Example 21

Demostrate the use of numpy.full(),vstack(),hstack(),column_stack(

```
import numpy as np

f1=np.full((2,2),5)
f1
```

```
array([[5, 5],
       [5, 5]])
```

```
f2 = np.full((2,2), 3)
f2
```

```
array([[3, 3],
       [3, 3]])
```

```
a = np.vstack([f1, f2])
a
```

```
array([[5, 5],
       [5, 5],
       [3, 3],
       [3, 3]])
```

```
b = np.hstack([f1, f2])
b
```

```
array([[5, 5, 3, 3],
       [5, 5, 3, 3]])
```

```
a = np.column_stack([f1, f2])
a
```

```
array([[5, 5, 3, 3],
       [5, 5, 3, 3]])
```

Example 22

Save and load a matrix in the memory

```
import numpy as np

a = np.full((2,3), 5)
a
```

```
array([[5, 5, 5],
       [5, 5, 5]])
```

```
np.save("untitled.npy", a)
```

```
savedMatrix = np.load('untitled.npy')
savedMatrix
```

```
array([[5, 5, 5],
       [5, 5, 5]])
```

Example 23

Demonstrate the use of numoy.dot() and compare it with simple multiplication

```
import numpy as np

f1=np. full((2,2),5)
print("\nf1 = \n",f1)

f2=np.full((2,2), 3)
print("\nf2 = \n", f2)

print("point to point multiplication = ",f1*f2)

print("point to point multiplication = ", np.dot(f1,f2))
```

```
f1 =
 [[5 5]
 [5 5]]

f2 =
 [[3 3]
 [3 3]]
point to point multiplication =  [[15 15]
 [15 15]]
point to point multiplication =  [[30 30]
 [30 30]]
```

1. **Lab Task 02**: Pandas plays a pivotal role in AI by facilitating data preparation, exploration, and transformation, which are essential steps in the machine learning pipeline. It empowers data scientists and AI practitioners to efficiently work with structured data and prepare it for training and evaluation of AI models.
   Try to implement all the examples of panads library mentioned in lab 02 link (**Click here**).

Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it

[CLO-01, PLO-02, P-3(Guided Response), Rubric (Coding)]

02. Pandas

Example 01

Create a Dictionary and convert them into data frames also check its datatype

```
#create a dictionary

StuDict={"Name": ["Aqsa","Esha", "Ayesha", "Ayra", "Arfa", "Afsa", "Abdul", "Saadia", "Abu Bakar","Atif"],
"ID": ["SID-1","SID-2", "SID-3", "SID-4", "SID-5","SID-6", "SID-7", "SID-8", "SID-9","SID-10"],
"Rol1_no": [1,2,3,4,5,6,7,8,9,10],
"Semester" : [7,7,7,7,6,6,6,5,8,8]}

StuDict
```

```
{'Name': ['Aqsa',
  'Esha',
  'Ayesha',
  'Ayra',
  'Arfa',
  'Afsa',
  'Abdul',
  'Saadia',
  'Abu Bakar',
  'Atif'],
 'ID': ['SID-1',
  'SID-2',
  'SID-3',
  'SID-4',
  'SID-5',
  'SID-6',
  'SID-7',
  'SID-8',
  'SID-9',
  'SID-10'],
 'Rol1_no': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 'Semester': [7, 7, 7, 7, 6, 6, 6, 5, 8, 8]}
```

```python
#convert into data frames

import pandas as pd

data=pd.DataFrame (StuDict)

print(data)

print("\n\nThe data type of above given syntax is :",type (data))
```

```
        Name      ID  Rol1_no  Semester
0       Aqsa   SID-1        1         7
1       Esha   SID-2        2         7
2     Ayesha   SID-3        3         7
3       Ayra   SID-4        4         7
4       Arfa   SID-5        5         6
5       Afsa   SID-6        6         6
6      Abdul   SID-7        7         6
7     Saadia   SID-8        8         5
8  Abu Bakar   SID-9        9         8
9       Atif  SID-10       10         8


The data type of above given syntax is : <class 'pandas.core.frame.DataFrame'>
```

Example 02

Demonstrate the use of describe function for a data frame

```
print(data.describe())
```

```
         Rol1_no   Semester
count  10.00000  10.000000
mean    5.50000   6.700000
std     3.02765   0.948683
min     1.00000   5.000000
25%     3.25000   6.000000
50%     5.50000   7.000000
75%     7.75000   7.000000
max    10.00000   8.000000
```

Example 03

Demonstrate the use of head function for a data frame

```
print(data.head())
```

```
      Name     ID  Rol1_no  Semester
0     Aqsa  SID-1        1         7
1     Esha  SID-2        2         7
2   Ayesha  SID-3        3         7
3     Ayra  SID-4        4         7
4     Arfa  SID-5        5         6
```

Example 03

Demonstrate the use of head function for a data frame

```
print(data.head())
```

```
     Name     ID  Rol1_no  Semester
0    Aqsa  SID-1        1         7
1    Esha  SID-2        2         7
2  Ayesha  SID-3        3         7
3    Ayra  SID-4        4         7
4    Arfa  SID-5        5         6
```

Example 04

Demonstrate the use of tail function for a data frame

```
print(data.tail())
```

```
         Name      ID  Rol1_no  Semester
5        Afsa   SID-6        6         6
6       Abdul   SID-7        7         6
7      Saadia   SID-8        8         5
8  Abu Bakar    SID-9        9         8
9        Atif  SID-10       10         8
```

Example 05

Demonstrate the use of info function for a data frame

```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Name      10 non-null     object
 1   ID        10 non-null     object
 2   Rol1_no   10 non-null     int64
 3   Semester  10 non-null     int64
dtypes: int64(2), object(2)
memory usage: 452.0+ bytes
None
```

Example 06 Convert the data frame in a variable to CSV file

```
data.to_csv('student.csv')
```

Example 07 Remove the indexes from the csv file

```
data.to_csv('Without_index.csv', index=False)
```

