INSTITUTE OF SPACE TECHNOLOGY
KICSIT, Kahuta Campus

Lab Task 2

Student Detail

Name: M. Nasarullah

Roll Number: 202101014

Program Detail

Program: BSCE-7

Course Title: AI (Lab)

Submitted to: Sir Zia Ur Rehman

DEPARTMENT OF COMPUTER ENGINEERING

Artificial Intelligence Lab (02)
Assignment/Quiz/Lab Task/Lab Report

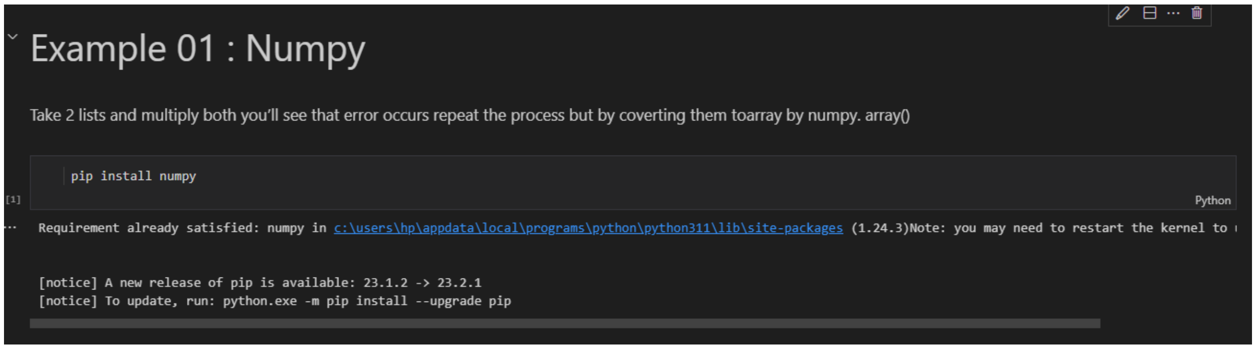
Assigned Date:

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

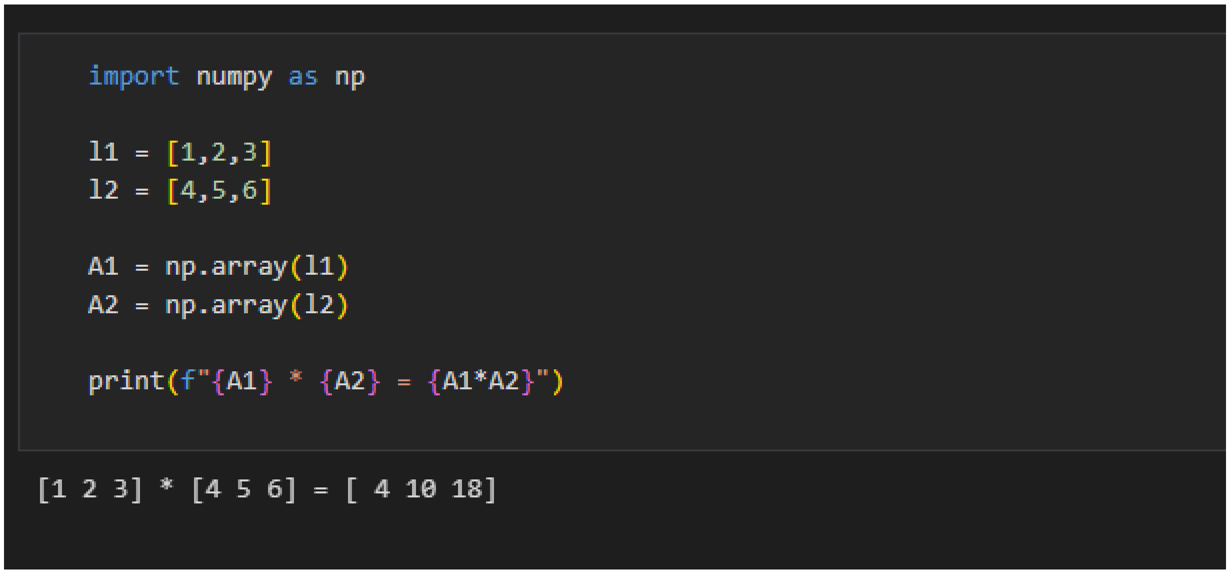


Example 01 : Numpy

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

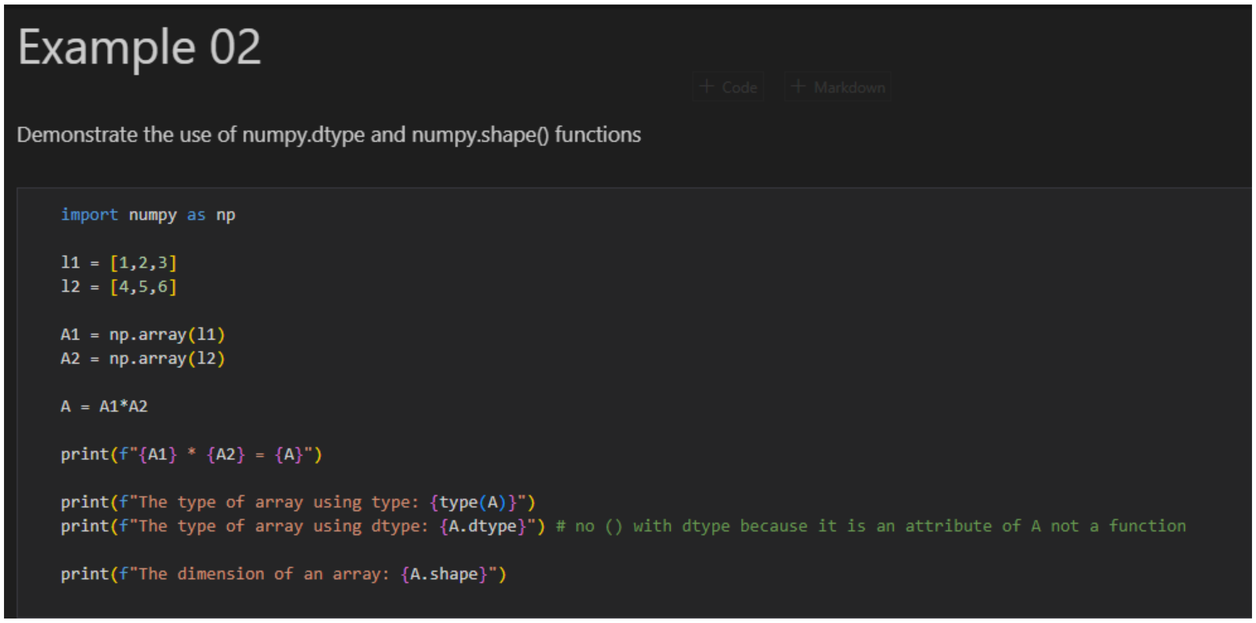
```
pip install numpy
```

```
Requirement already satisfied: numpy in c:\users\hp\appdata\local\programs\python\python311\lib\site-packages (1.24.3)Note: you may need to restart the kernel to

[notice] A new release of pip is available: 23.1.2 -> 23.2.1
[notice] To update, run: python.exe -m pip install --upgrade pip
```



```
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}")
```

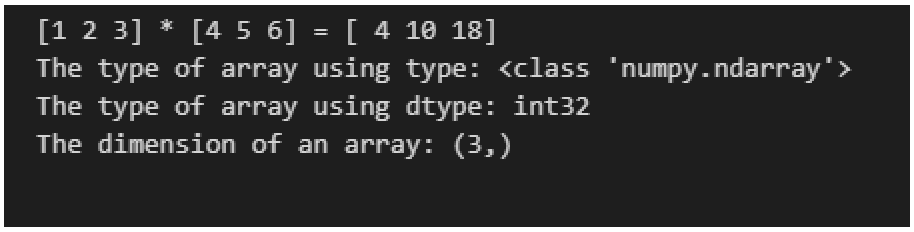
```
[1 2 3] * [4 5 6] = [ 4 10 18]
```



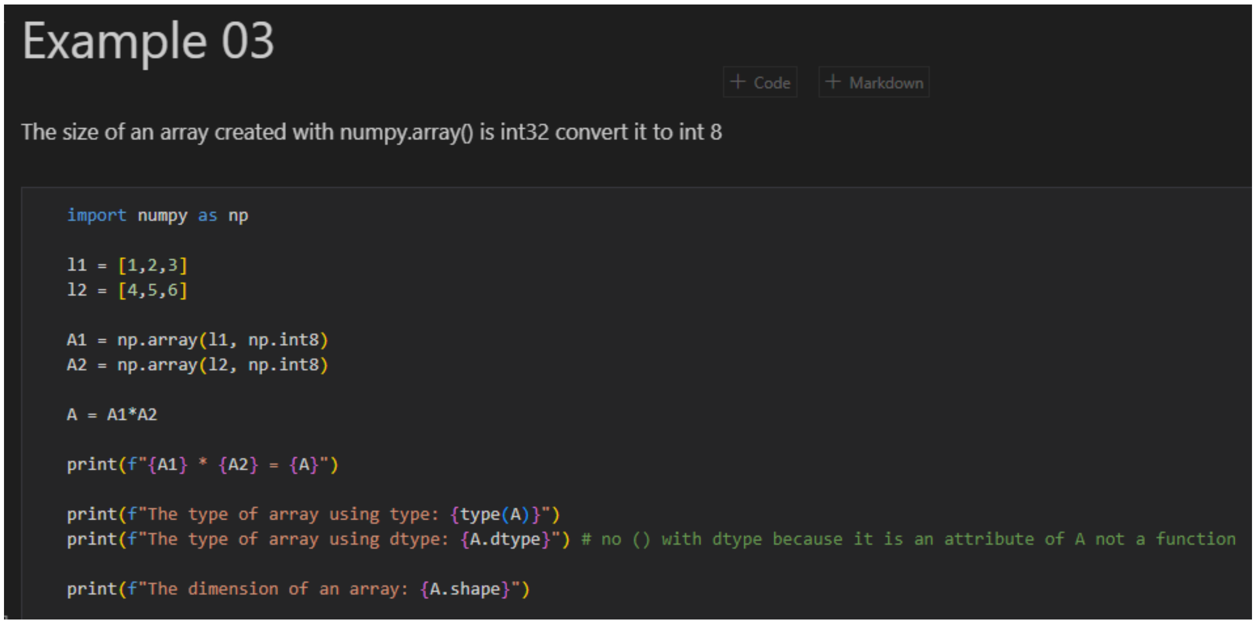
Example 02

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}")
```



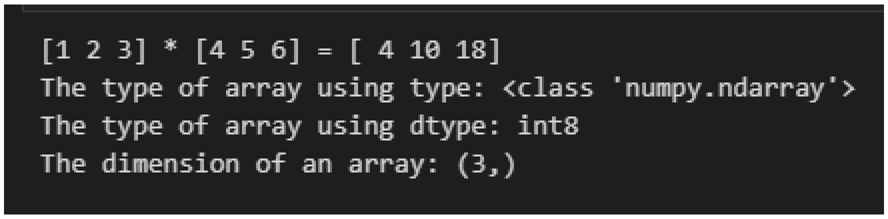
```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (3,)
```



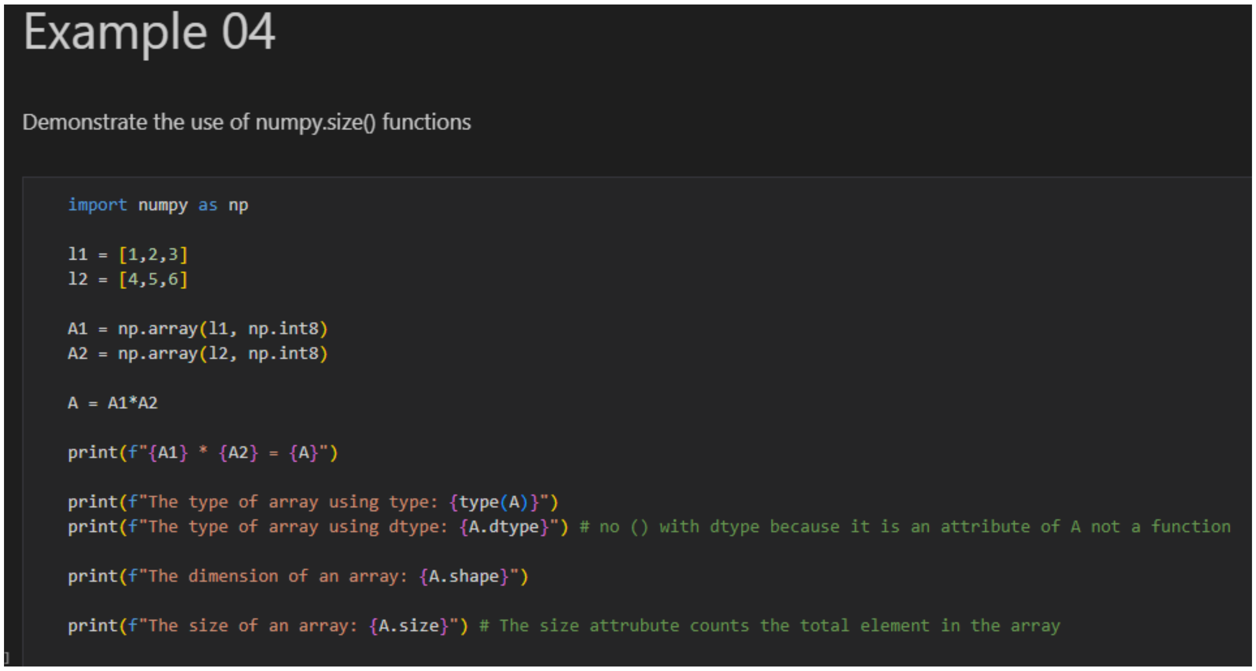
Example 03

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}")
```



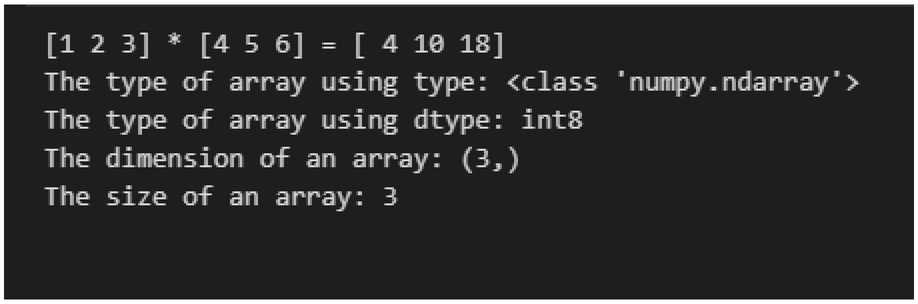
```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int8
The dimension of an array: (3,)
```



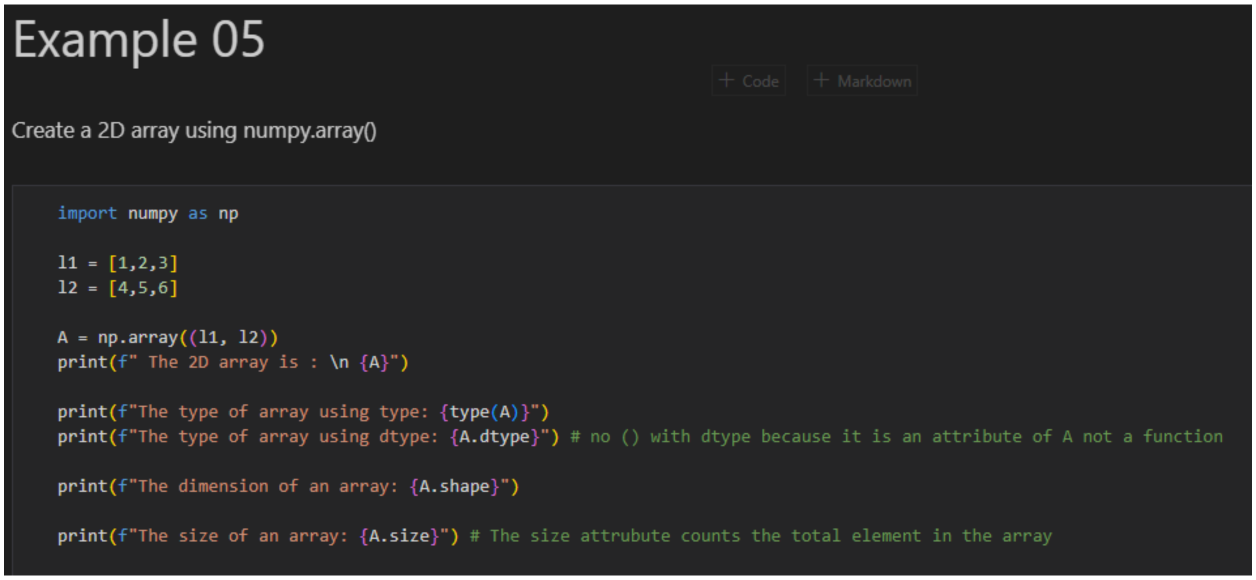
Example 04

Demonstrate the use of numpy.size() functions

```python
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
```



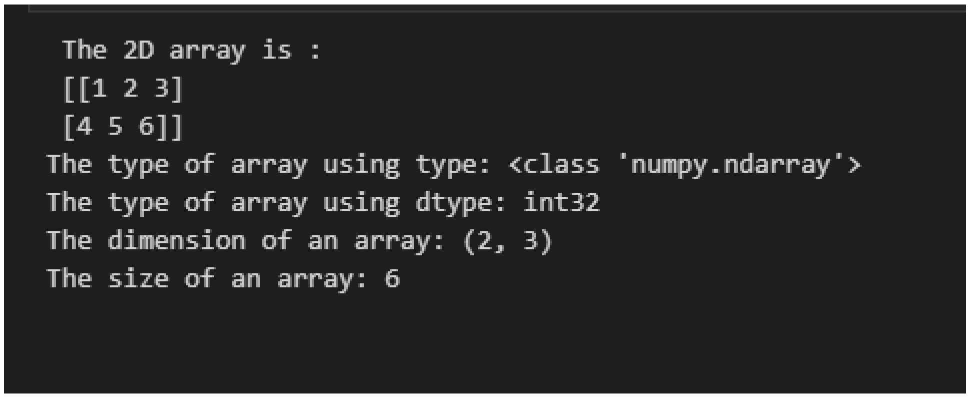
```
[1 2 3] * [4 5 6] = [ 4 10 18]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int8
The dimension of an array: (3,)
The size of an array: 3
```



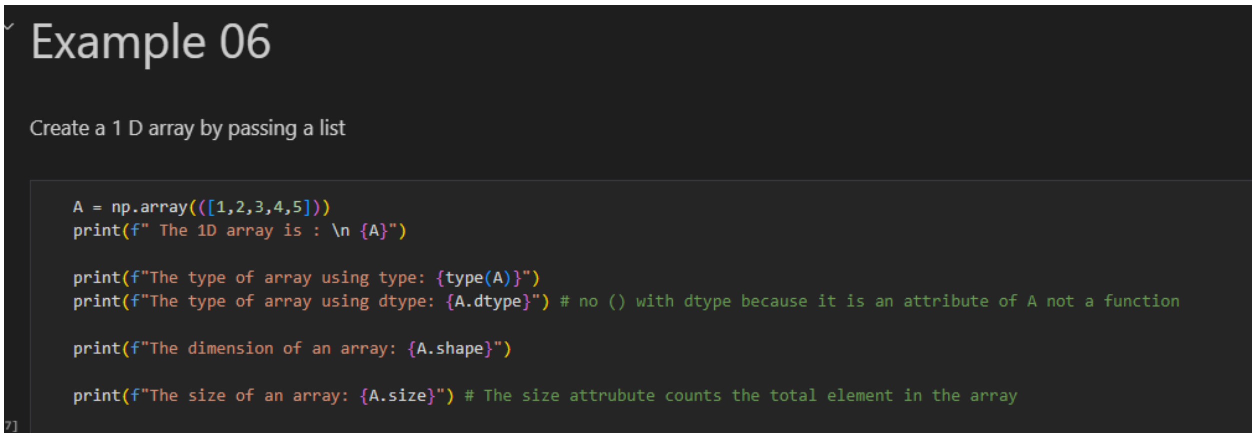
Example 05

Create a 2D array using numpy.array()

```python
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
```