Example 08 Read from csv file

```
df = pd.read_csv('student.csv')
df
```

Example 09 Use describe,head,tail and info function for CSV file

```
import pandas as pd
df = pd.read_csv('student.csv')

print(f"Describe Function \n {df.describe()}, \n head Function \n {df.head()} \n tail Function \n {df.tail()}")
print(f"\n info Function \n {df.info()}")
```

```
Describe Function
        Unnamed: 0   Rol1_no   Semester
count    10.00000  10.00000  10.000000
mean      4.50000   5.50000   6.700000
std       3.02765   3.02765   0.948683
min       0.00000   1.00000   5.000000
25%       2.25000   3.25000   6.000000
50%       4.50000   5.50000   7.000000
75%       6.75000   7.75000   7.000000
max       9.00000  10.00000   8.000000,
 head Function
    Unnamed: 0    Name     ID  Rol1_no  Semester
0           0    Aqsa  SID-1        1         7
1           1    Esha  SID-2        2         7
2           2  Ayesha  SID-3        3         7
3           3    Ayra  SID-4        4         7
4           4    Arfa  SID-5        5         6
 tail Function
```

```
    Unnamed: 0       Name      ID  Rol1_no  Semester
5            5       Afsa   SID-6        6         6
6            6      Abdul   SID-7        7         6
7            7     Saadia   SID-8        8         5
8            8  Abu Bakar   SID-9        9         8
9            9       Atif  SID-10       10         8
<class 'pandas.core.frame.DataFrame'>
...
memory usage: 532.0+ bytes

 info Function
 None
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Example 10 Access a column by its name

```
import pandas as pd

df['Name']
```

```
0         Aqsa
1         Esha
2       Ayesha
3         Ayra
4         Arfa
5         Afsa
6        Abdul
7       Saadia
8    Abu Bakar
9         Atif
Name: Name, dtype: object
```

Example 11 Access the 1st element of a column

```
df['Name'][8]
```

```
'Abu Bakar'
```

Example 12 Update the value in the column

```
df['Name'][0] = 'Saddam'
df
```

```
C:\Users\Falcon\AppData\Local\Temp\ipykernel_6952\2832195598.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df['Name'][0] = 'Saddam'
```

Example 13 Find the columns and indexes in a data frame

```
df.columns
```

```
Index(['Unnamed: 0', 'Name', 'ID', 'Rol1_no', 'Semester'], dtype='object')
```

```
df.index
```

```
RangeIndex(start=0, stop=10, step=1)
```

Example 14 Create a series of 50 random numbers and check their data type and shape

```
import pandas as pd
import numpy as np

s = pd.Series(np.random.rand(50))
print(s)
print(f"Using dtype: {s.dtype}")
print(f"Using type: {type(s)}")
print(f"Using Shape: {s.shape}")
```

```
0     0.145202
1     0.534662
2     0.741749
3     0.940588
4     0.702558
5     0.125774
6     0.524639
7     0.764519
8     0.907276
9     0.780484
10    0.779356
11    0.500806
12    0.257662
13    0.487086
14    0.771487
15    0.927485
16    0.975931
17    0.629163
18    0.823954
19    0.911563
20    0.377728
21    0.927874
22    0.175054
23    0.974362
24    0.700084
...

dtype: float64
Using dtype: float64
Using type: <class 'pandas.core.series.Series'>
Using Shape: (50,)
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Example 15 Create a 50 x 5 data set from random values

```
import pandas as pd
import numpy as np

dataf = pd.DataFrame(np.random.rand(50,5))
print(s)
```

```
0     0.145202
1     0.534662
2     0.741749
3     0.940588
4     0.702558
5     0.125774
6     0.524639
7     0.764519
8     0.907276
9     0.780484
10    0.779356
11    0.500806
12    0.257662
13    0.487086
14    0.771487
15    0.927485
16    0.975931
17    0.629163
18    0.823954
19    0.911563
```

```
20    0.377728
21    0.927874
22    0.175054
23    0.974362
24    0.700084
...
47    0.271523
48    0.472256
49    0.009996
dtype: float64
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Example 16 Find the minimum maximum and mean values column wise in a dataset

```
dataf.min()
```

```
0    0.007086
1    0.026261
2    0.001513
3    0.026857
4    0.016388
dtype: float64
```

```
dataf.max()
```

```
0    0.951014
1    0.957882
2    0.942145
3    0.998660
4    0.989068
dtype: float64
```

```
dataf.mean()
```

```
0    0.465671
1    0.456569
2    0.421697
3    0.483558
4    0.467064
dtype: float64
```

Example 17 Find the maximum value in 1st column

```
dataf[0].max()
```

```
0.95101418352417 09
```

Example 18 Convert the dataset into numpy array and also take transpose of it

```
d1 = dataf.to_numpy()
d1
```

```
array([[0.81800156, 0.69912332, 0.92050335, 0.36977564, 0.89211492],
       [0.74694264, 0.8126709 , 0.28531898, 0.52826528, 0.43001568],
       [0.19282411, 0.17201277, 0.11814032, 0.77631757, 0.6097806 ],
       [0.50123085, 0.95788219, 0.20960535, 0.59528145, 0.02901539],
       [0.72615245, 0.06424098, 0.58470371, 0.4077009 , 0.8506021 ],
       [0.06774963, 0.30263532, 0.37451239, 0.52556451, 0.27990389],
       [0.57619813, 0.671159  , 0.82340189, 0.91809502, 0.98906819],
       [0.10349276, 0.13058516, 0.03075786, 0.7673907 , 0.02678578],
       [0.67102865, 0.30594025, 0.28270142, 0.31635002, 0.69697101],
       [0.95101418, 0.67923939, 0.8465084 , 0.99865967, 0.06836426],
       [0.59946978, 0.50903266, 0.8673587 , 0.07870138, 0.72227662],
       [0.43570644, 0.86648114, 0.77064959, 0.50190638, 0.59935719],
       [0.19431767, 0.33445761, 0.01452628, 0.26073319, 0.77728355],
       [0.66970924, 0.66501526, 0.80907623, 0.34761996, 0.67867881],
       [0.18037832, 0.15006953, 0.03545436, 0.57721446, 0.04863471],
       [0.08685797, 0.31377045, 0.19154818, 0.49756929, 0.89814456],
       [0.11299151, 0.15314305, 0.46261696, 0.71575654, 0.69350094],
       [0.63449355, 0.3994742 , 0.14740068, 0.02882023, 0.10386001],
       [0.68073509, 0.55874066, 0.24505871, 0.98246422, 0.59958276],
       [0.10566315, 0.77487853, 0.85757252, 0.86782849, 0.77655113],
       [0.4985468 , 0.04816482, 0.04899402, 0.54065651, 0.82241182],
       [0.81639725, 0.40550348, 0.03573331, 0.60615679, 0.33095124],
       [0.94411487, 0.81386241, 0.36921181, 0.10882449, 0.34717876],
```

```
       [0.20447347, 0.02626056, 0.65854862, 0.10642769, 0.62785798],
       [0.4363815 , 0.49055904, 0.00151341, 0.36858459, 0.35183703],
...
       [0.52100455, 0.34751345, 0.54591251, 0.8497092 , 0.31126664],
       [0.86991469, 0.11778735, 0.65651015, 0.23096171, 0.27415629],
       [0.7222734 , 0.27963102, 0.05717662, 0.17739786, 0.51686661],
       [0.01453424, 0.78863946, 0.00320828, 0.0436443 , 0.15493464],
       [0.00708563, 0.07328581, 0.4836743 , 0.75429741, 0.90829987]])
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

```
d1.T
```

```
array([[0.81800156, 0.74694264, 0.19282411, 0.50123085, 0.72615245,
        0.06774963, 0.57619813, 0.10349276, 0.67102865, 0.95101418,
        0.59946978, 0.43570644, 0.19431767, 0.66970924, 0.18037832,
        0.08685797, 0.11299151, 0.63449355, 0.68073509, 0.10566315,
        0.4985468 , 0.81639725, 0.94411487, 0.20447347, 0.4363815 ,
        0.38044229, 0.55870586, 0.72663449, 0.57197626, 0.22045933,
        0.52145851, 0.87601895, 0.0200121 , 0.83909097, 0.4484107 ,
        0.14693213, 0.52737705, 0.59416786, 0.02829982, 0.73152434,
        0.15814559, 0.43482893, 0.03905682, 0.65178111, 0.71855285,
        0.52100455, 0.86991469, 0.7222734 , 0.01453424, 0.00708563],
       [0.69912332, 0.8126709 , 0.17201277, 0.95788219, 0.06424098,
        0.30263532, 0.671159  , 0.13058516, 0.30594025, 0.67923939,
        0.50903266, 0.86648114, 0.33445761, 0.66501526, 0.15006953,
        0.31377045, 0.15314305, 0.3994742 , 0.55874066, 0.77487853,

         0.04816482, 0.40550348, 0.81386241, 0.02626056, 0.49055904,
         0.29513477, 0.4537389 , 0.53176879, 0.45422995, 0.144078  ,
         0.34360905, 0.8127101 , 0.3492097 , 0.58918087, 0.42764047,
         0.57491346, 0.63598116, 0.36190463, 0.45138456, 0.2475908 ,
         0.777563  , 0.6565783 , 0.0383573 , 0.87728474, 0.89382332,
         0.34751345, 0.11778735, 0.27963102, 0.78863946, 0.07328581],
        [0.92050335, 0.28531898, 0.11814032, 0.20960535, 0.58470371,
         0.37451239, 0.82340189, 0.03075786, 0.28270142, 0.8465084 ,
         0.8673587 , 0.77064959, 0.01452628, 0.80907623, 0.03545436,
         0.19154818, 0.46261696, 0.14740068, 0.24505871, 0.85757252,
         0.04899402, 0.03573331, 0.36921181, 0.65854862, 0.00151341,
...
         0.95120441, 0.1868687 , 0.38930084, 0.29920254, 0.04109224,
         0.32520873, 0.04658173, 0.59136578, 0.16812414, 0.45955717,
         0.55971428, 0.79491285, 0.75359838, 0.29972298, 0.91491346,
         0.17142039, 0.01638761, 0.0318168 , 0.54307484, 0.39286793,
         0.31126664, 0.27415629, 0.51686661, 0.15493464, 0.90829987]])
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Example 19 Change names of the columns.

```
dataf.columns = ['A', 'B', 'C', 'D', 'E']
dataf
```

Example 20 Display column B and C from the dataset and also use head function

```
dataf[['B', 'C']]
```

```
dataf.head()
```

Example 21 Demonstrate the use of iloc function

```
dataf.iloc[:, 0:2]   # : means all rows and 0:2 means cloumns till 2
```

Example 22 Print column A to C and fimd the value on 0,0

```
dataf.loc[:, 'A':'C'] # loc function use to specify the columns label or name
```

Example 23 Print 1st 12 elements of column 2 and 4

```
dataf.iloc[0:12, 2:4]
```

1. **Lab Task 03**: Matplotlib plays a vital role in AI by providing a versatile toolkit for

data visualization, model evaluation, debugging, and presenting results. Its ability to create a wide range of plots and its integration with other AI-related libraries make it a valuable tool for AI practitioners and researchers. **Try** to implement all the examples of matplotlib library mentioned in lab 02 ➐ link (**Click here**).

[CLO-01, PLO-02, P-3(Guided Response), Rubric (Coding)]

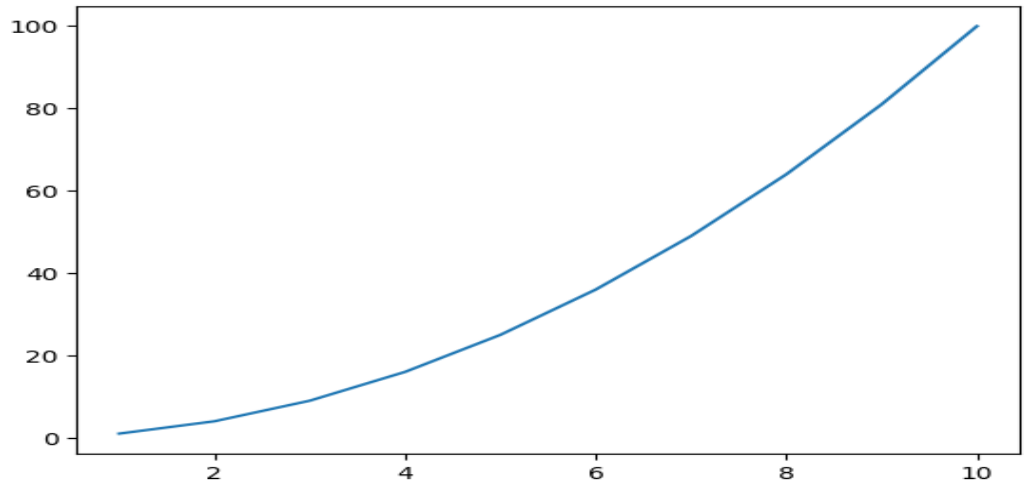
Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it

03. Matplotlib

Example 01 Use plot and show function to create and show the graph

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array((1,2,3,4,5,6,7,8,9,10))
y = x**2

print(x)

plt.plot(x,y)
plt.show()
```