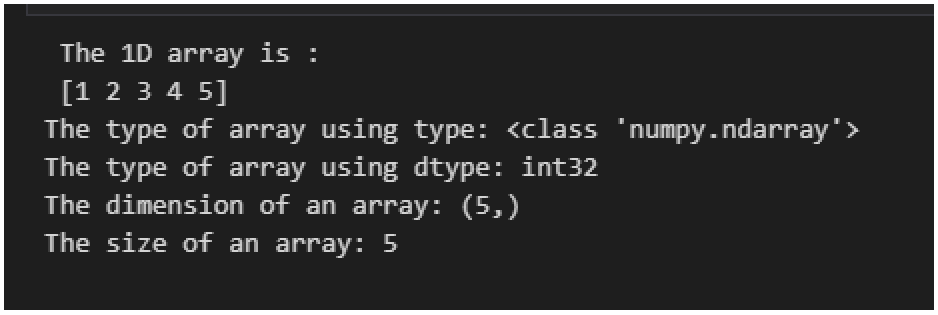
```
 The 2D array is :
 [[1 2 3]
 [4 5 6]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (2, 3)
The size of an array: 6
```



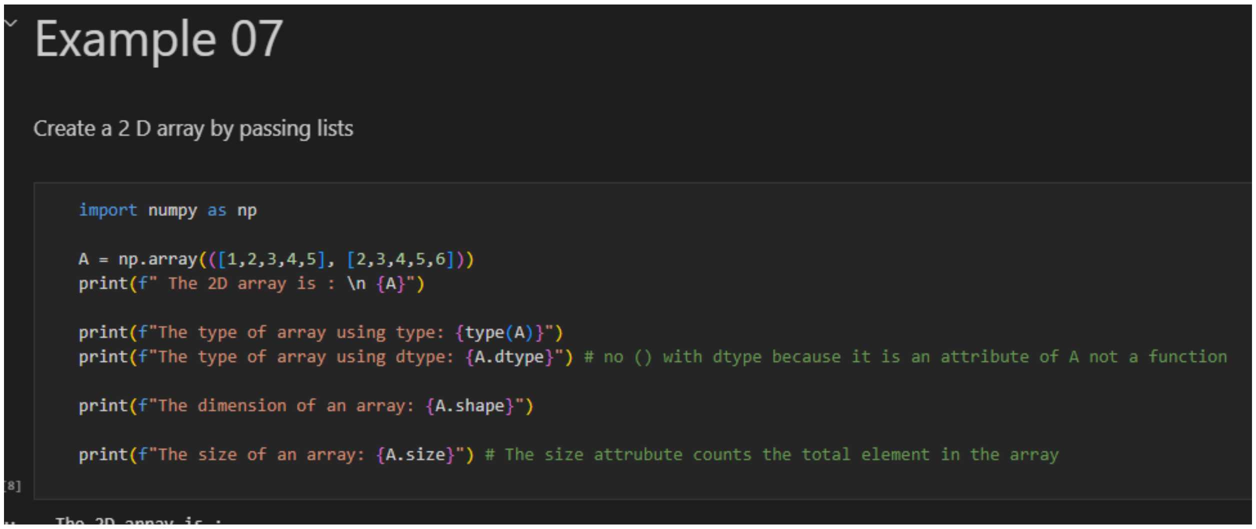
Example 06

Create a 1 D array by passing a list

```python
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
```



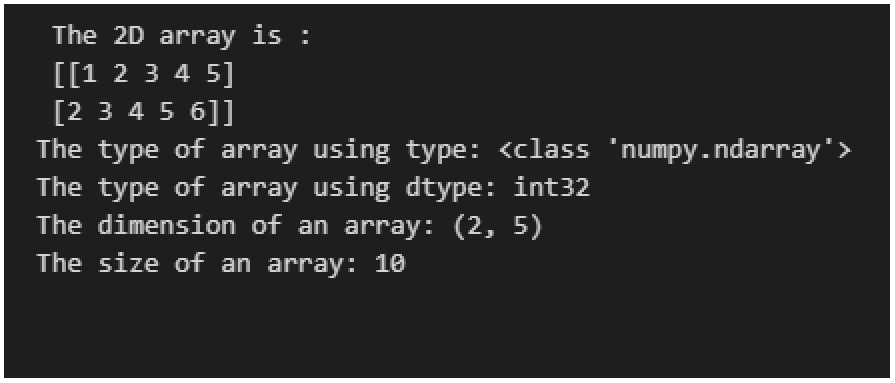
```
 The 1D array is :
 [1 2 3 4 5]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (5,)
The size of an array: 5
```



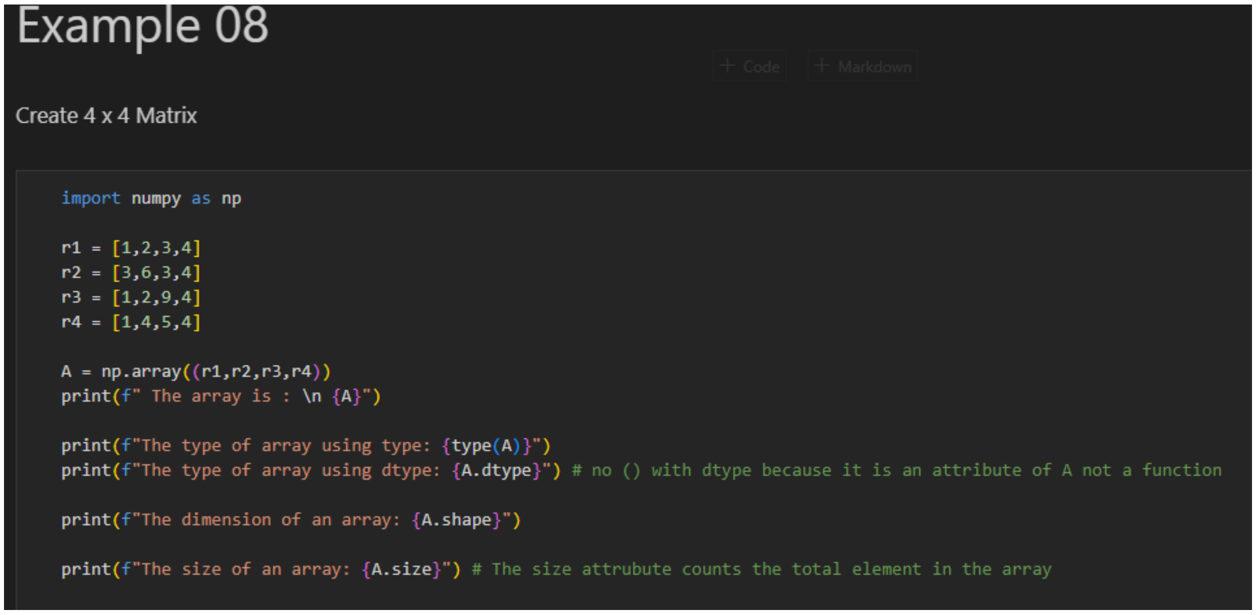
Example 07

Create a 2 D array by passing lists

```python
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
```



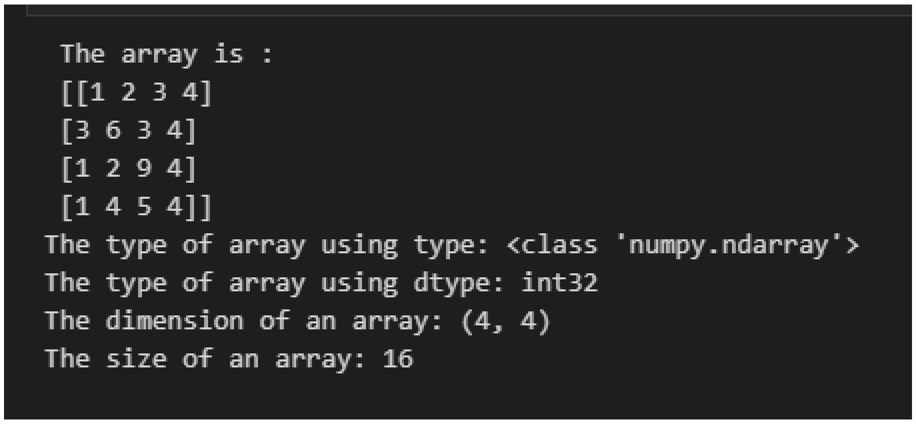
```
 The 2D array is :
 [[1 2 3 4 5]
 [2 3 4 5 6]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (2, 5)
The size of an array: 10
```



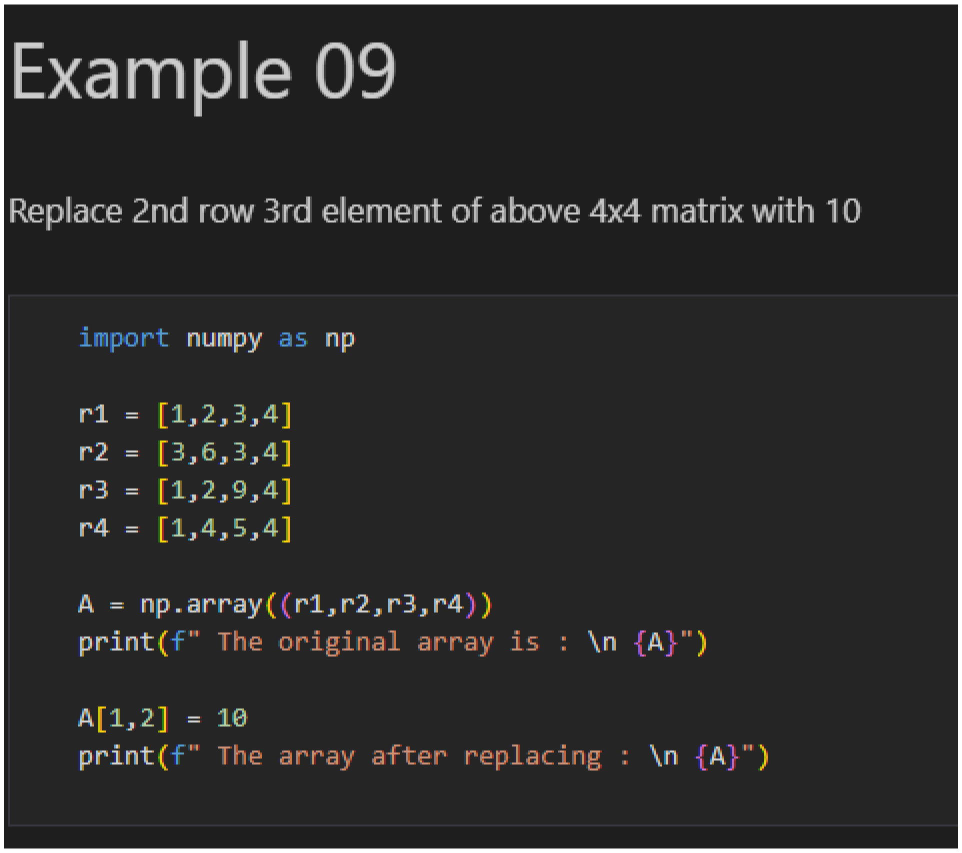
Example 08

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
```



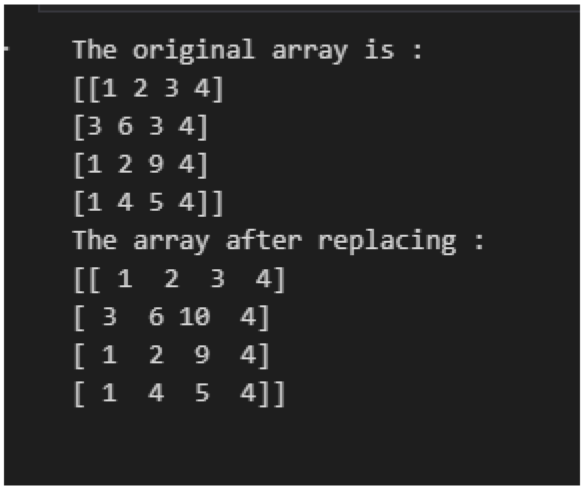
```
  The array is :
  [[1 2 3 4]
  [3 6 3 4]
  [1 2 9 4]
  [1 4 5 4]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (4, 4)
The size of an array: 16
```



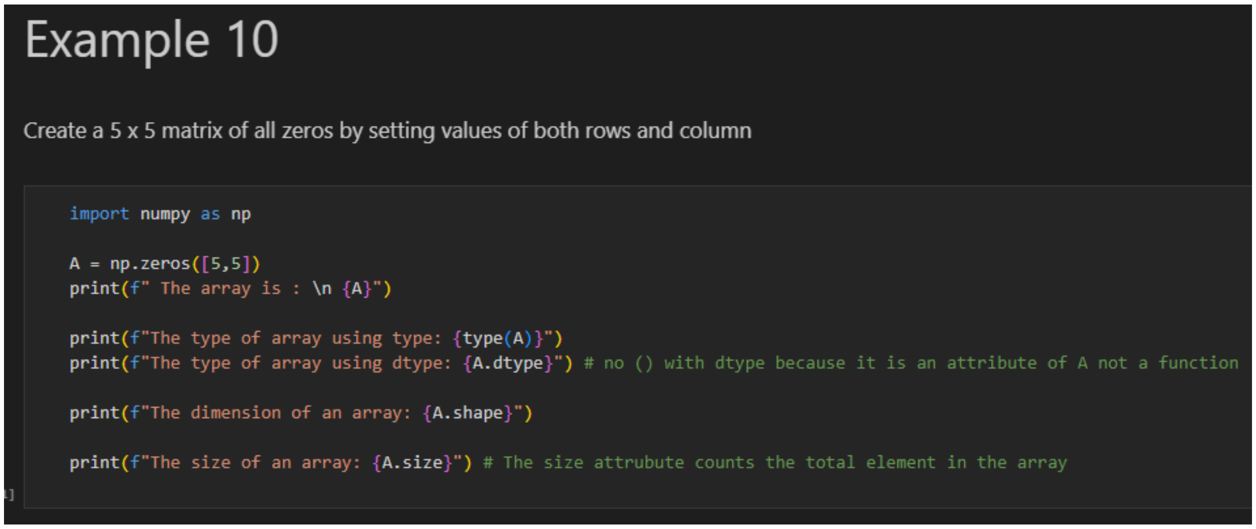
Example 09

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

```python
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}")
```



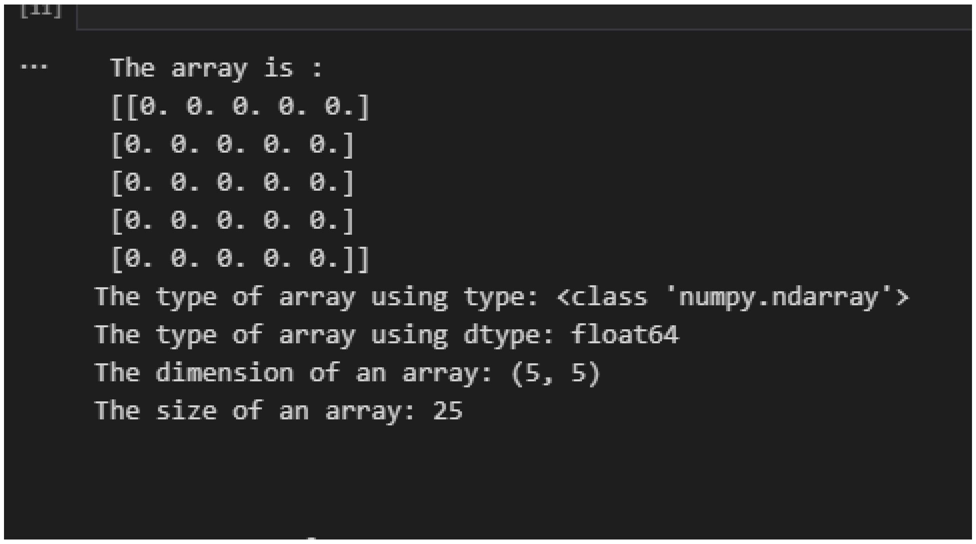
```
The original array is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The array after replacing :
[[ 1  2  3  4]
[ 3  6 10  4]
[ 1  2  9  4]
[ 1  4  5  4]]
```



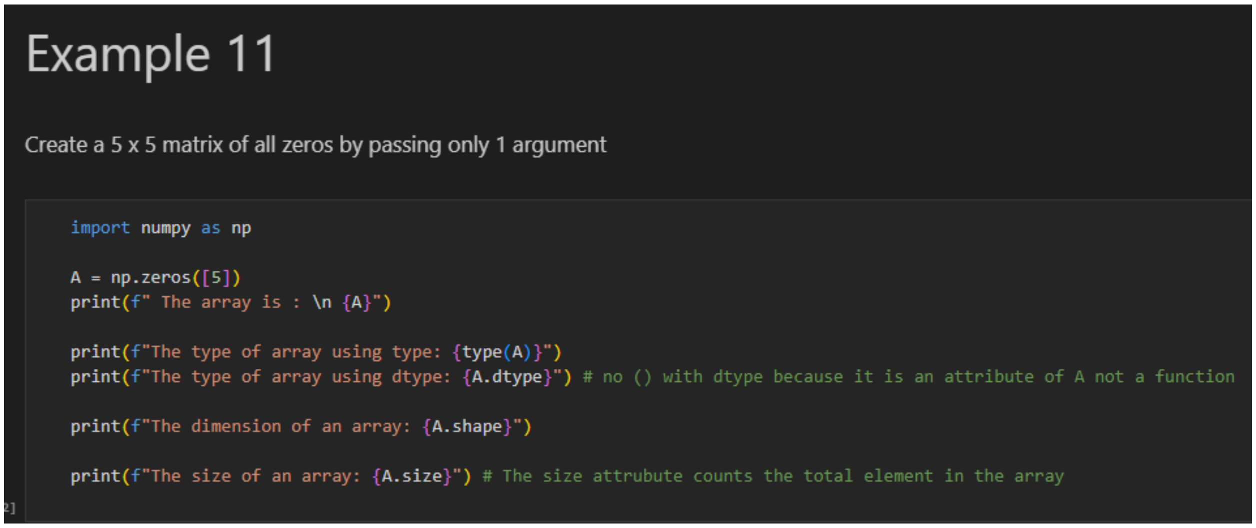
Example 10

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

```python
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
```



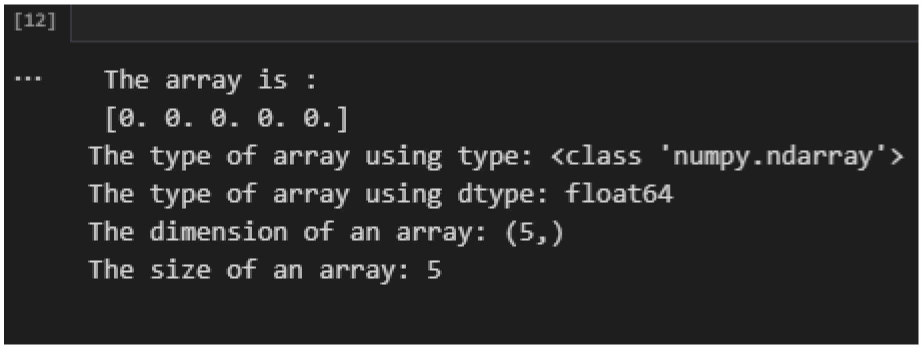
```
 The array is :
 [[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (5, 5)
The size of an array: 25
```



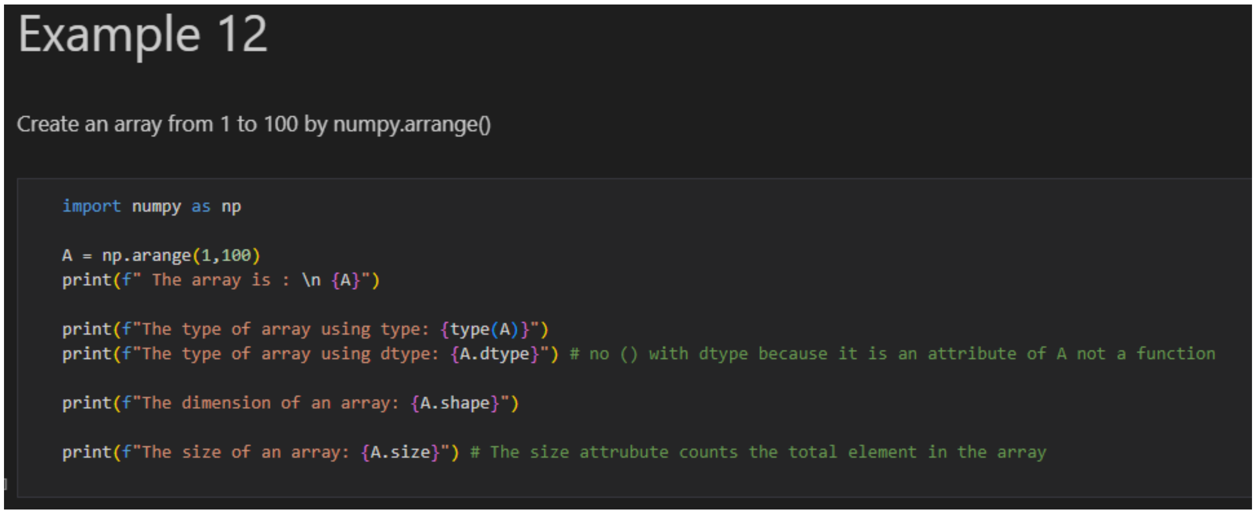
Example 11

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

```python
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
```



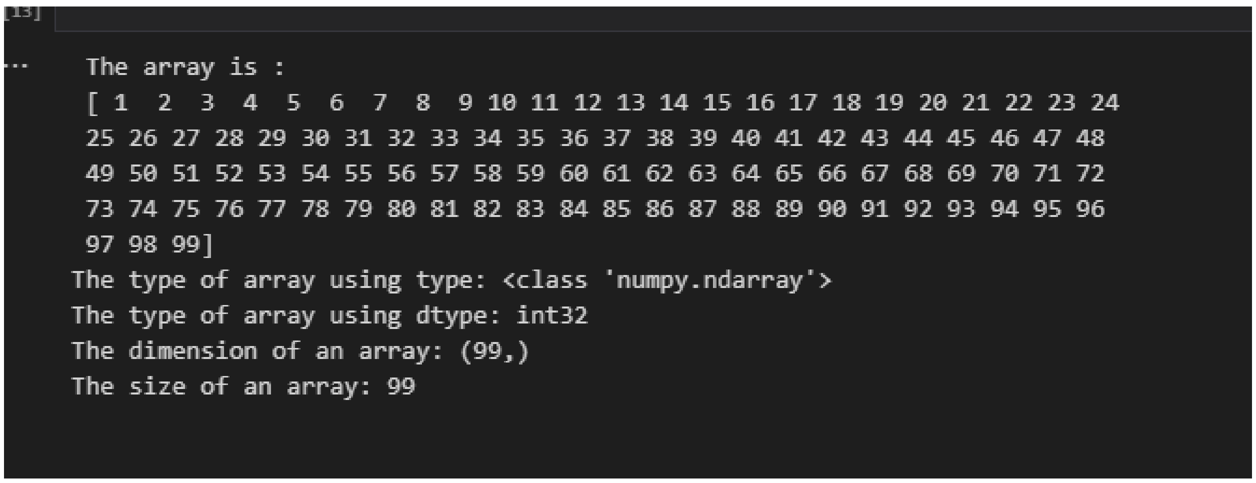
```
[12]

...    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
```



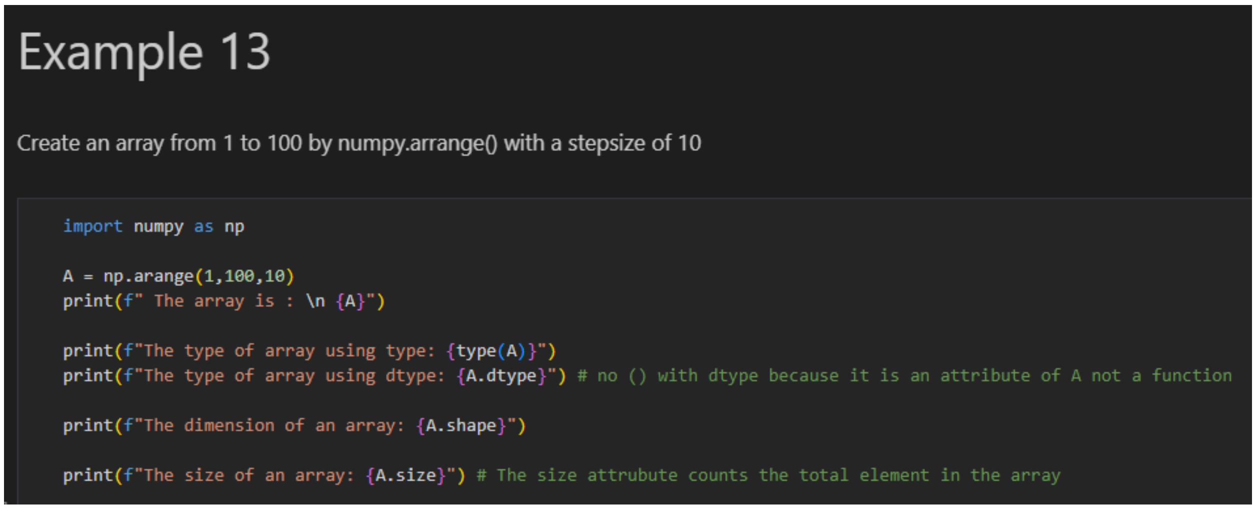
Example 12

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

```python
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
```



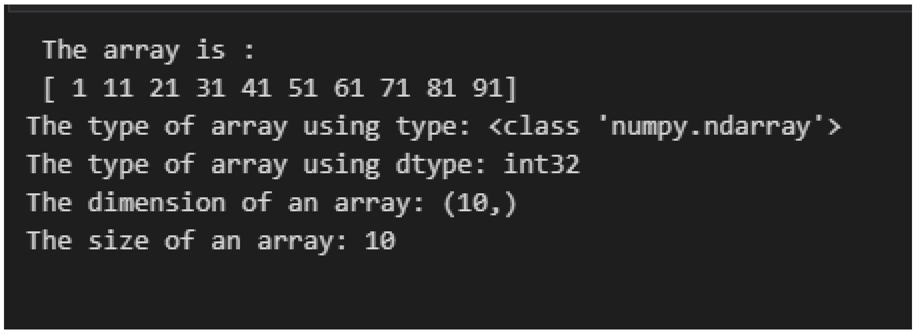
```
[13]

...    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
```



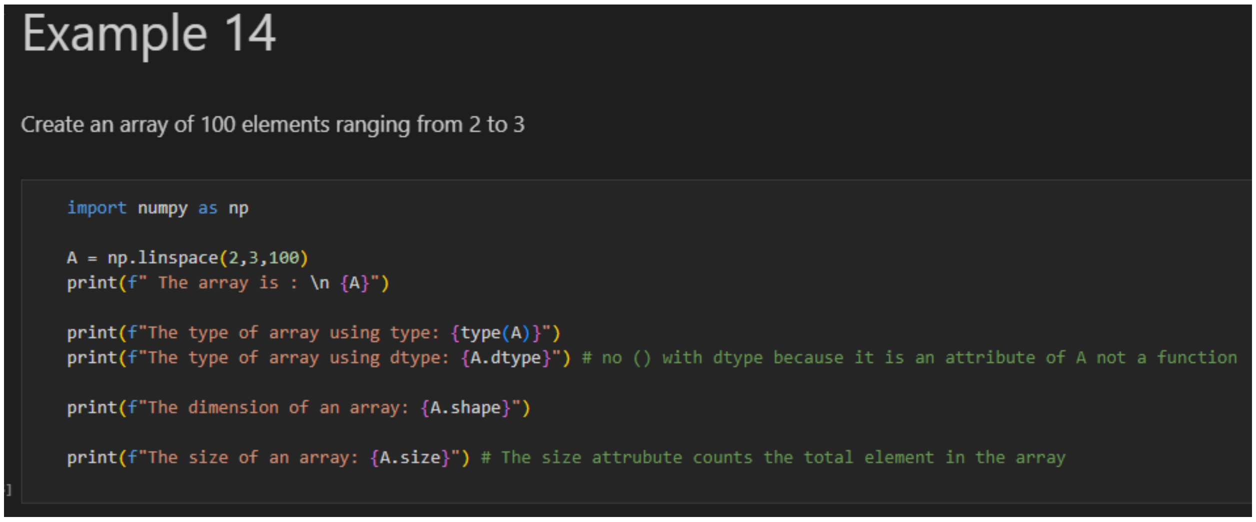
Example 13

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
```



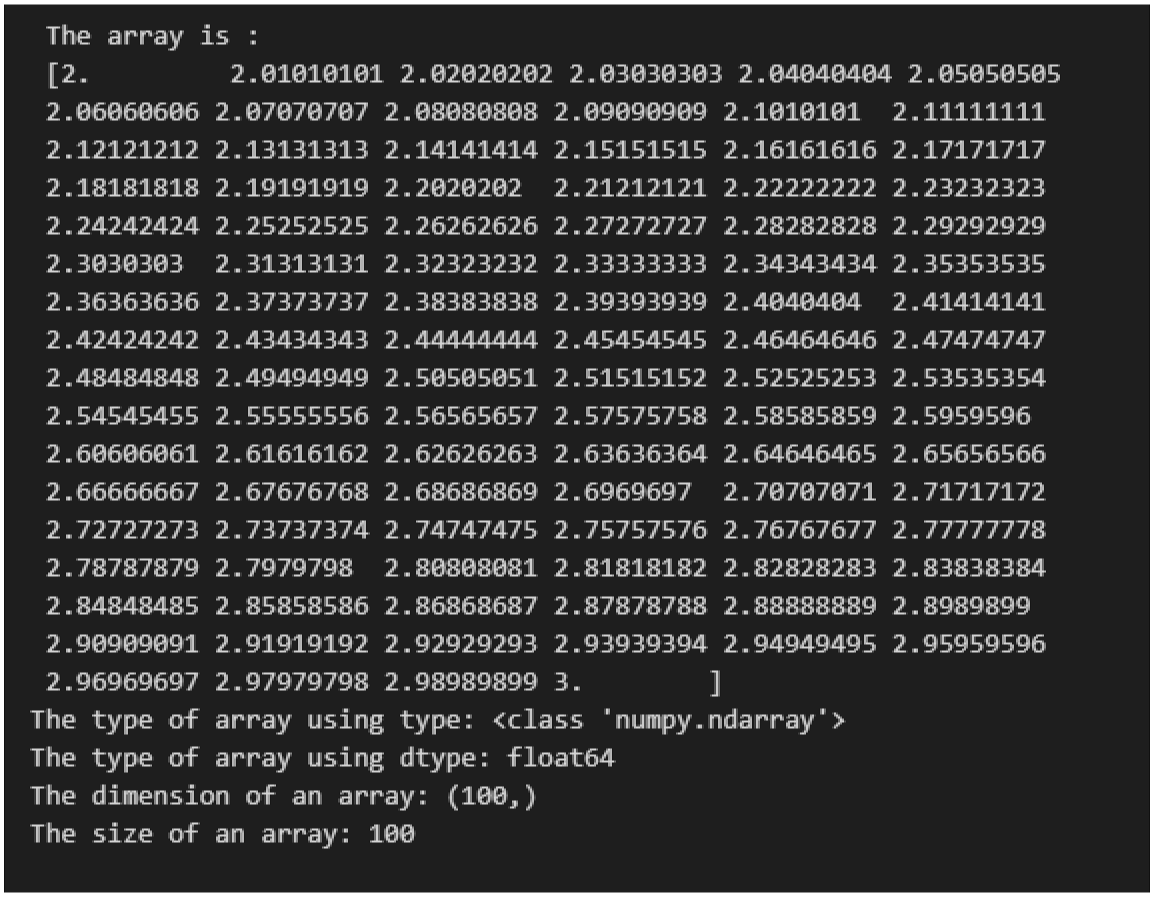
```
 The array is :
 [ 1 11 21 31 41 51 61 71 81 91]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (10,)
The size of an array: 10
```



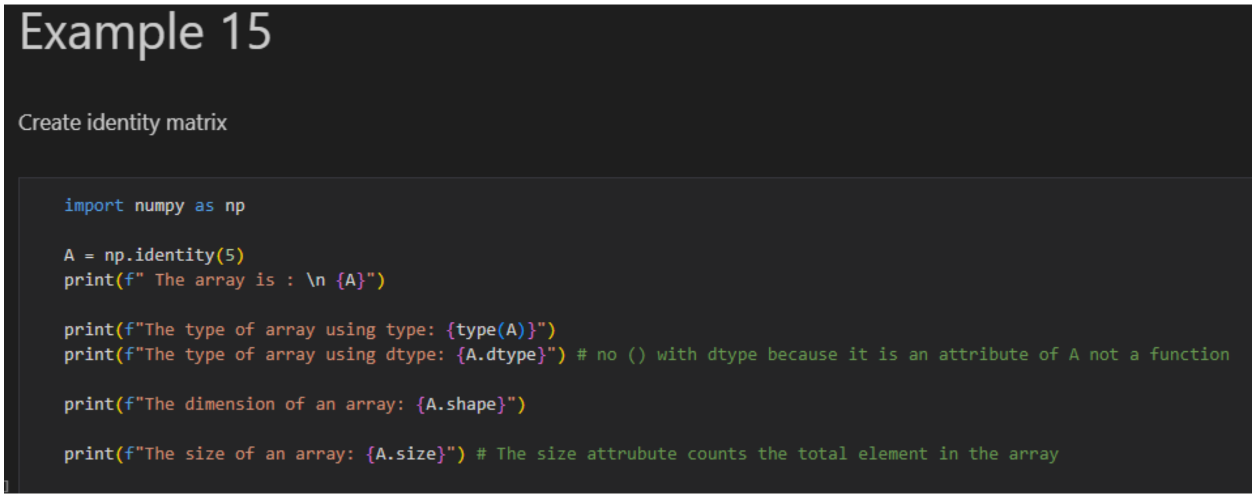
Example 14

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
```



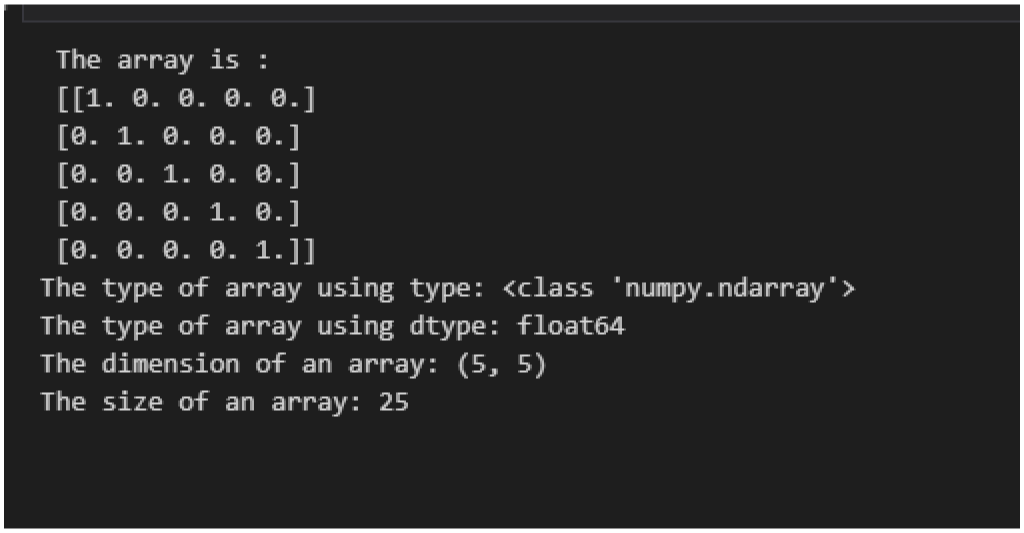
```
 The array is :
 [2.         2.01010101 2.02020202 2.03030303 2.04040404 2.05050505
 2.06060606 2.07070707 2.08080808 2.09090909 2.1010101  2.11111111
 2.12121212 2.13131313 2.14141414 2.15151515 2.16161616 2.17171717
 2.18181818 2.19191919 2.2020202  2.21212121 2.22222222 2.23232323
 2.24242424 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
```

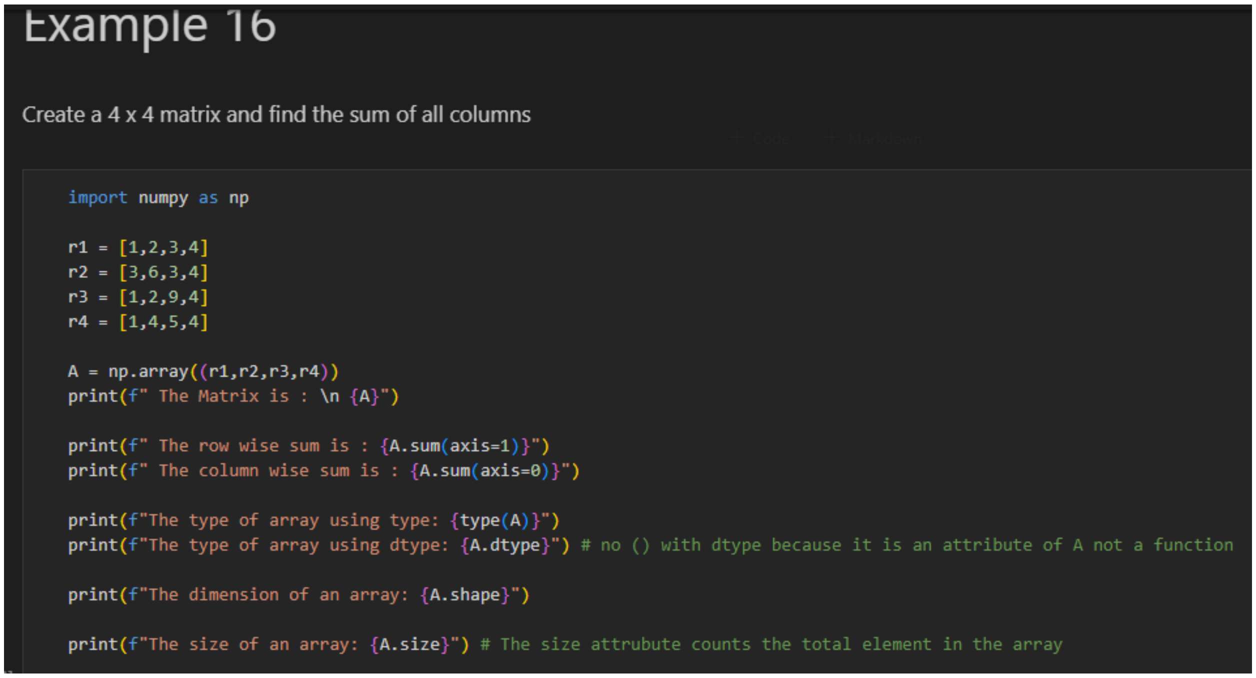


Example 15

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
```

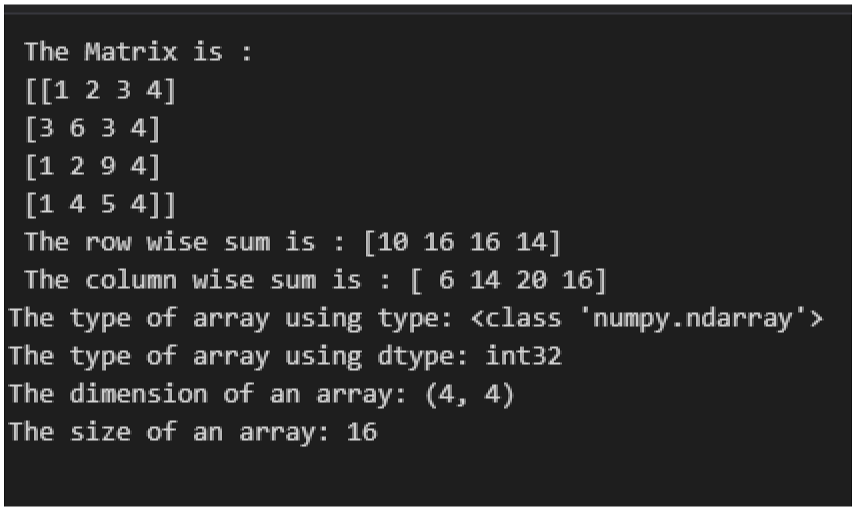

```
The array is :
[[1. 0. 0. 0. 0.]
[0. 1. 0. 0. 0.]
[0. 0. 1. 0. 0.]
[0. 0. 0. 1. 0.]
[0. 0. 0. 0. 1.]]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: float64
The dimension of an array: (5, 5)
The size of an array: 25
```