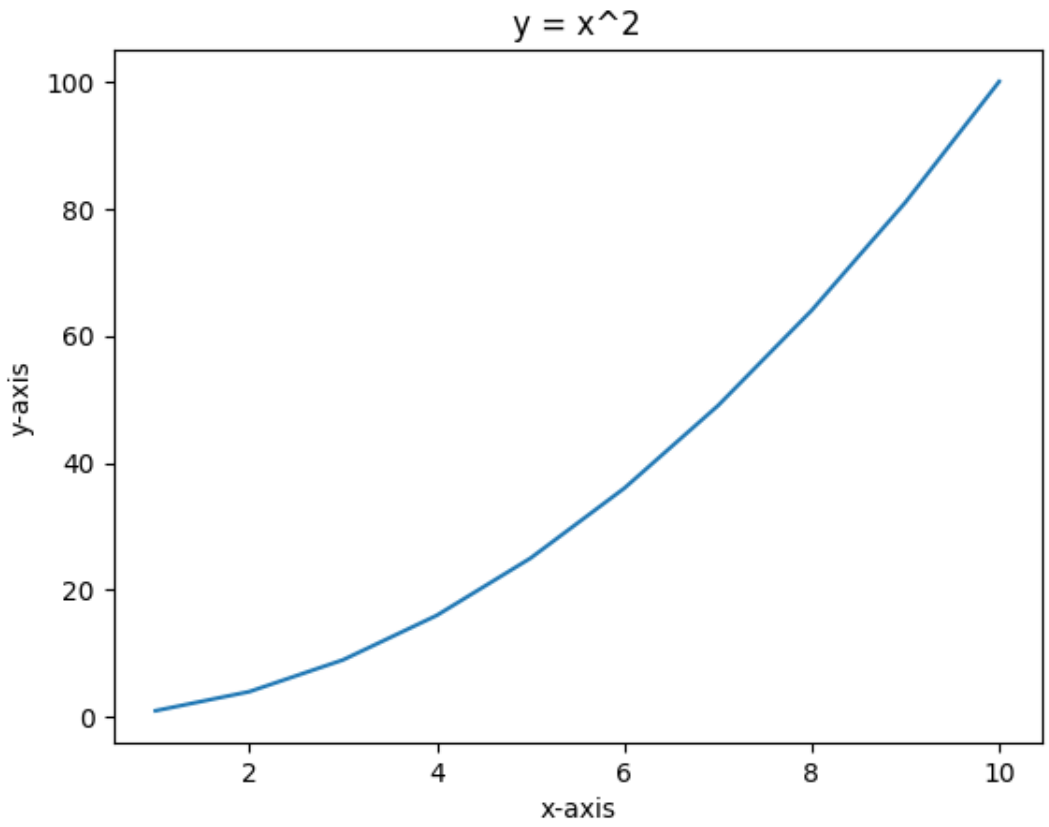
```
[ 1  2  3  4  5  6  7  8  9 10]
```



Example 02 Add labels and tittle to the graph

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array((1,2,3,4,5,6,7,8,9,10))
y = x**2

print(x)

plt.plot(x,y)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title("y = x^2")
plt.show()
```

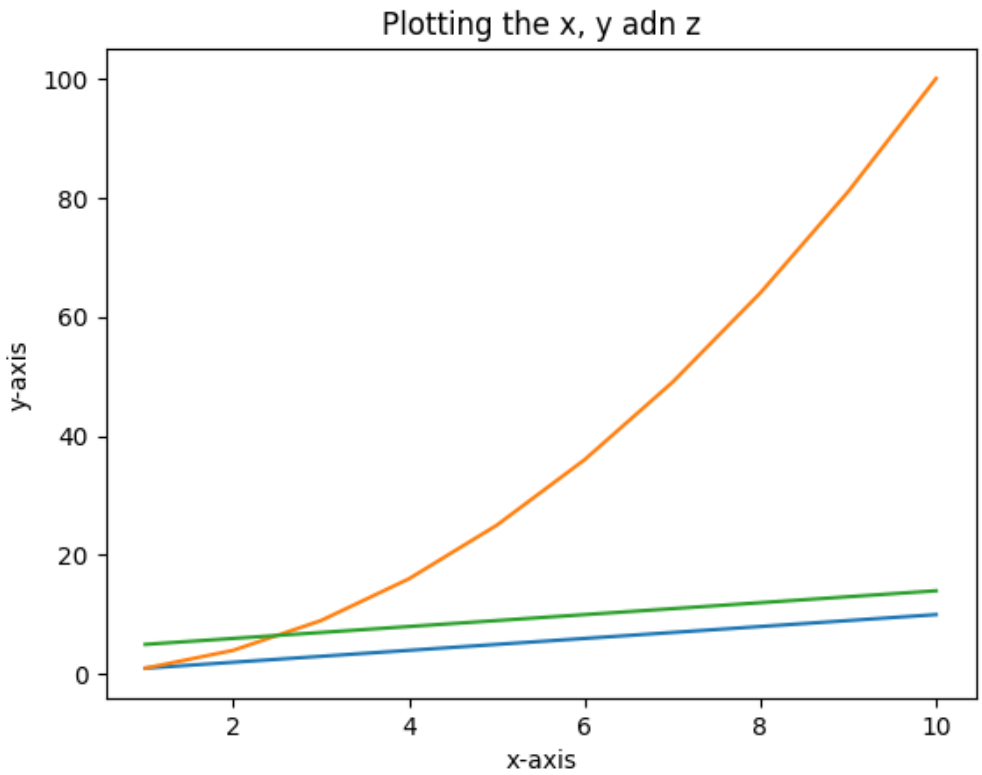
```
[ 1  2  3  4  5  6  7  8  9 10]
```



Exampel 03 Plot 3 variables on a single graph

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array((1,2,3,4,5,6,7,8,9,10))
y = x**2
z = x + 4

print(f"{x}\n{y}\n{z}")

plt.plot(x,x)
plt.plot(x,y)
plt.plot(x,z)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title("Plotting the x, y adn z")
plt.show()
```