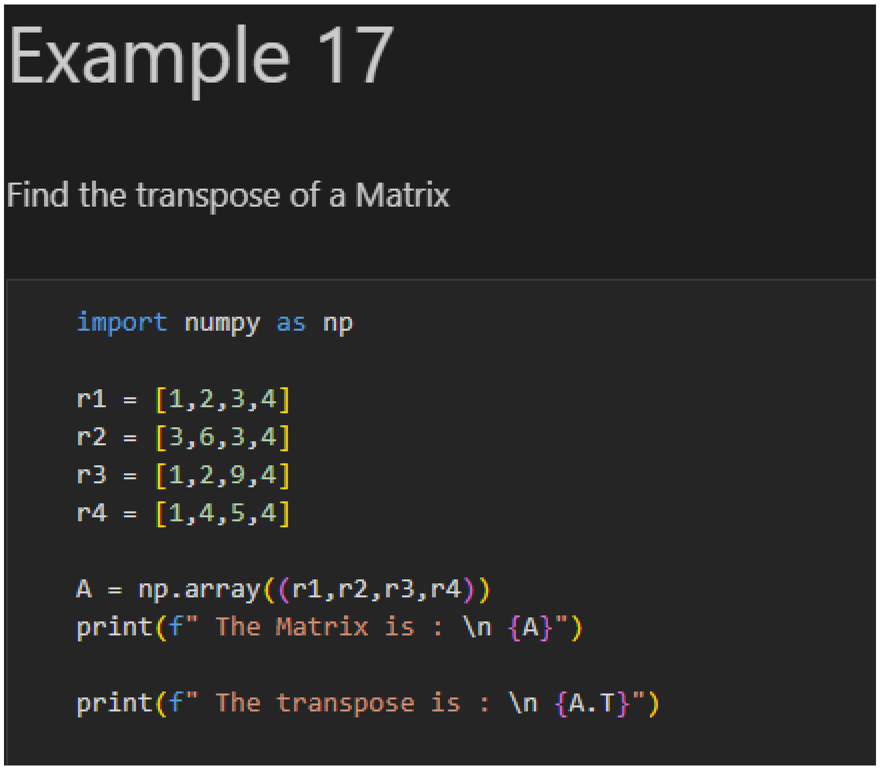
Example 16

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
```



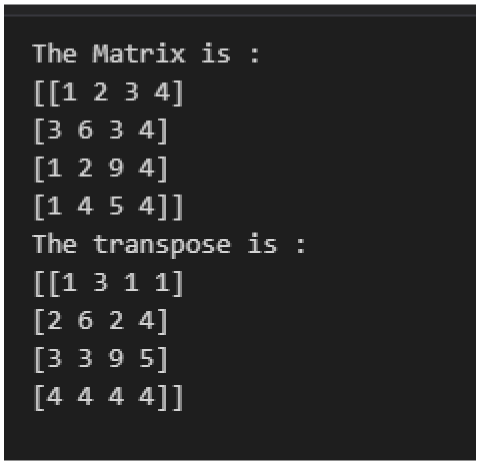
```
 The Matrix is :
 [[1 2 3 4]
 [3 6 3 4]
 [1 2 9 4]
 [1 4 5 4]]
 The row wise sum is : [10 16 16 14]
 The column wise sum is : [ 6 14 20 16]
The type of array using type: <class 'numpy.ndarray'>
The type of array using dtype: int32
The dimension of an array: (4, 4)
The size of an array: 16
```



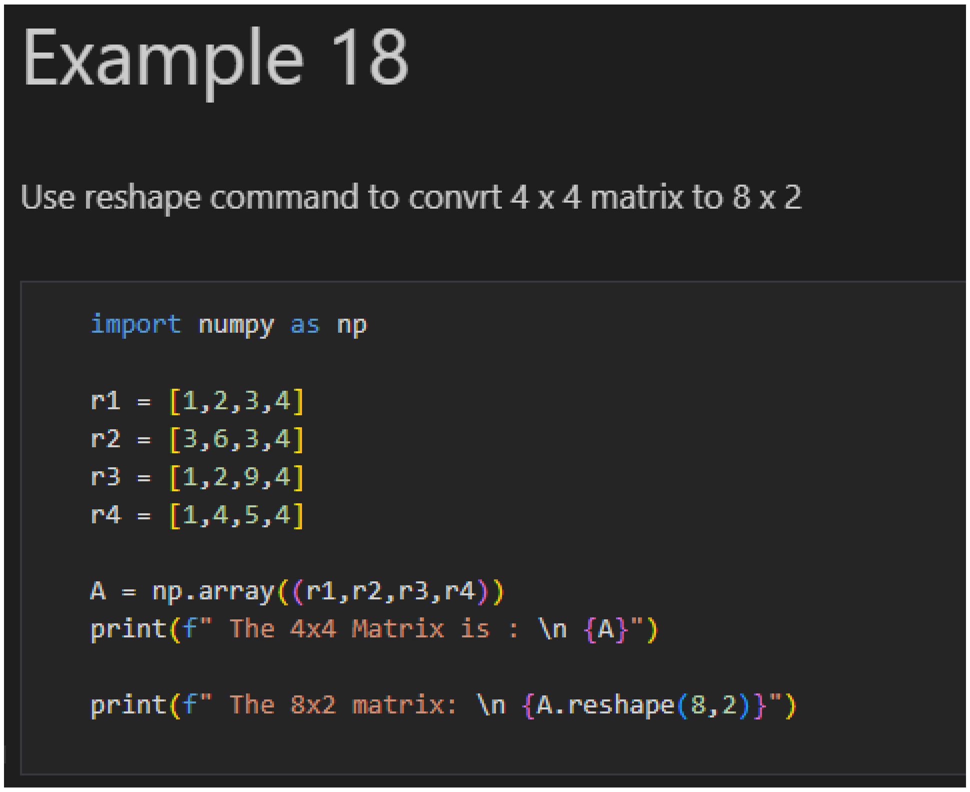
Example 17

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}")
```



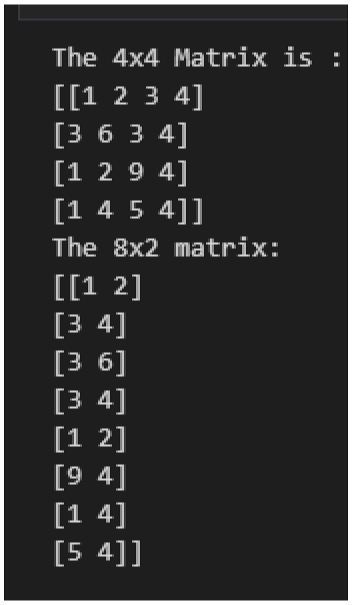
```
The Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The transpose is :
[[1 3 1 1]
[2 6 2 4]
[3 3 9 5]
[4 4 4 4]]
```



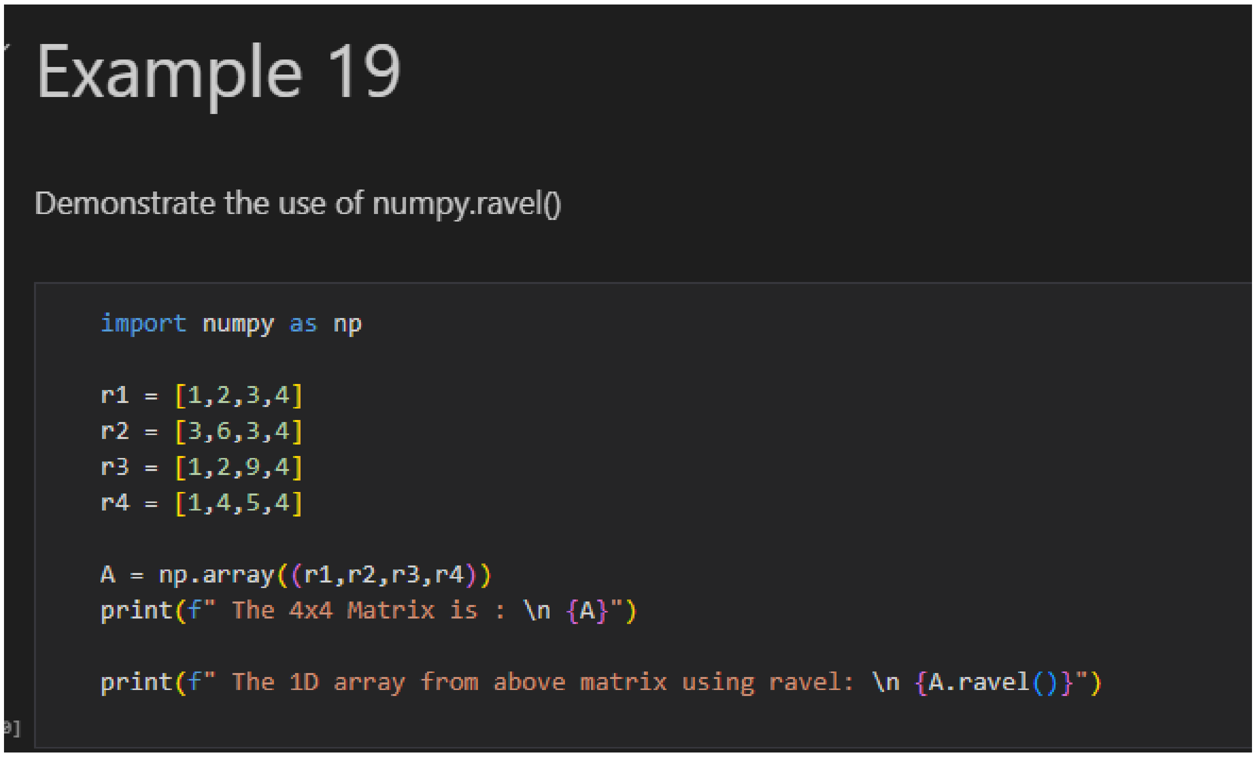
Example 18

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)}")
```



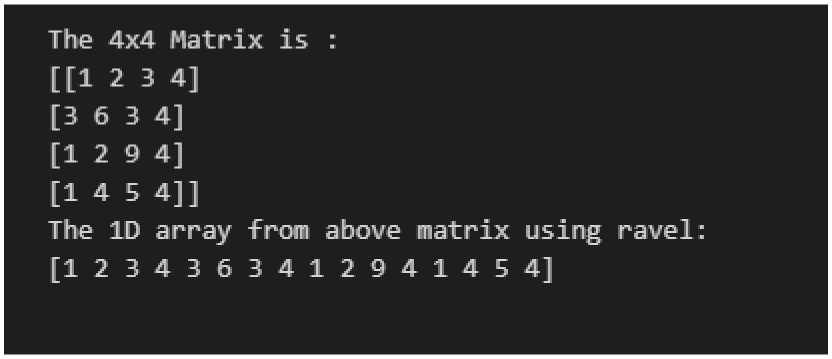
```
The 4x4 Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The 8x2 matrix:
[[1 2]
[3 4]
[3 6]
[3 4]
[1 2]
[9 4]
[1 4]
[5 4]]
```



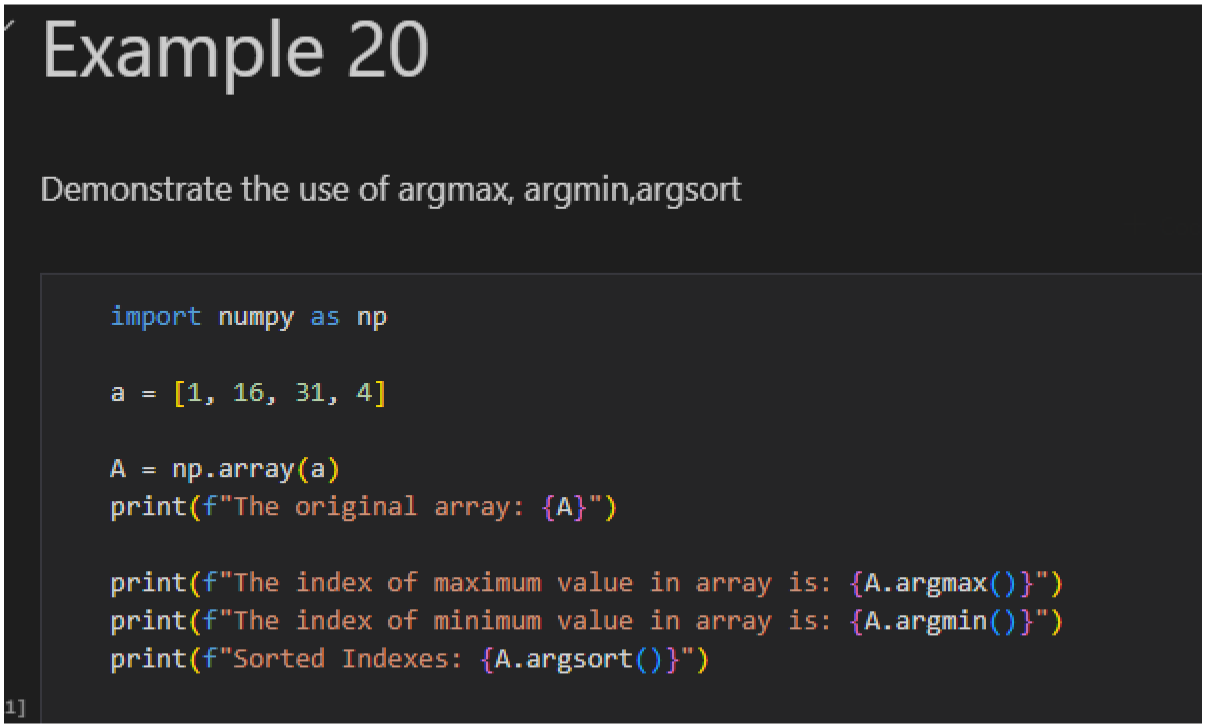
Example 19

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()}")
```



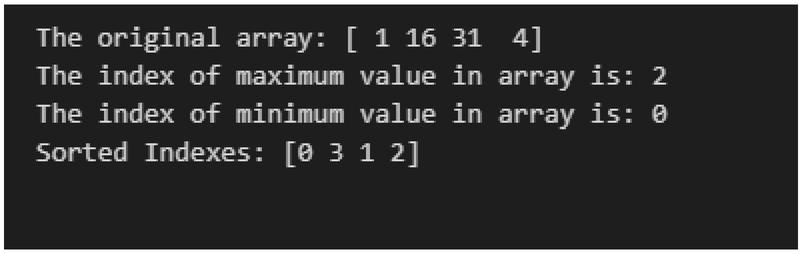
```
The 4x4 Matrix is :
[[1 2 3 4]
[3 6 3 4]
[1 2 9 4]
[1 4 5 4]]
The 1D array from above matrix using ravel:
[1 2 3 4 3 6 3 4 1 2 9 4 1 4 5 4]
```



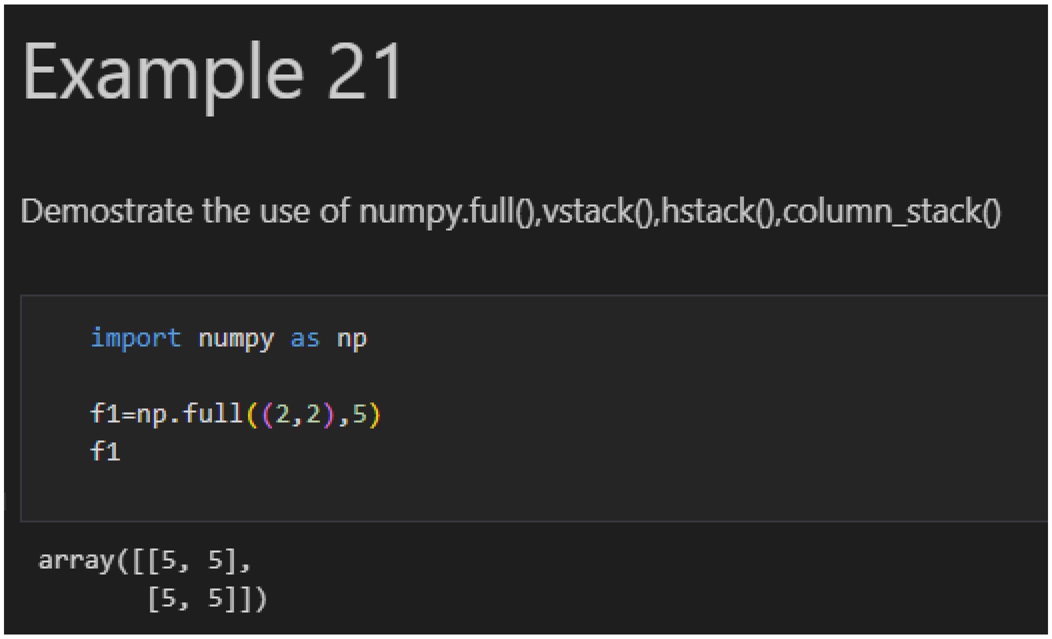
Example 20

Demonstrate the use of argmax, argmin,argsort

```python
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]
```

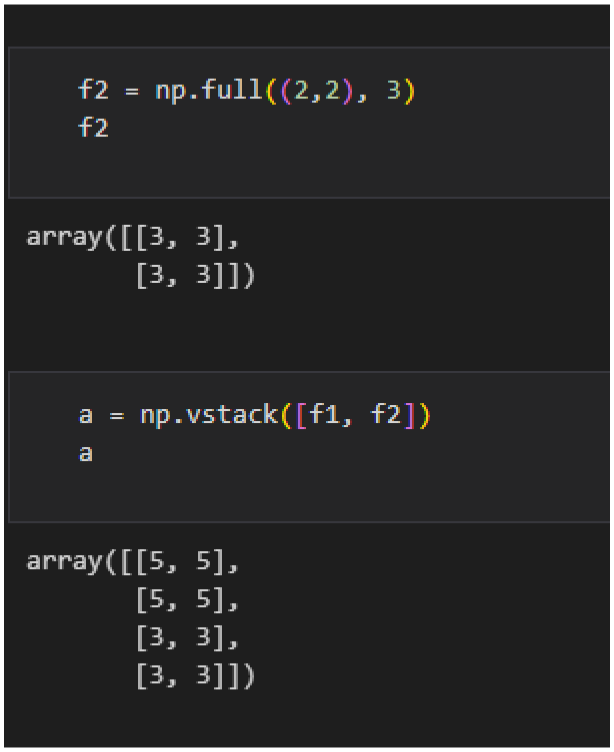


Example 21

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]])
```

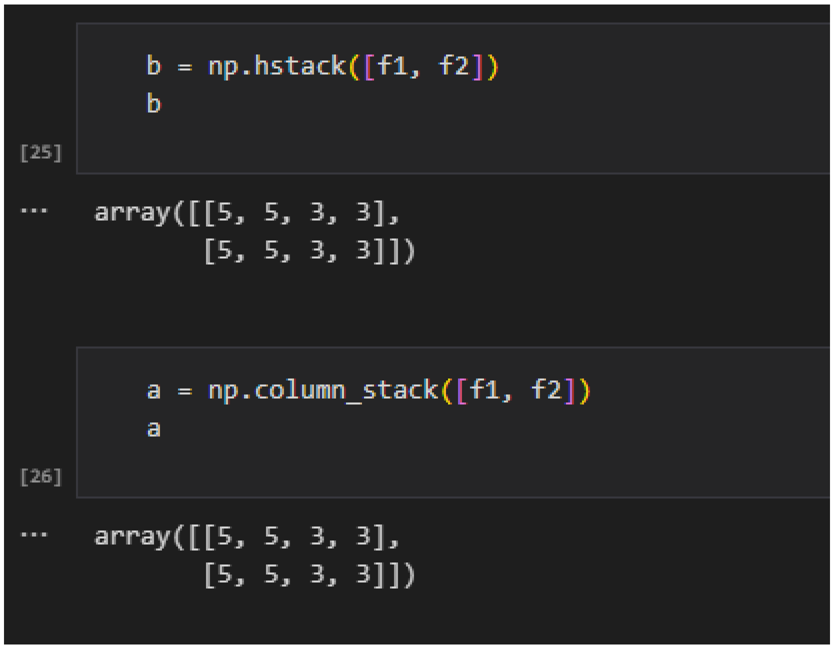


```
f2 = np.full((2,2), 3)
f2
```

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

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

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

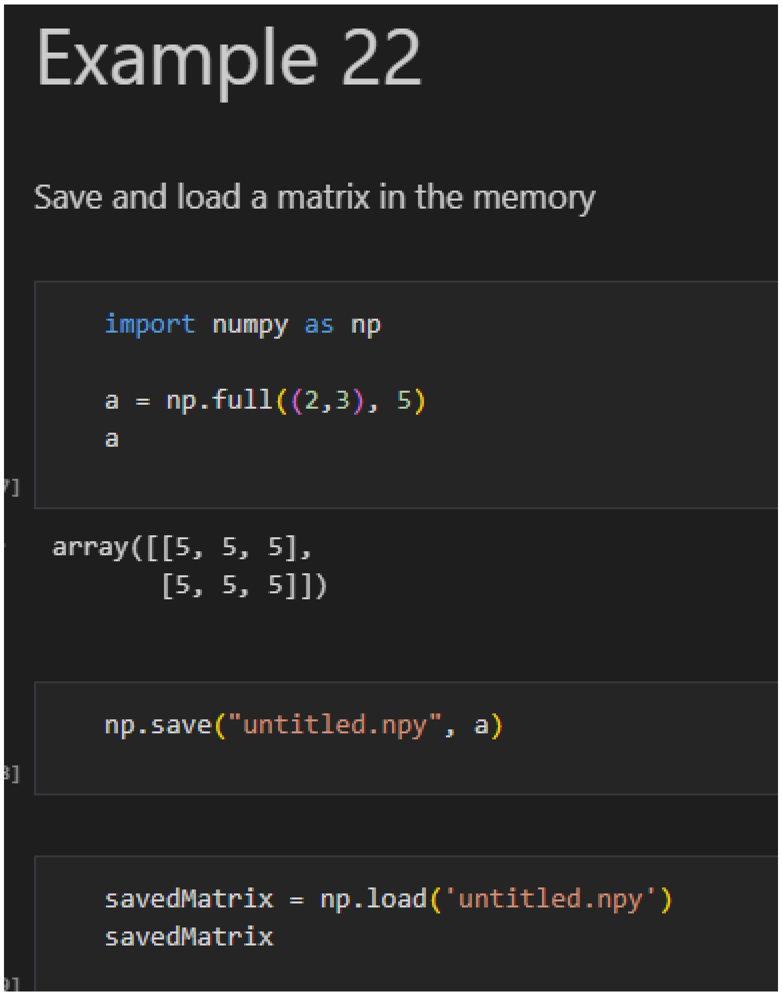


```
b = np.hstack([f1, f2])
b
```
[25]

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

```
a = np.column_stack([f1, f2])
a
```
[26]

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



Example 22

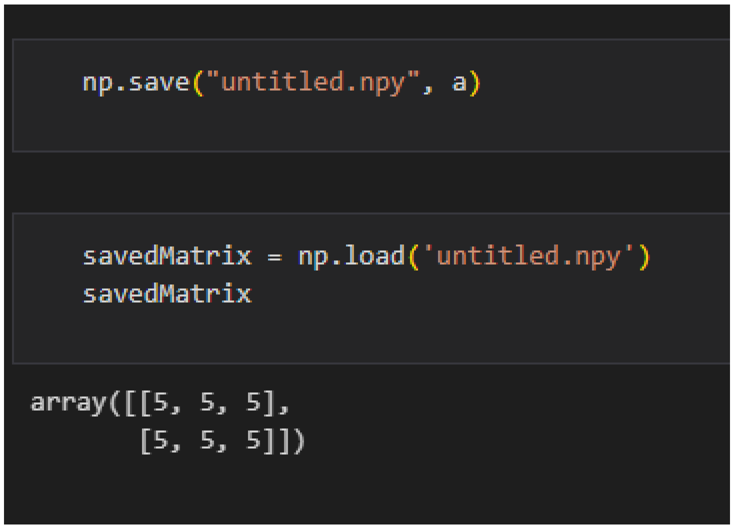
Save and load a matrix in the memory

```
import numpy as np

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

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

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

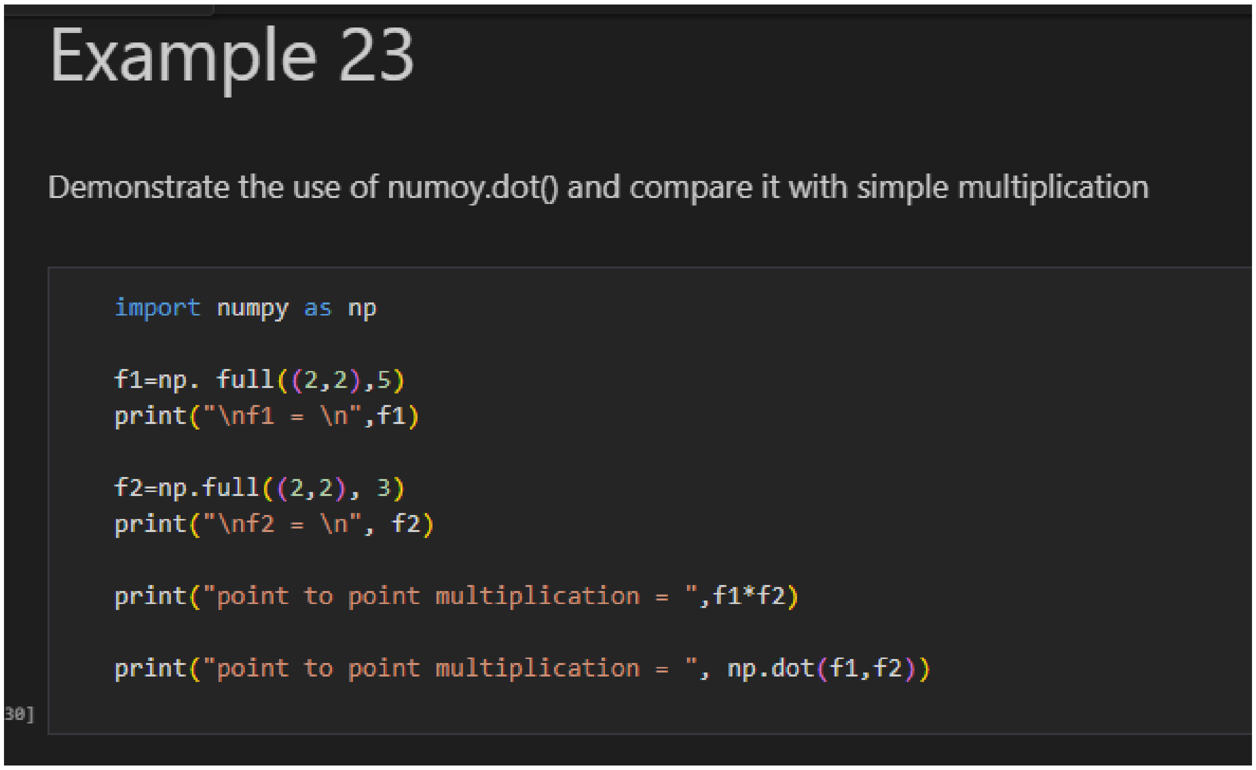
```
savedMatrix = np.load('untitled.npy')
savedMatrix
```



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

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

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

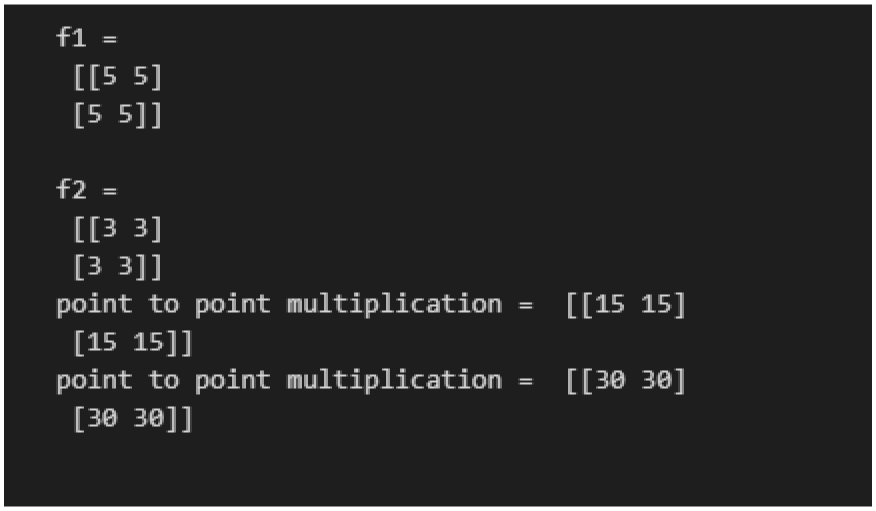


Example 23

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))
```

30]

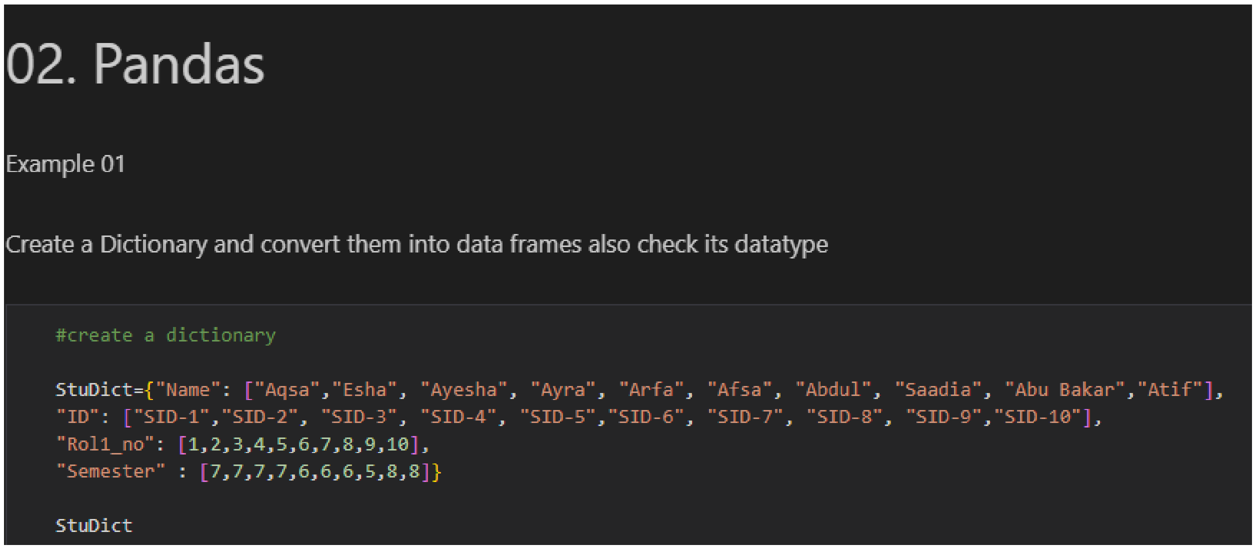


```
f1 =
 [[5 5]
 [5 5]]

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

2. **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).

[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

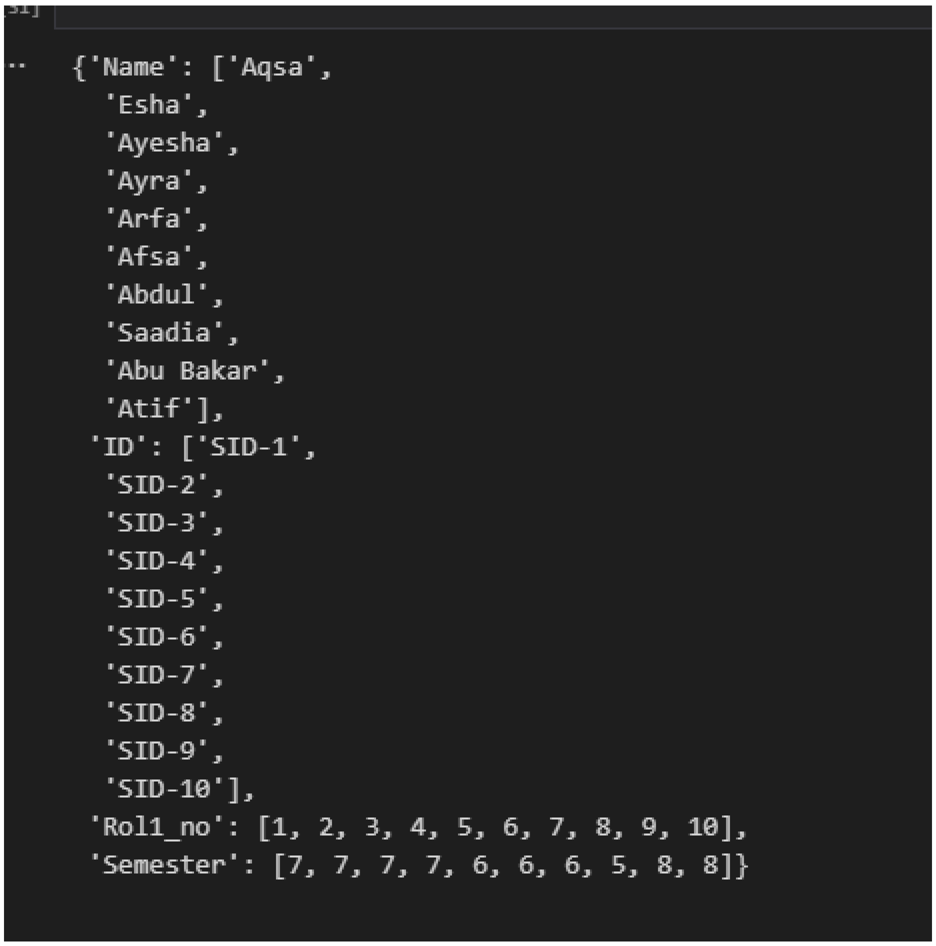


02. Pandas

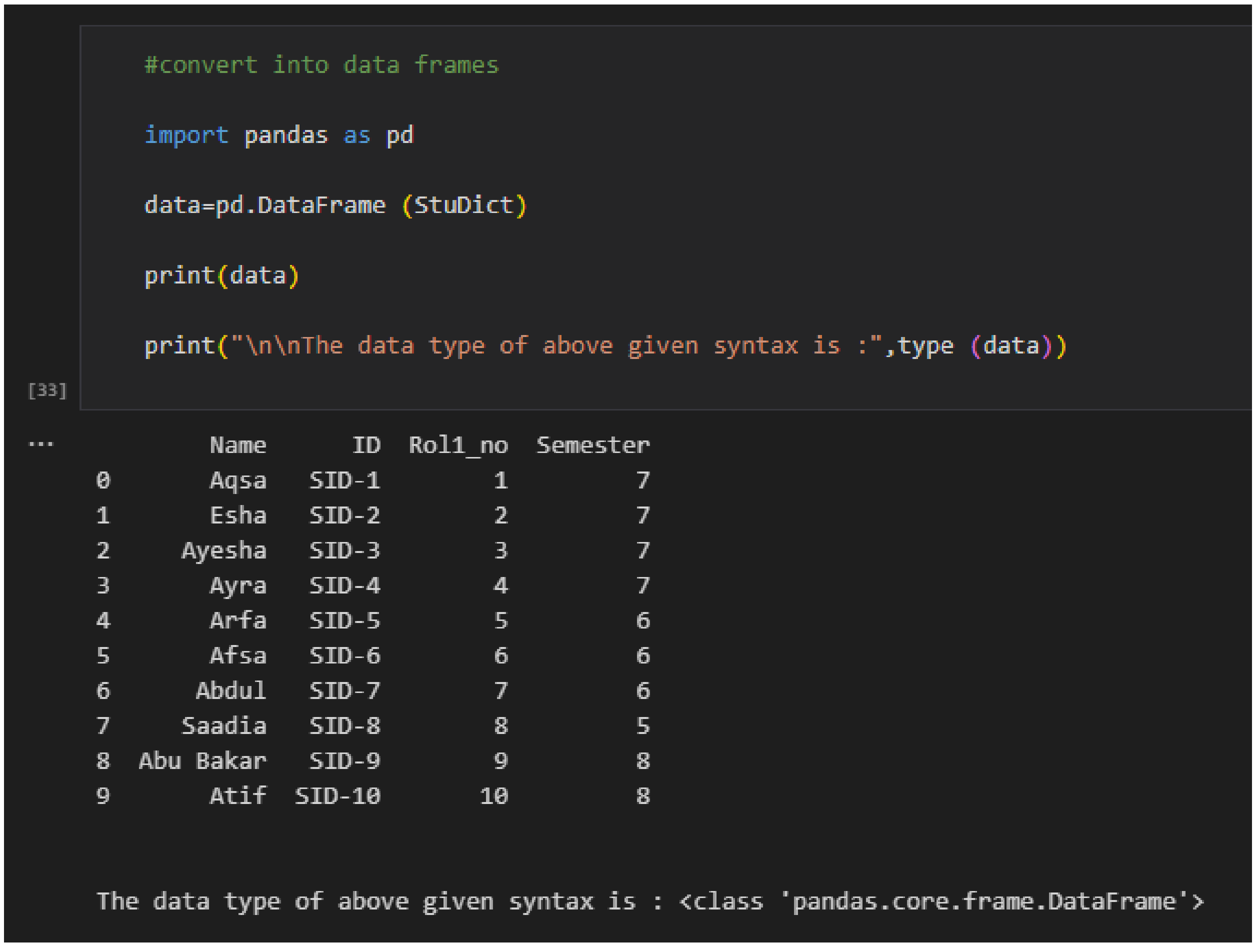
Example 01

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"],
"Roll_no": [1,2,3,4,5,6,7,8,9,10],
"Semester" : [7,7,7,7,6,6,6,5,8,8]}

StuDict
```



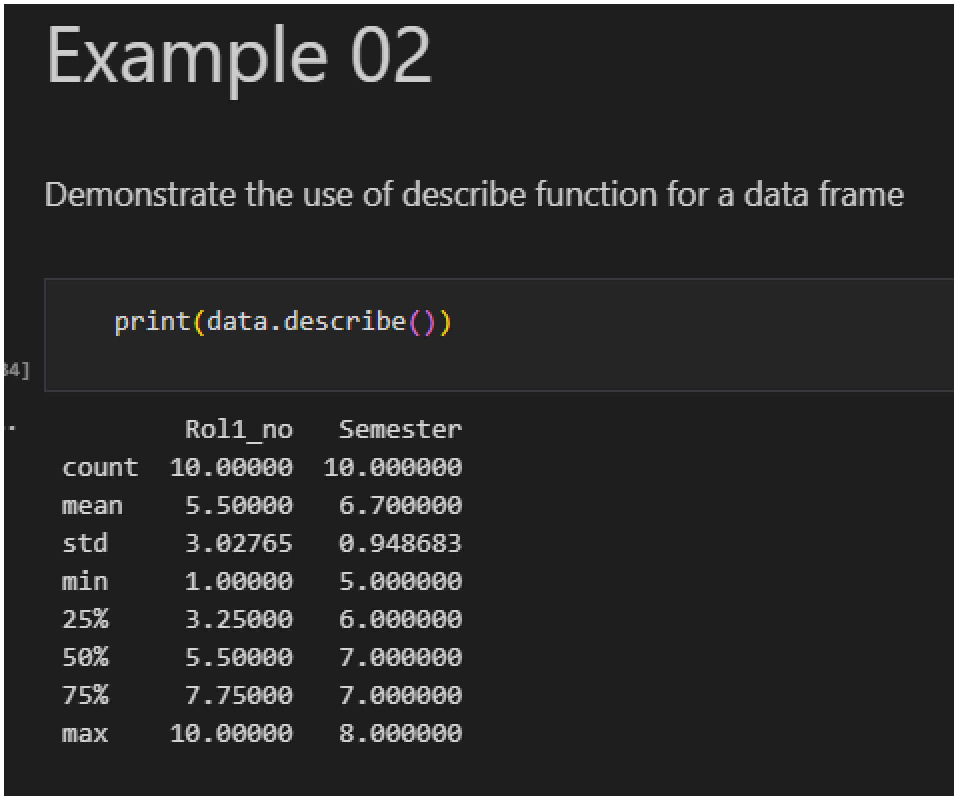
```
{'Name': ['Aqsa',
  'Esha',
  'Ayesha',
  'Ayra',
  'Arfa',
  'Afsa',
  'Abdul',
  'Saadia',
  'Abu Bakar',
  'Atif'],
 'ID': ['SID-1',
  'SID-2',
  'SID-3',
  'SID-4',
  'SID-5',
  'SID-6',
  'SID-7',
  'SID-8',
  'SID-9',
  'SID-10'],
 'Roll_no': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
 'Semester': [7,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



```
#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))
```

[33]

```
        Name      ID  Roll_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'>
```