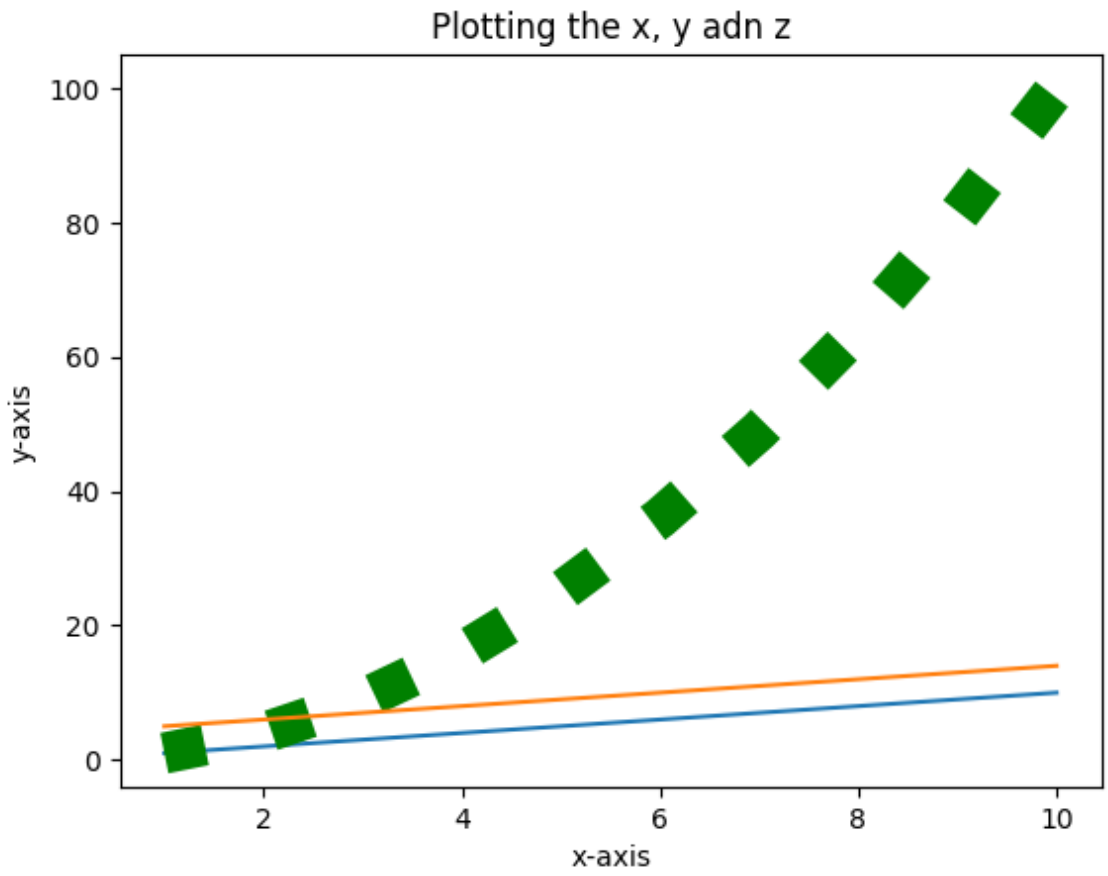
```
[ 1  2  3  4  5  6  7  8  9 10]
[  1   4   9  16  25  36  49  64  81 100]
[ 5  6  7  8  9 10 11 12 13 14]
```



Example 04 Change the color linestyle and linewidth of the graph

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array((1,2,3,4,5,6,7,8,9,10))
y = x**2
z = x + 4

print(f"{x}\n{y}\n{z}")

plt.plot(x,x)
plt.plot(x,y, color='g', linestyle = ':', linewidth=15)
plt.plot(x,z)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title("Plotting the x, y adn z")
plt.show()
```

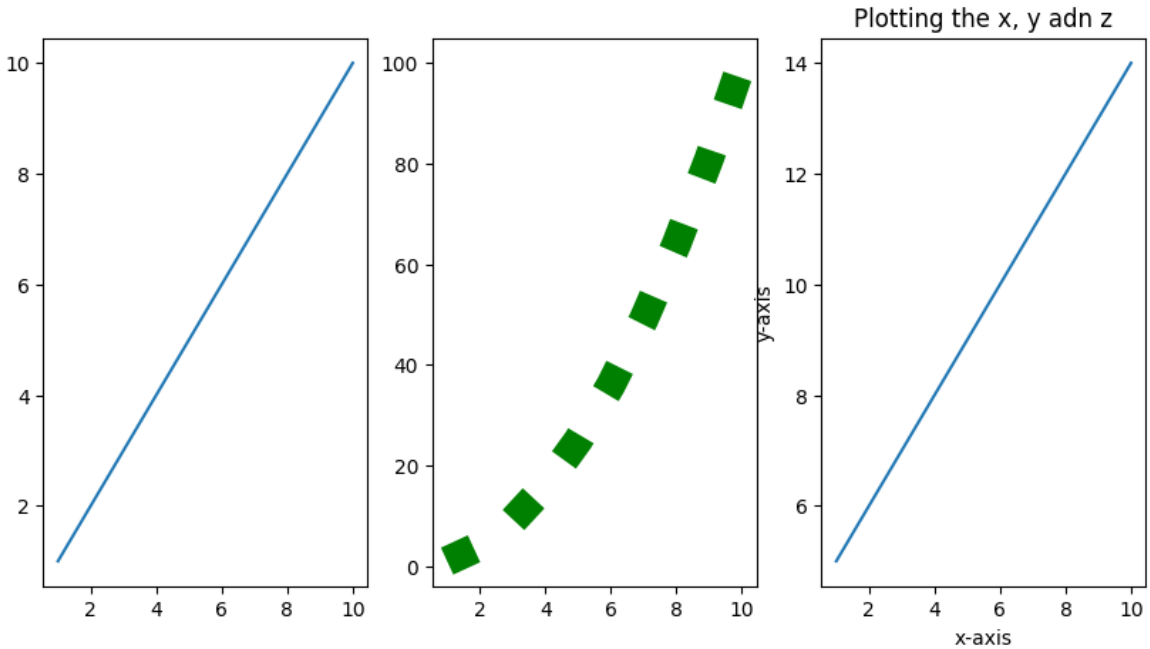
```
[ 1  2  3  4  5  6  7  8  9 10]
[  1   4   9  16  25  36  49  64  81 100]
[ 5  6  7  8  9 10 11 12 13 14]
```



Example 05 Plot using subplot

```
import numpy as np
import matplotlib.pyplot as plt

x = np.array((1,2,3,4,5,6,7,8,9,10))
y = x**2
z = x + 4

print(f"{x}\n{y}\n{z}")

plt.figure(figsize=(10,5))

plt.subplot(1,3,1)
plt.plot(x,x)

plt.subplot(1,3,2)
plt.plot(x,y, color='g', linestyle = ':', linewidth=15)

plt.subplot(1,3,3)
plt.plot(x,z)
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title("Plotting the x, y adn z")
plt.show()
```

```
[ 1  2  3  4  5  6  7  8  9 10]
[  1   4   9  16  25  36  49  64  81 100]
```

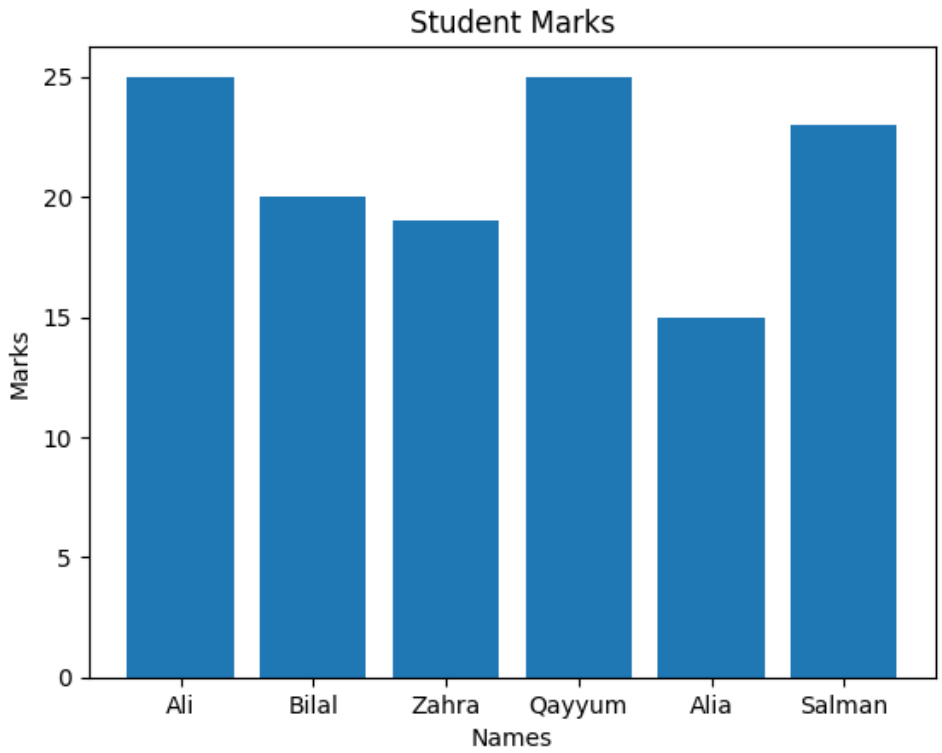


Example 06 Print the marks of students w.r.t their names using Dictionary