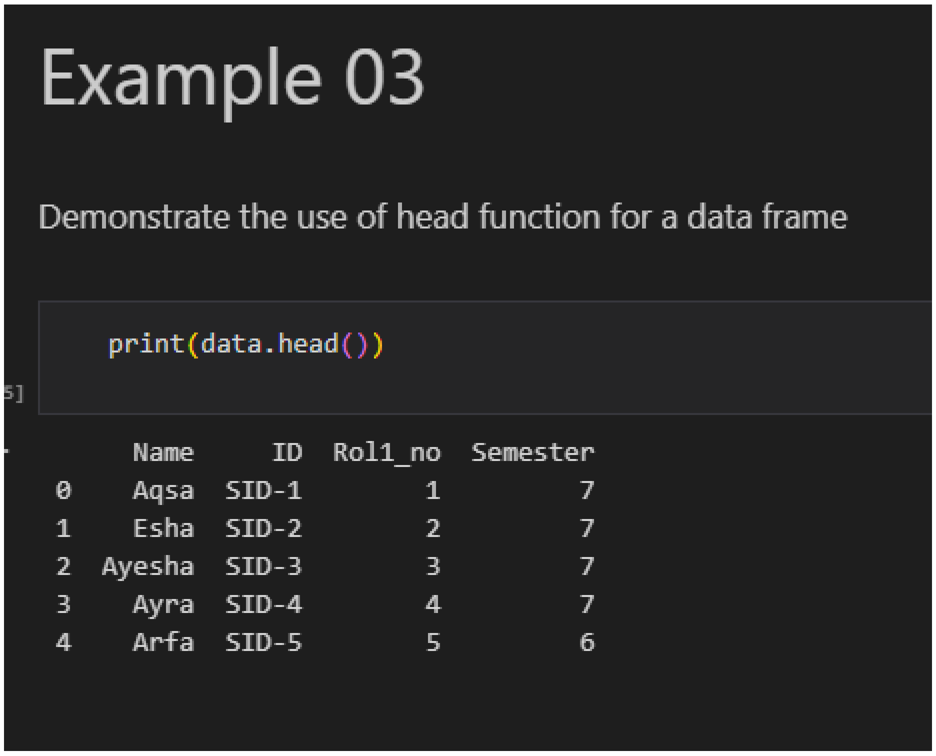
Example 02

Demonstrate the use of describe function for a data frame

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

```
4]
```

```
         Roll_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
```



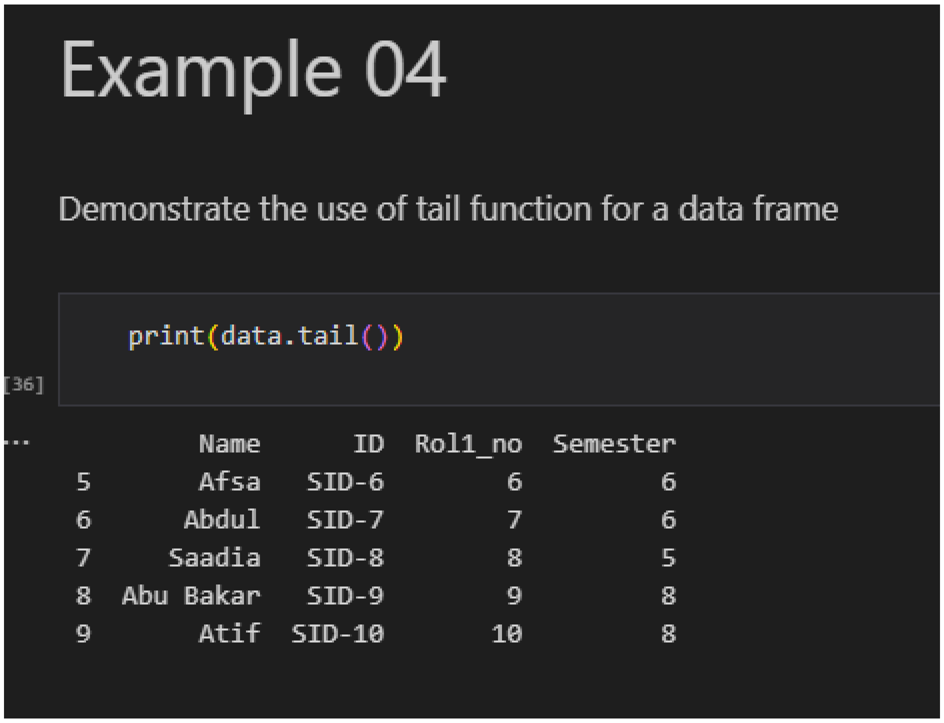
Example 03

Demonstrate the use of head function for a data frame

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

```
5]
```

```
     Name     ID  Roll_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
```



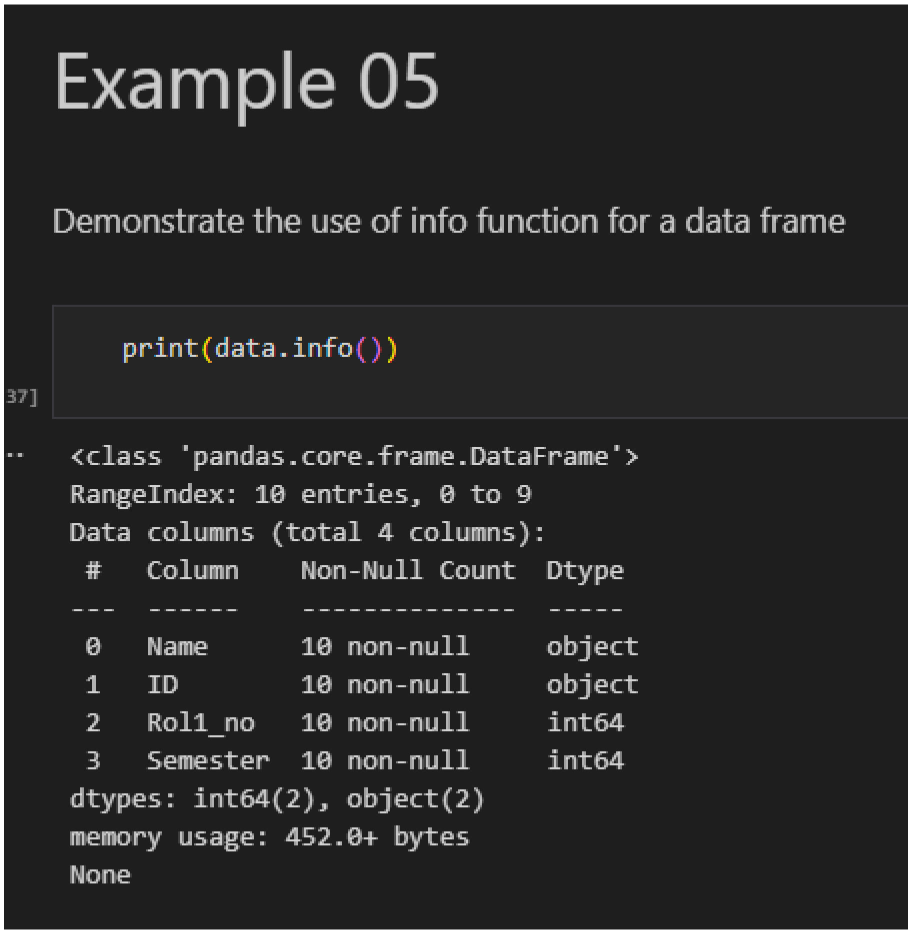
Example 04

Demonstrate the use of tail function for a data frame

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

[36]

```
        Name      ID  Roll_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
```



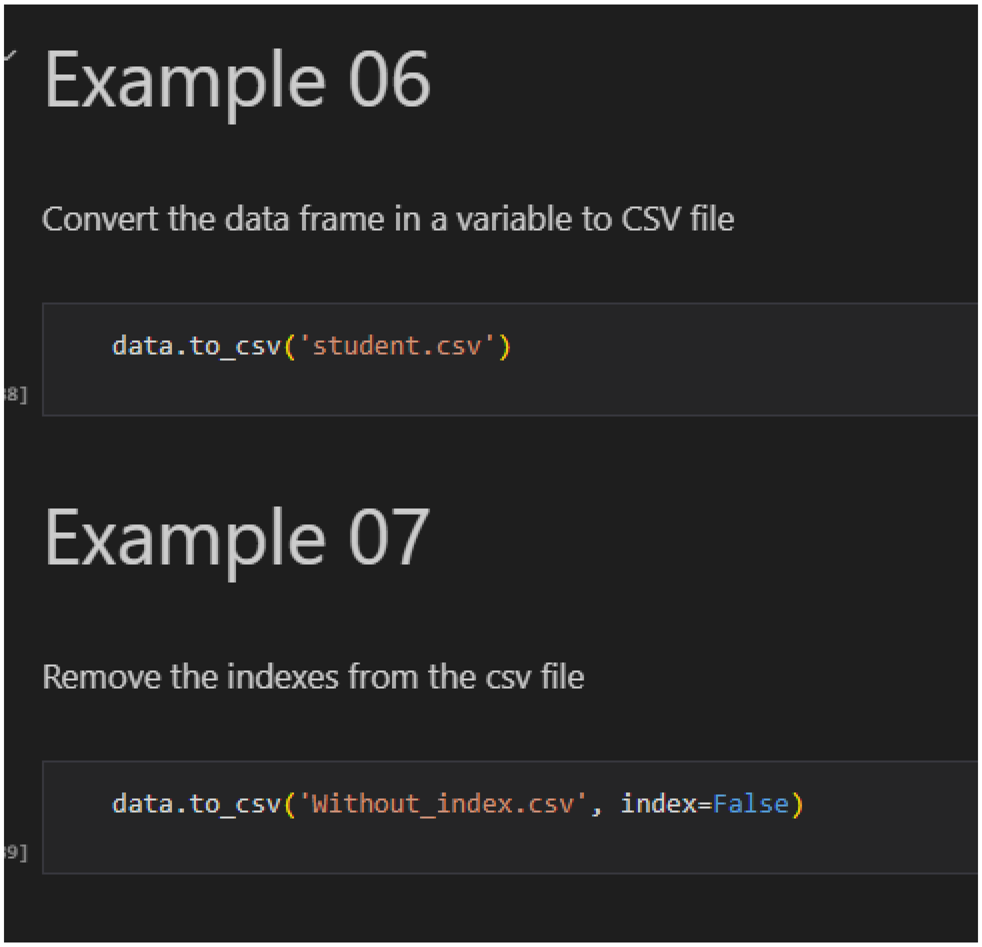
Example 05

Demonstrate the use of info function for a data frame

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

37]

```
<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   Roll_no   10 non-null     int64 
 3   Semester  10 non-null     int64 
dtypes: int64(2), object(2)
memory usage: 452.0+ bytes
None
```



Example 06

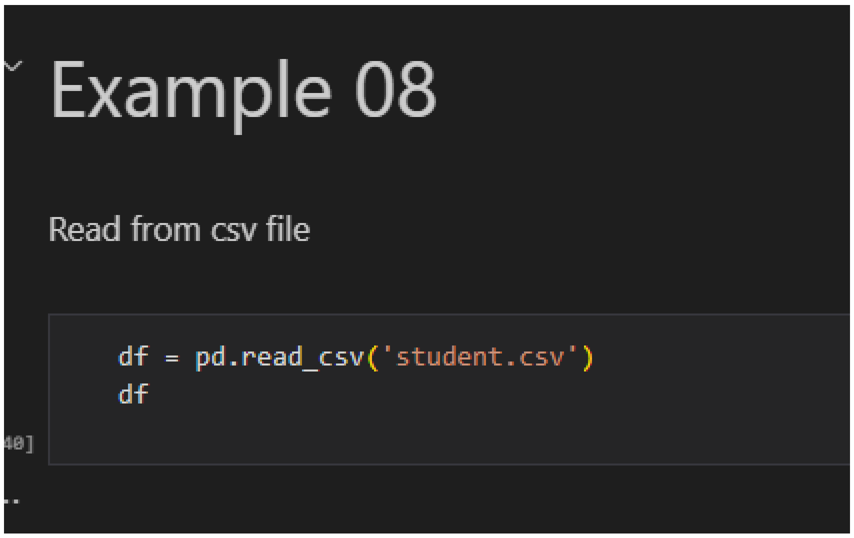
Convert the data frame in a variable to CSV file

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

Example 07

Remove the indexes from the csv file

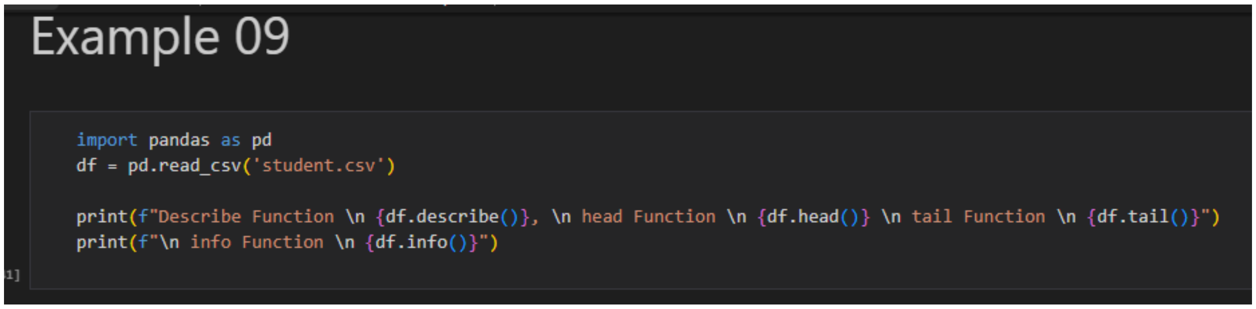
```
data.to_csv('Without_index.csv', index=False)
```



Example 08

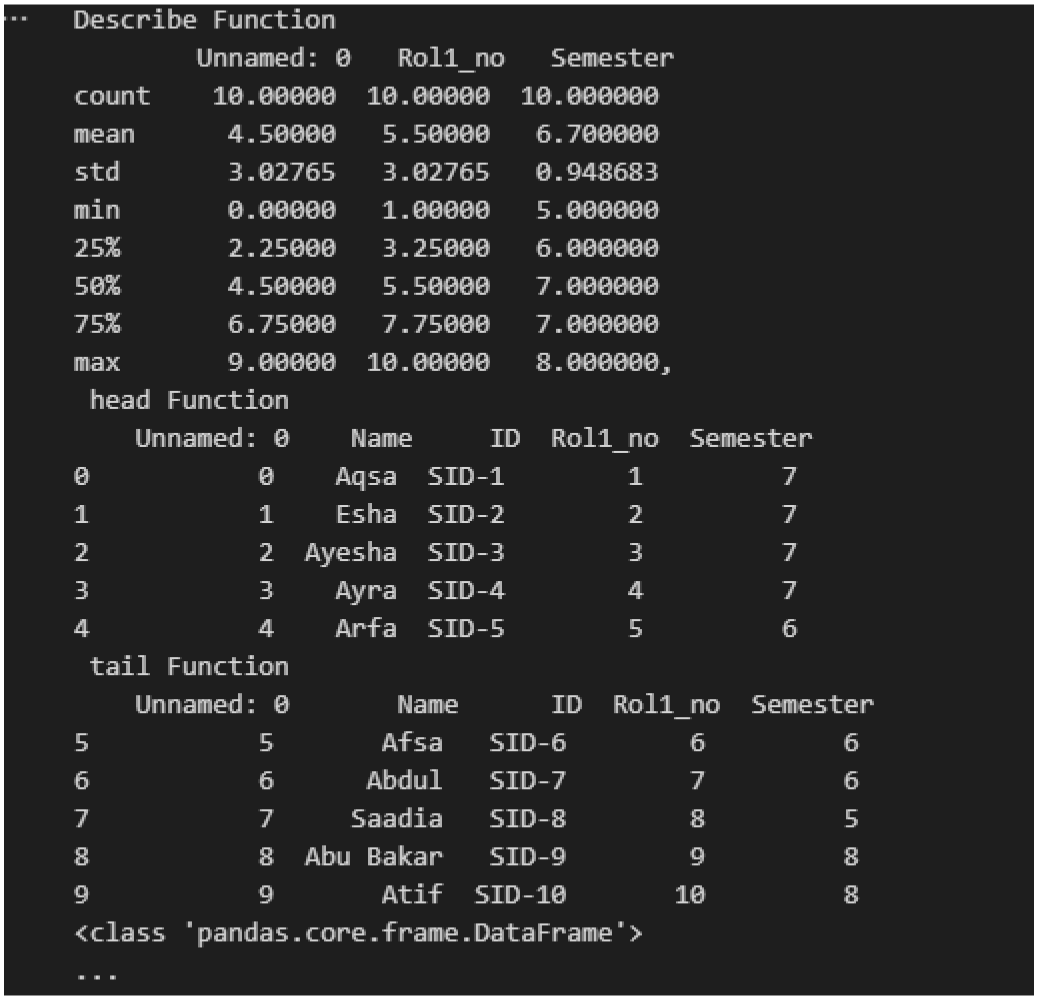
Read from csv file

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

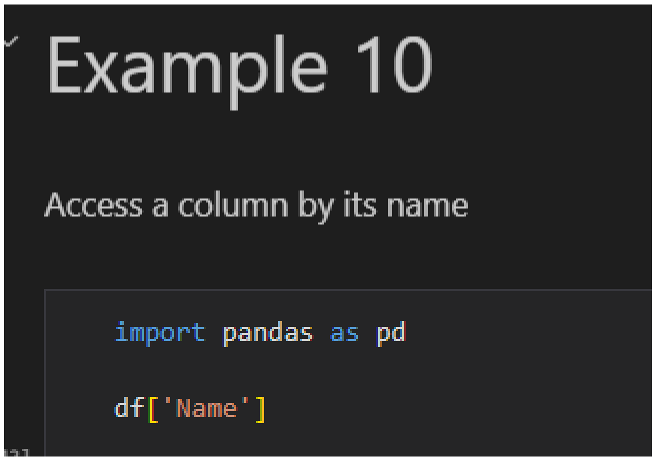


Example 09

```
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()}")
```



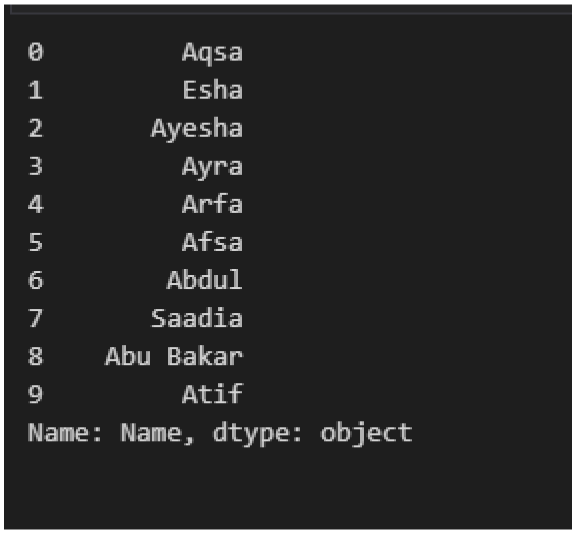
```
Describe Function
        Unnamed: 0   Roll_no   Semester
count    10.00000  10.00000  10.000000
mean      4.50000   5.50000   6.700000
std       3.02765   3.02765   0.948683
min       0.00000   1.00000   5.000000
25%       2.25000   3.25000   6.000000
50%       4.50000   5.50000   7.000000
75%       6.75000   7.75000   7.000000
max       9.00000  10.00000   8.000000,
 head Function
    Unnamed: 0    Name     ID  Roll_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  Roll_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'>
...
```



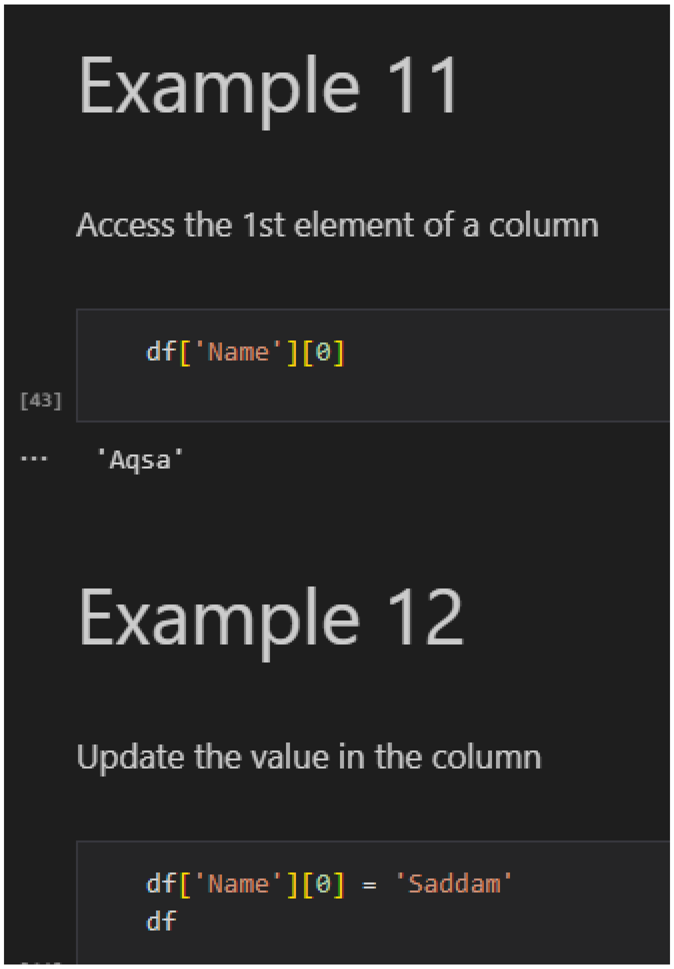
Example 10

Access a column by its name

```python
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'][0]
```