```
stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

k = stuMarks.keys()
v = stuMarks.values()

plt.title("Student Marks")
plt.xlabel("Names")
plt.ylabel("Marks")
plt.bar(k,v)
plt.show()
```

```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```

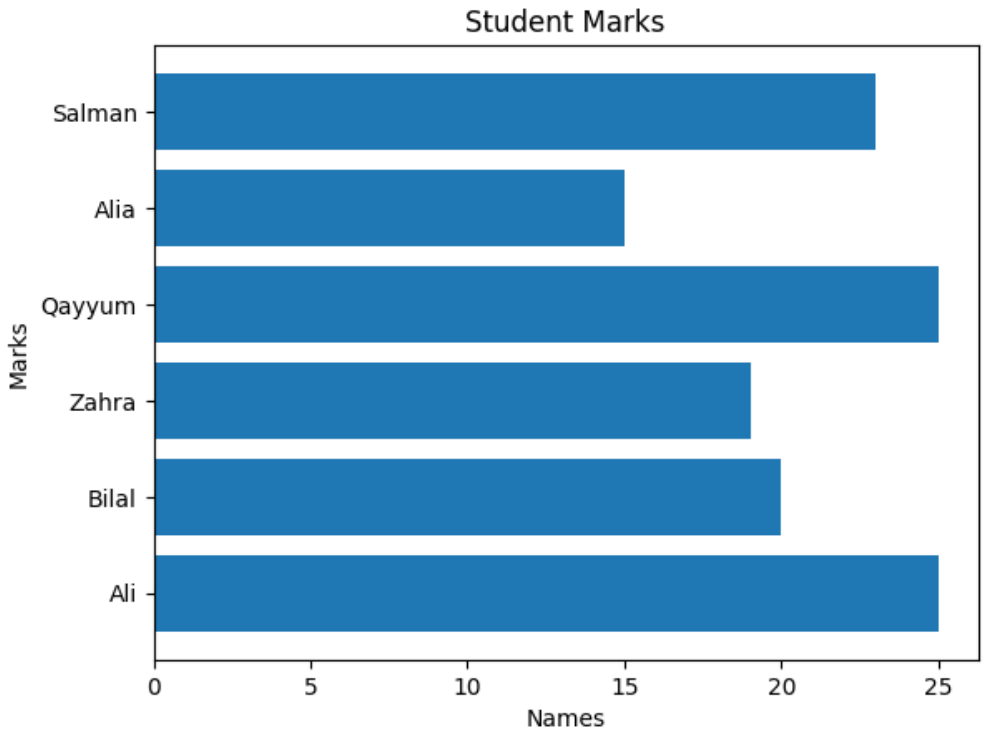

```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```

Example 07 Plot horizonatal bar garaph

```
stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

k = list(stuMarks.keys())
v = list(stuMarks.values())

plt.title("Student Marks")
plt.xlabel("Names")
plt.ylabel("Marks")
plt.barh(k,v)
plt.show()
```

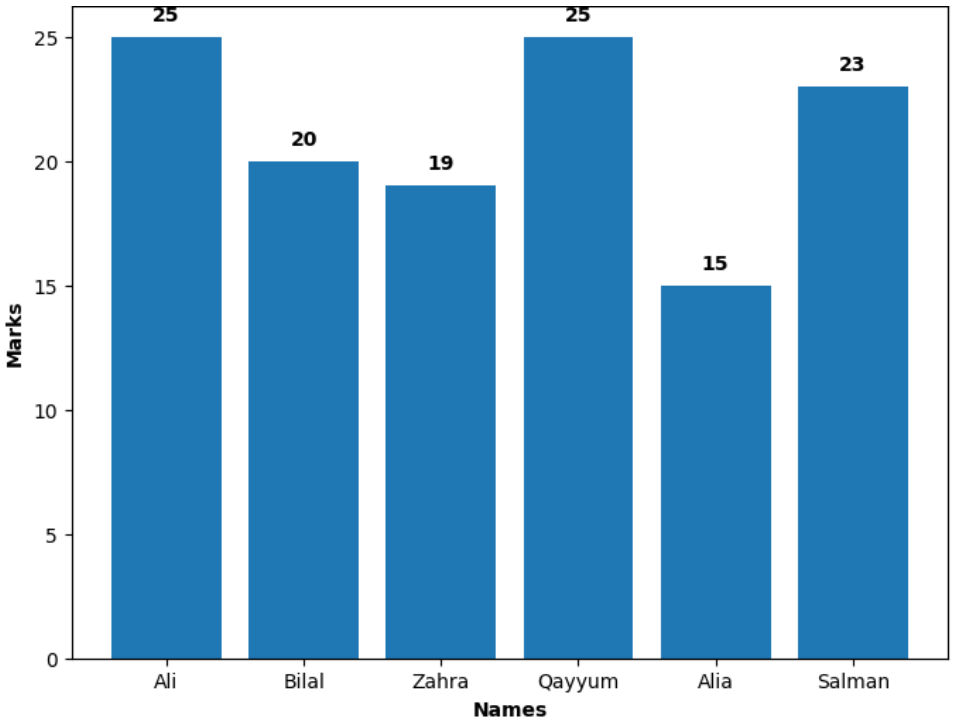
```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```



Example 08 Bold the tittle and xlabel y label and also show the value of yaxis on top of bars

```
import matplotlib.pyplot as plt

stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

k = stuMarks.keys()
v = stuMarks.values()

plt.figure(figsize=(8, 6))  # Set the figure size

# Plot the bar chart
plt.bar(k, v)

# Customize the plot
plt.title("Student Marks", fontweight="bold")  # Make the title bold
plt.xlabel("Names", fontweight="bold")  # Make xlabel bold
plt.ylabel("Marks", fontweight="bold")  # Make ylabel bold

# Annotate the values on top of the bars
for key, value in stuMarks.items():
    plt.text(key, value + 0.5, str(value), ha='center', va='bottom', fontweight='bold')

plt.show()
```

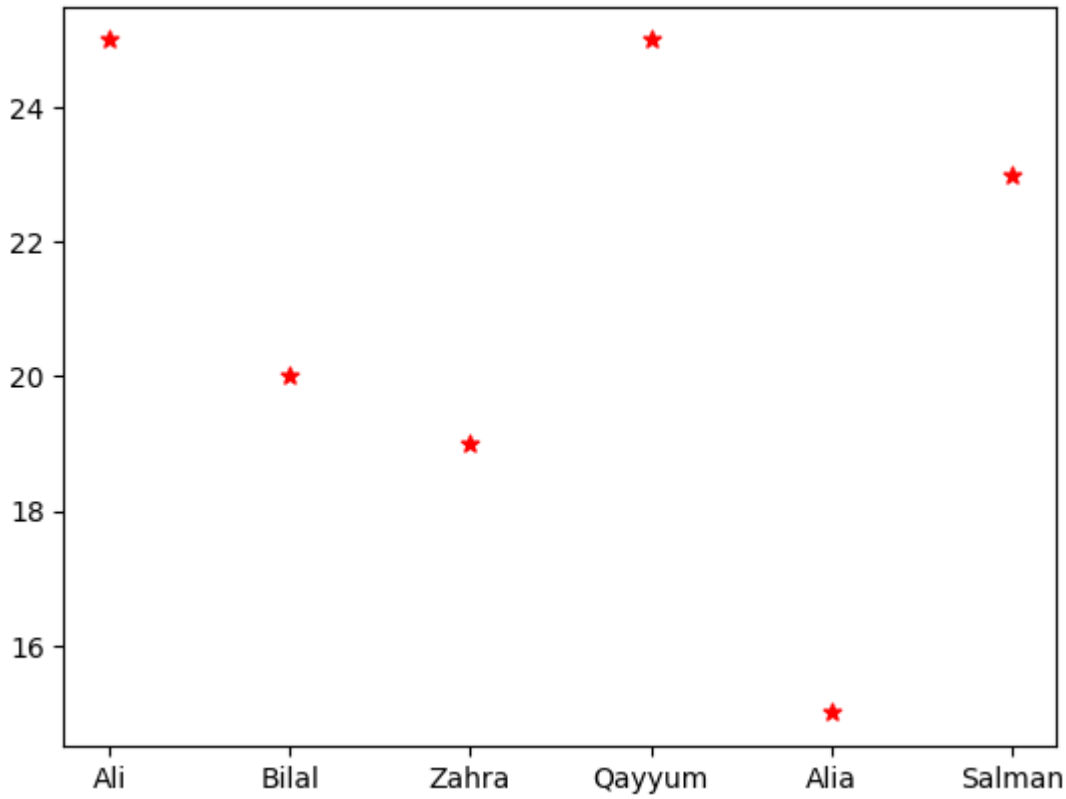
```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```



Example 09 Plot using scatter function

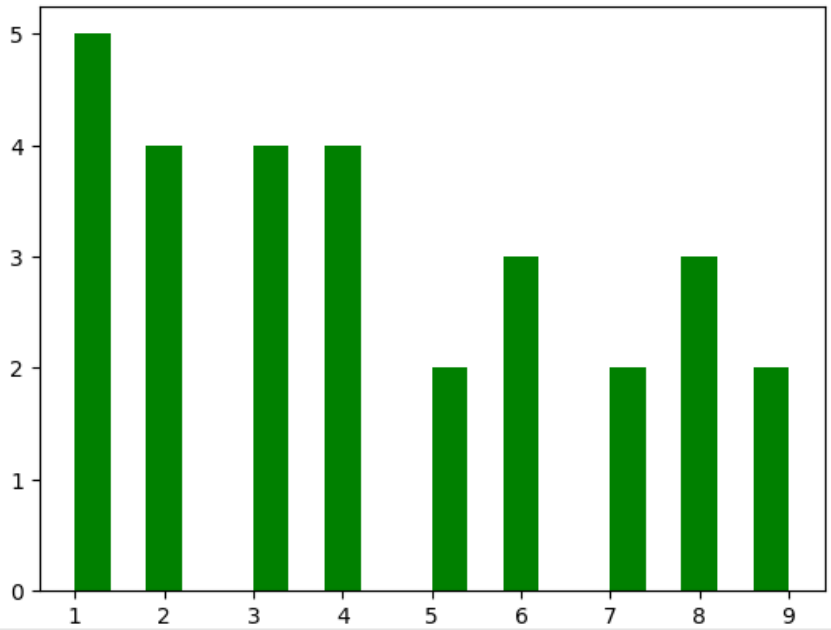
```
import matplotlib.pyplot as plt

stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

k = stuMarks.keys()
v = stuMarks.values()