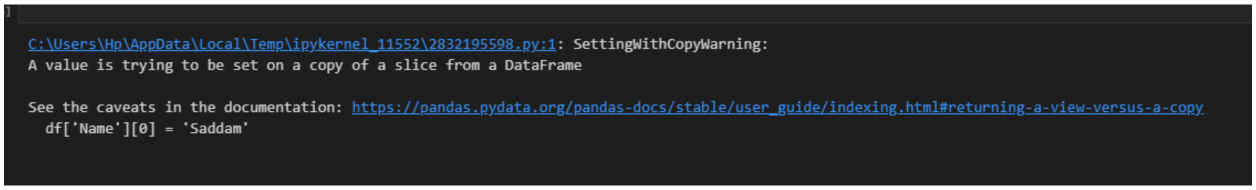
[43]

```
... 'Aqsa'
```

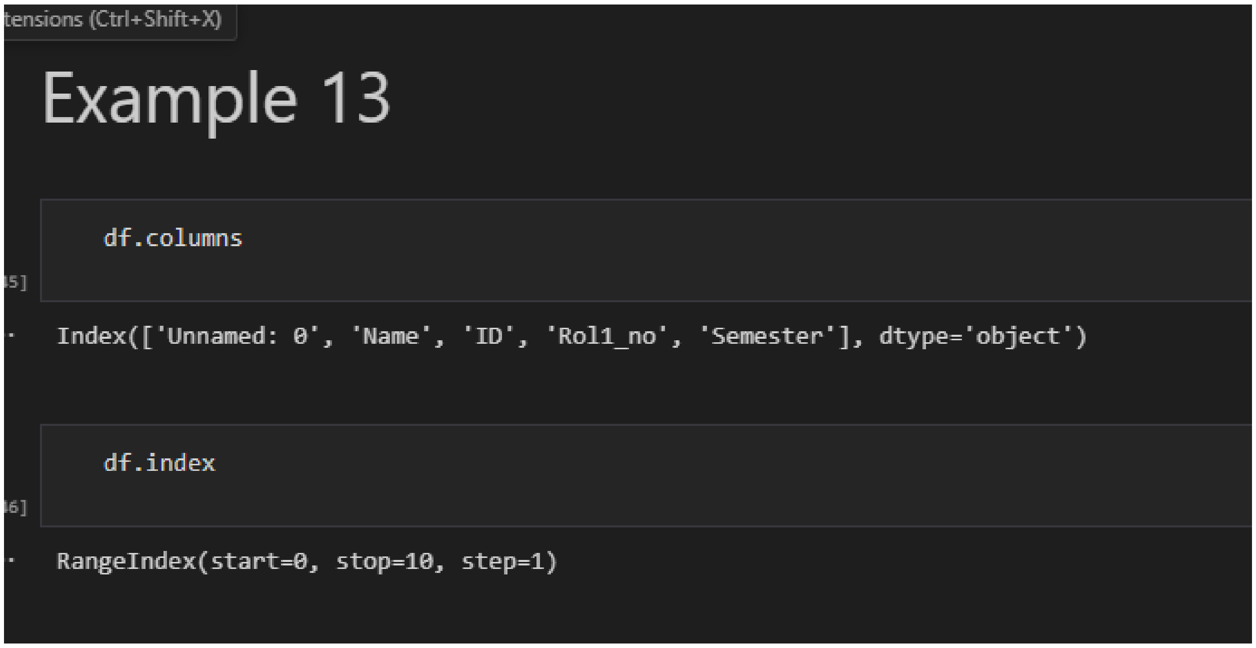
Example 12

Update the value in the column

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



```
C:\Users\Hp\AppData\Local\Temp\ipykernel_11552\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'
```



tensions (Ctrl+Shift+X)

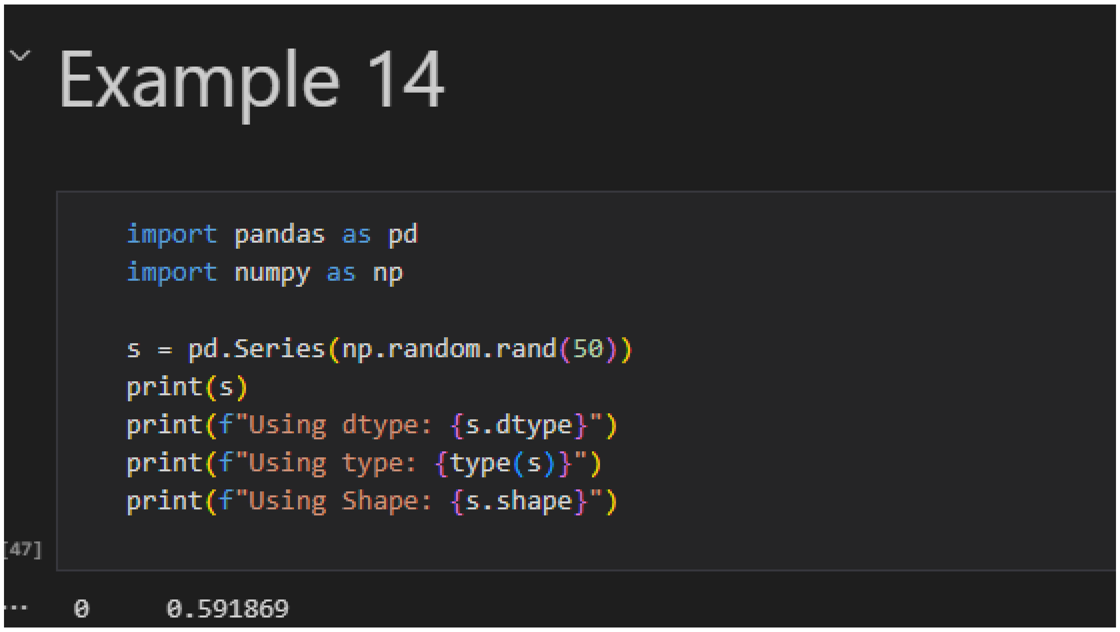
Example 13

```
df.columns
```

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

```
df.index
```

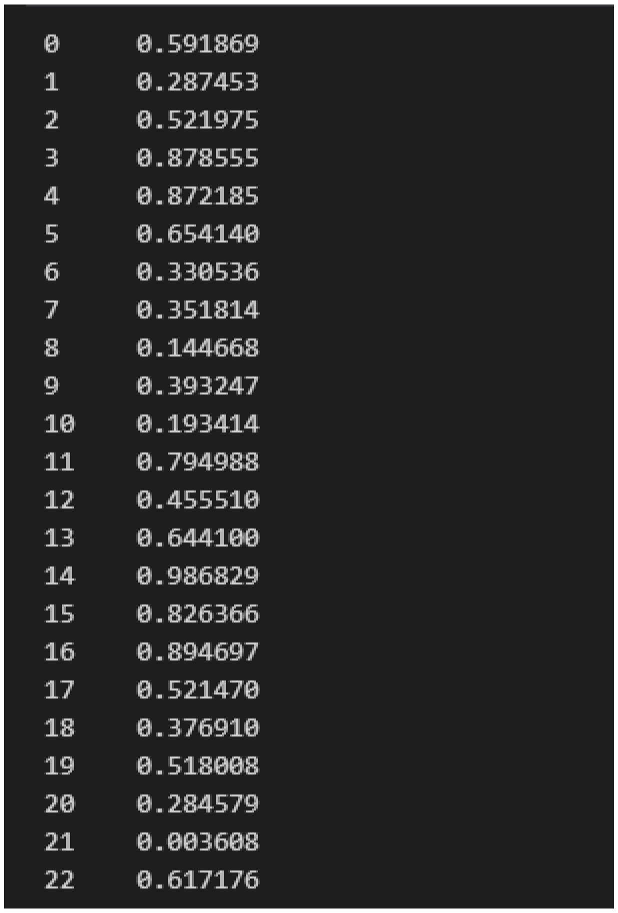
```
RangeIndex(start=0, stop=10, step=1)
```



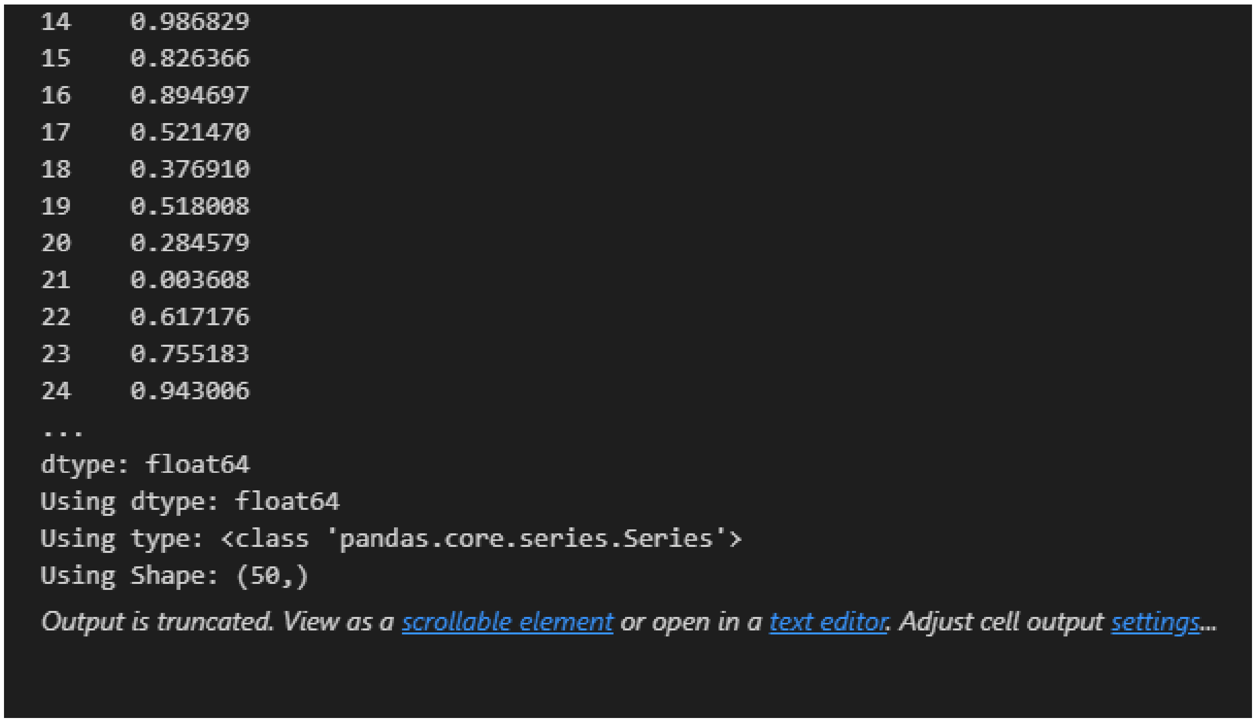
Example 14

```
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      0.591869
```

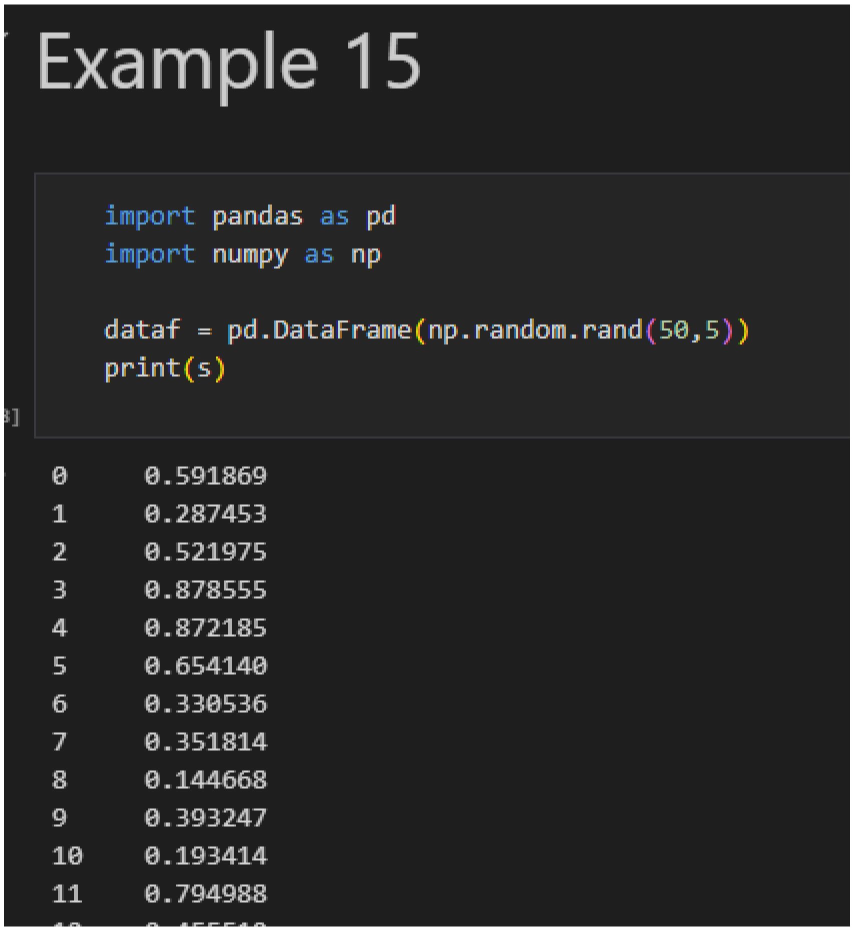


```
0     0.591869
1     0.287453
2     0.521975
3     0.878555
4     0.872185
5     0.654140
6     0.330536
7     0.351814
8     0.144668
9     0.393247
10    0.193414
11    0.794988
12    0.455510
13    0.644100
14    0.986829
15    0.826366
16    0.894697
17    0.521470
18    0.376910
19    0.518008
20    0.284579
21    0.003608
22    0.617176
```



```
14    0.986829
15    0.826366
16    0.894697
17    0.521470
18    0.376910
19    0.518008
20    0.284579
21    0.003608
22    0.617176
23    0.755183
24    0.943006
...
dtype: float64
Using dtype: float64
Using type: <class 'pandas.core.series.Series'>
Using Shape: (50,)
```

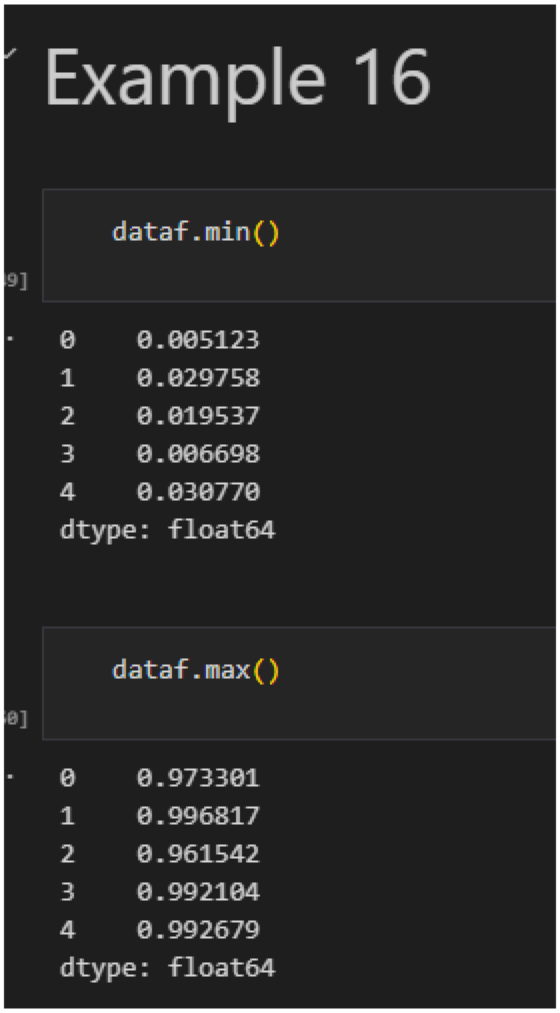
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...



Example 15

```
import pandas as pd
import numpy as np

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

```
0     0.591869
1     0.287453
2     0.521975
3     0.878555
4     0.872185
5     0.654140
6     0.330536
7     0.351814
8     0.144668
9     0.393247
10    0.193414
11    0.794988
```



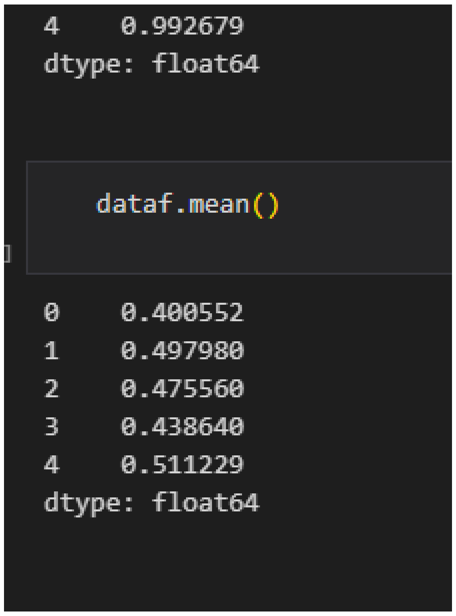
Example 16

```
dataf.min()
```

```
0    0.005123
1    0.029758
2    0.019537
3    0.006698
4    0.030770
dtype: float64
```

```
dataf.max()
```

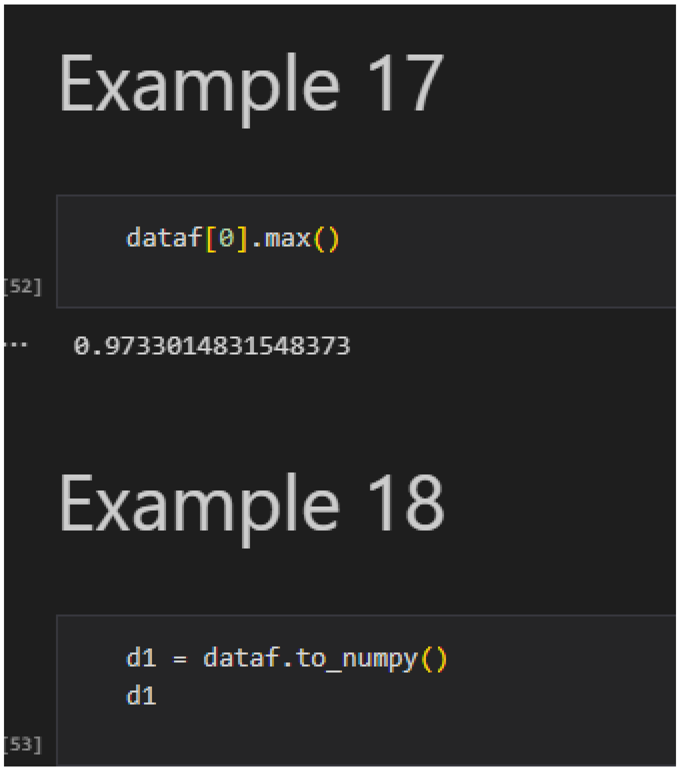
```
0    0.973301
1    0.996817
2    0.961542
3    0.992104
4    0.992679
dtype: float64
```