plt.scatter(k,v, color = 'r', marker="*", s = 40)
plt.show()
```

```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```



Example 10 Plot a histogram

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

plt.hist(a, bins=20, color='g')
plt.show()
```

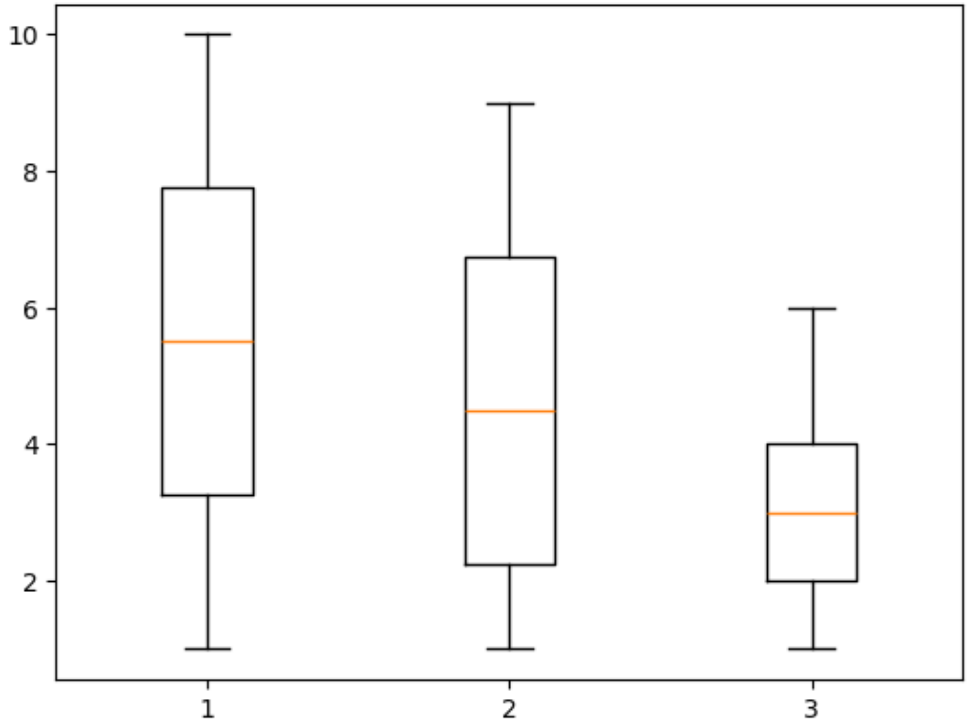


Example 11 Demonstrate the use of Box Plot

A Box Plot is also known as Whisker plot is created to display the summary of the set of data values having properties like minimum, first quartile, median, third quartile and maximum. In the box plot, a box is created from the first quartile to the third quartile, a vertical line is also there which goes through the box at the median. Here x-axis denotes the data to be plotted while the y-axis shows the frequency distribution.

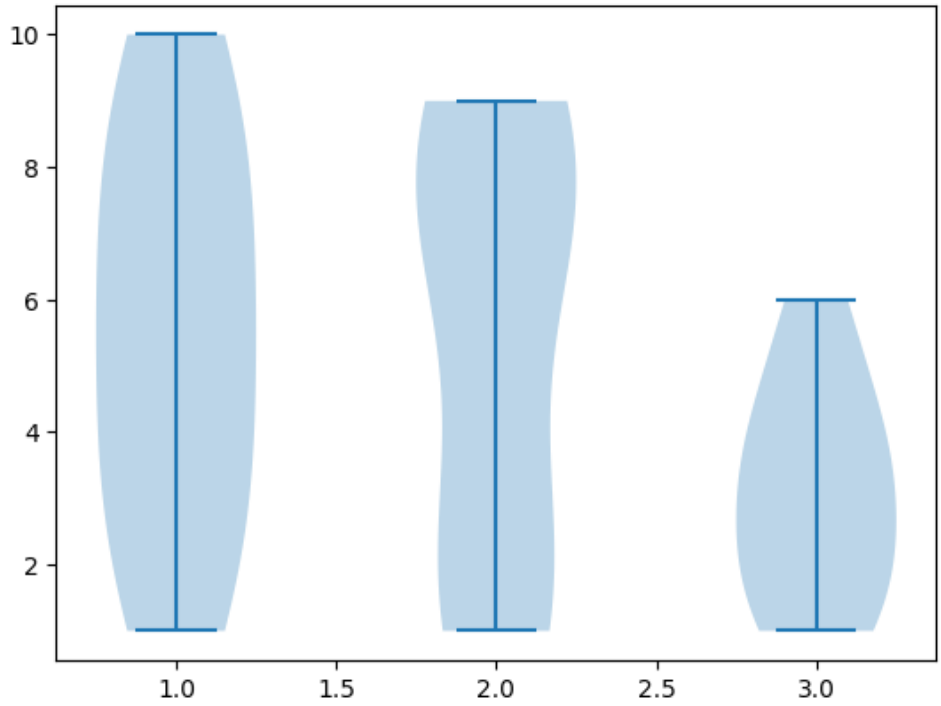
```
l1 = [1,2,3,4,5,6,7,8,9,10]
l2 = [3,4,5,6,7,1,2,8,9,1]
l3 = [1,2,3,4,1,2,3,4,5,6]

data = list([l1,l2,l3])

plt.boxplot(data)
plt.show()
```

Python



Example 12 Demonstrate the use of violin plot

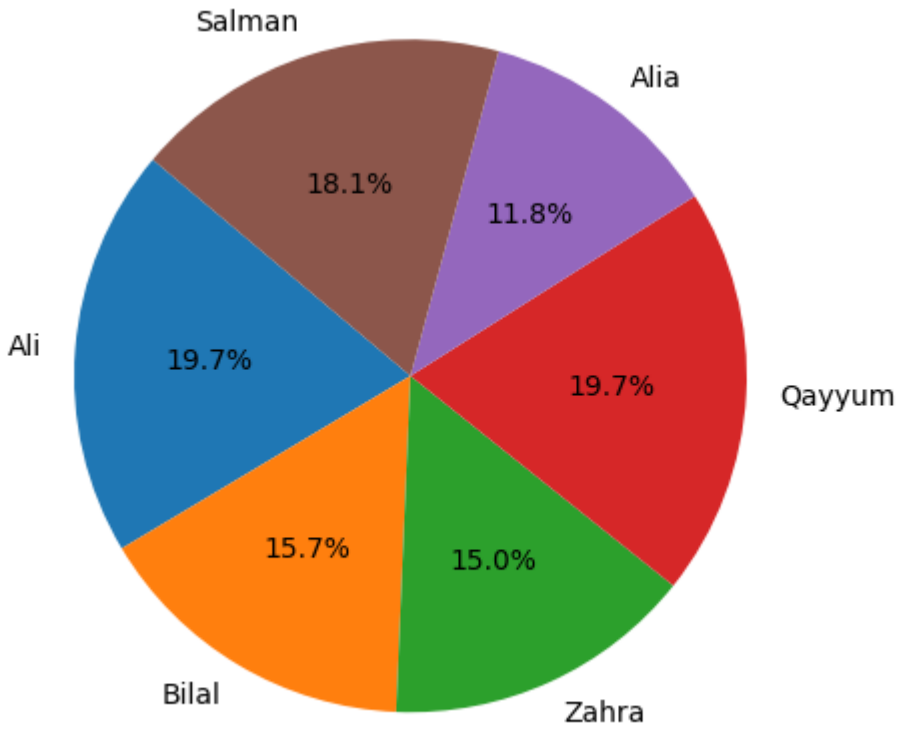
```
l1 = [1,2,3,4,5,6,7,8,9,10]
l2 = [3,8,9,6,7,1,2,8,9,1]
l3 = [1,2,3,4,1,2,3,4,5,6]

data = list([l1,l2,l3])

plt.violinplot(data)
plt.show()
```



Example 13 Show the example of pie plot

```
import matplotlib.pyplot as plt

stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

k = stuMarks.keys()
v = stuMarks.values()

plt.pie(v,labels=k, autopct='%1.1f%%', startangle=140)
plt.axis('equal')

plt.show()
```

```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```


Salman
Alia
18.1%
11.8%
Ali
19.7%
19.7%
Qayyum
15.7%
15.0%
Bilal
Zahra

……THE END……