```
4    0.992679
dtype: float64
```

```
dataf.mean()
```

```
0    0.400552
1    0.497980
2    0.475560
3    0.438640
4    0.511229
dtype: float64
```



Example 17

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

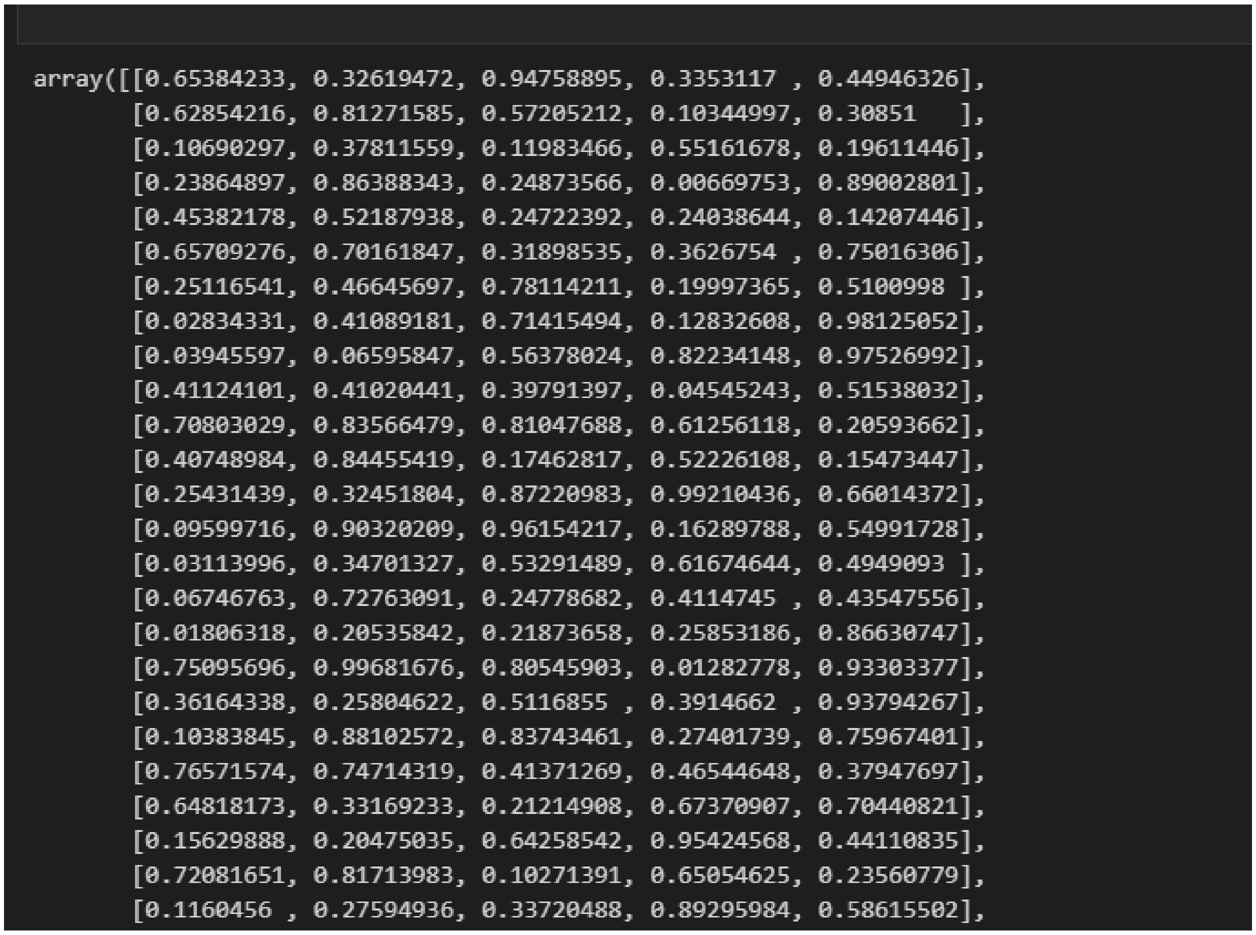
[52]

```
0.9733014831548373
```

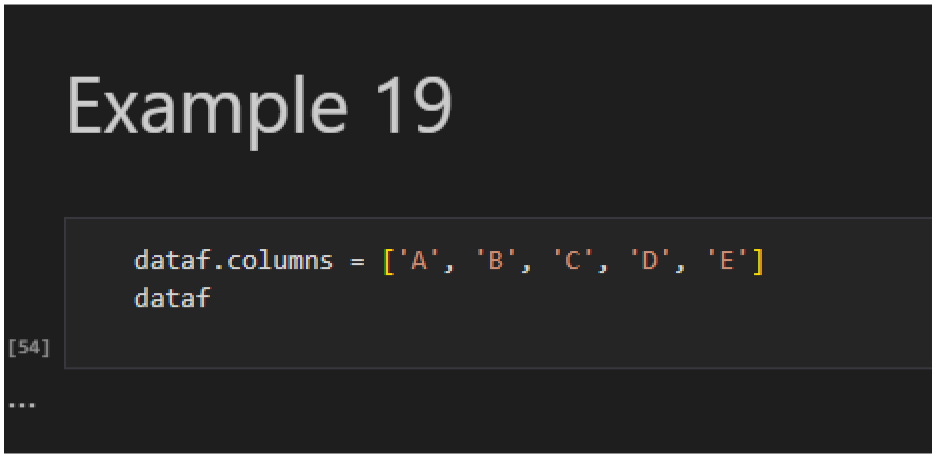
Example 18

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

[53]



```
array([[0.65384233, 0.32619472, 0.94758895, 0.3353117 , 0.44946326],
       [0.62854216, 0.81271585, 0.57205212, 0.10344997, 0.30851   ],
       [0.10690297, 0.37811559, 0.11983466, 0.55161678, 0.19611446],
       [0.23864897, 0.86388343, 0.24873566, 0.00669753, 0.89002801],
       [0.45382178, 0.52187938, 0.24722392, 0.24038644, 0.14207446],
       [0.65709276, 0.70161847, 0.31898535, 0.3626754 , 0.75016306],
       [0.25116541, 0.46645697, 0.78114211, 0.19997365, 0.5100998 ],
       [0.02834331, 0.41089181, 0.71415494, 0.12832608, 0.98125052],
       [0.03945597, 0.06595847, 0.56378024, 0.82234148, 0.97526992],
       [0.41124101, 0.41020441, 0.39791397, 0.04545243, 0.51538032],
       [0.70803029, 0.83566479, 0.81047688, 0.61256118, 0.20593662],
       [0.40748984, 0.84455419, 0.17462817, 0.52226108, 0.15473447],
       [0.25431439, 0.32451804, 0.87220983, 0.99210436, 0.66014372],
       [0.09599716, 0.90320209, 0.96154217, 0.16289788, 0.54991728],
       [0.03113996, 0.34701327, 0.53291489, 0.61674644, 0.4949093 ],
       [0.06746763, 0.72763091, 0.24778682, 0.4114745 , 0.43547556],
       [0.01806318, 0.20535842, 0.21873658, 0.25853186, 0.86630747],
       [0.75095696, 0.99681676, 0.80545903, 0.01282778, 0.93303377],
       [0.36164338, 0.25804622, 0.5116855 , 0.3914662 , 0.93794267],
       [0.10383845, 0.88102572, 0.83743461, 0.27401739, 0.75967401],
       [0.76571574, 0.74714319, 0.41371269, 0.46544648, 0.37947697],
       [0.64818173, 0.33169233, 0.21214908, 0.67370907, 0.70440821],
       [0.15629888, 0.20475035, 0.64258542, 0.95424568, 0.44110835],
       [0.72081651, 0.81713983, 0.10271391, 0.65054625, 0.23560779],
       [0.1160456 , 0.27594936, 0.33720488, 0.89295984, 0.58615502],
```

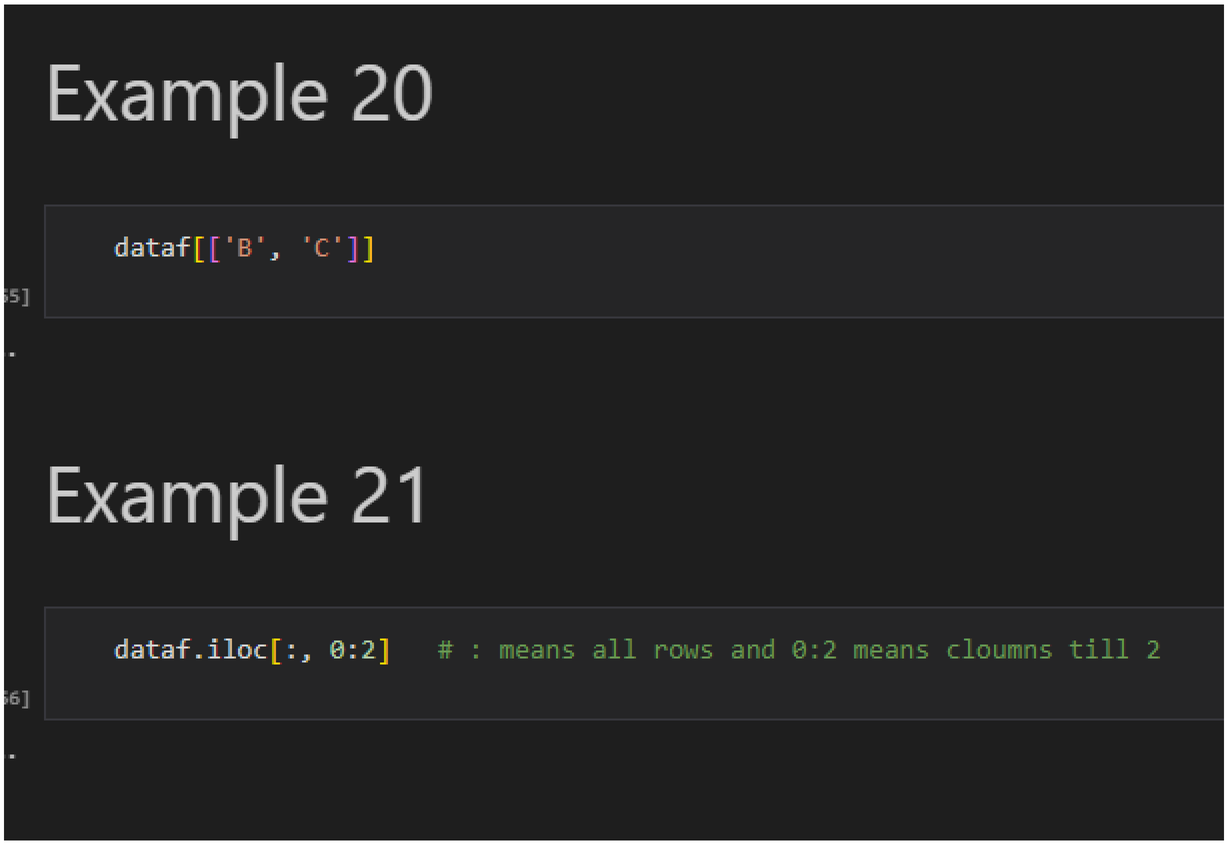


Example 19

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

[54]

...

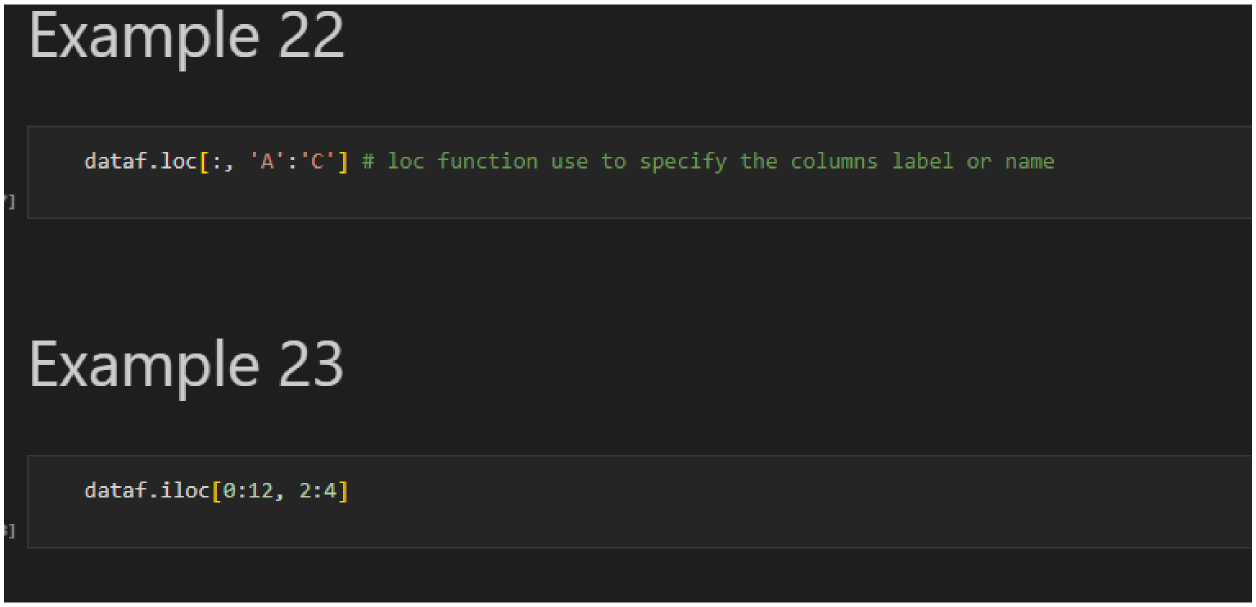


Example 20

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

Example 21

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



Example 22

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

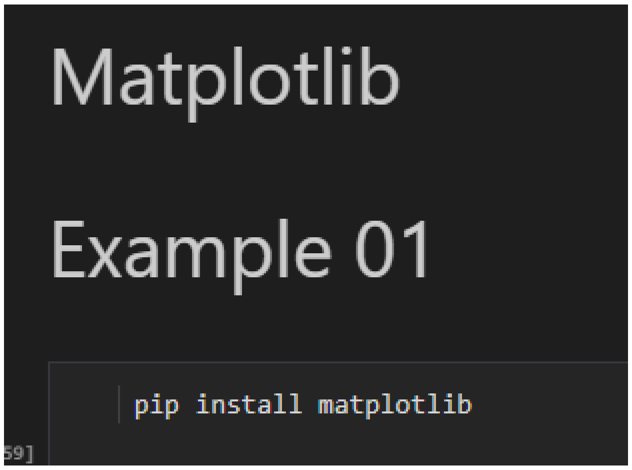
Example 23

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

3. **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 → link (Click here).

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

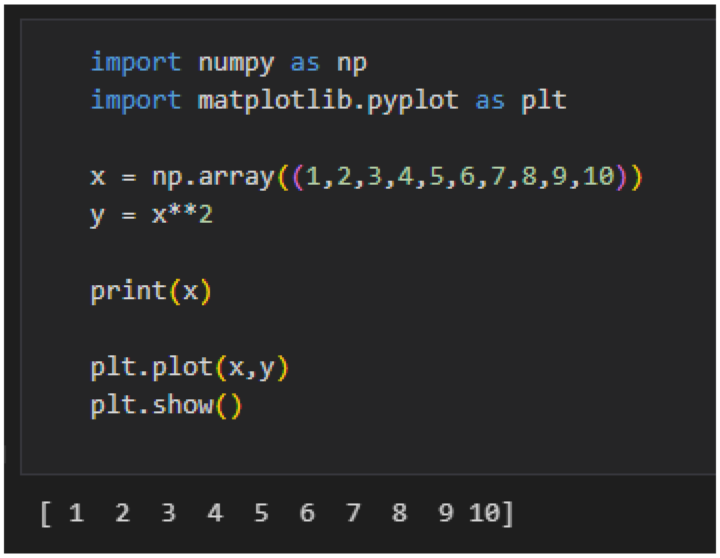
Marks	1	2	3	4
Coding	The code is not as per guidelines and requirements are not met	Some section of code is correct	Most section of code is correct and understands it well	The code is properly written, and have good understanding about it



Matplotlib

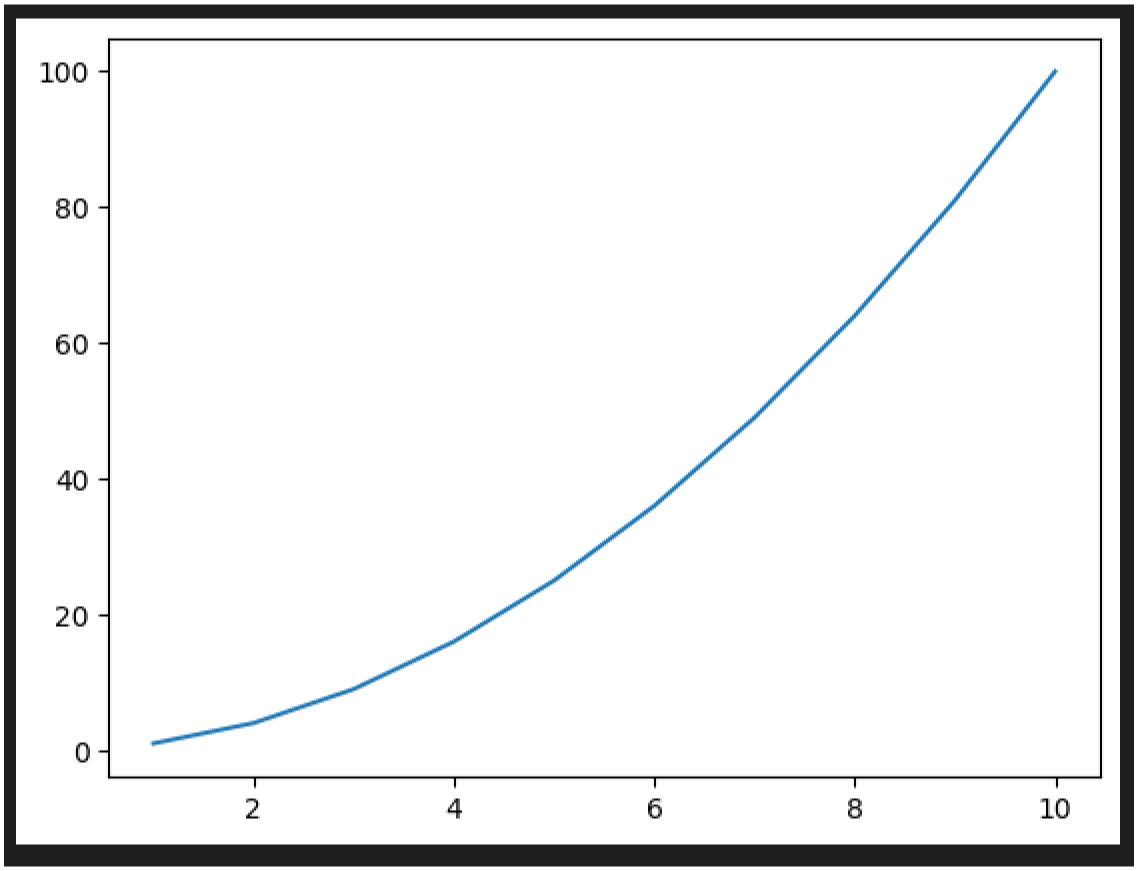
Example 01

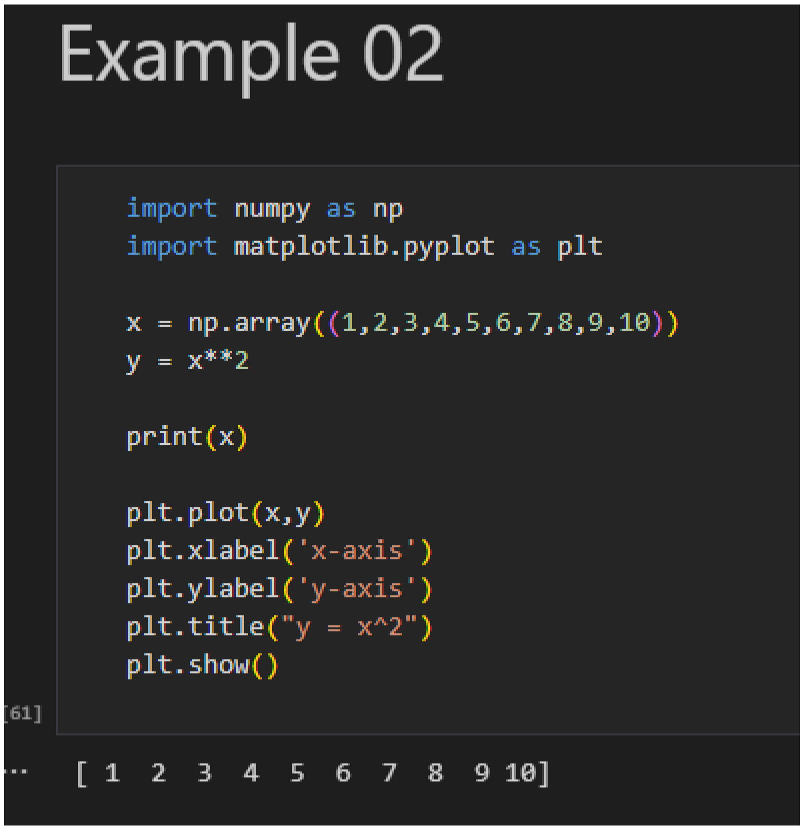
```
pip install matplotlib
```



```
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]
```



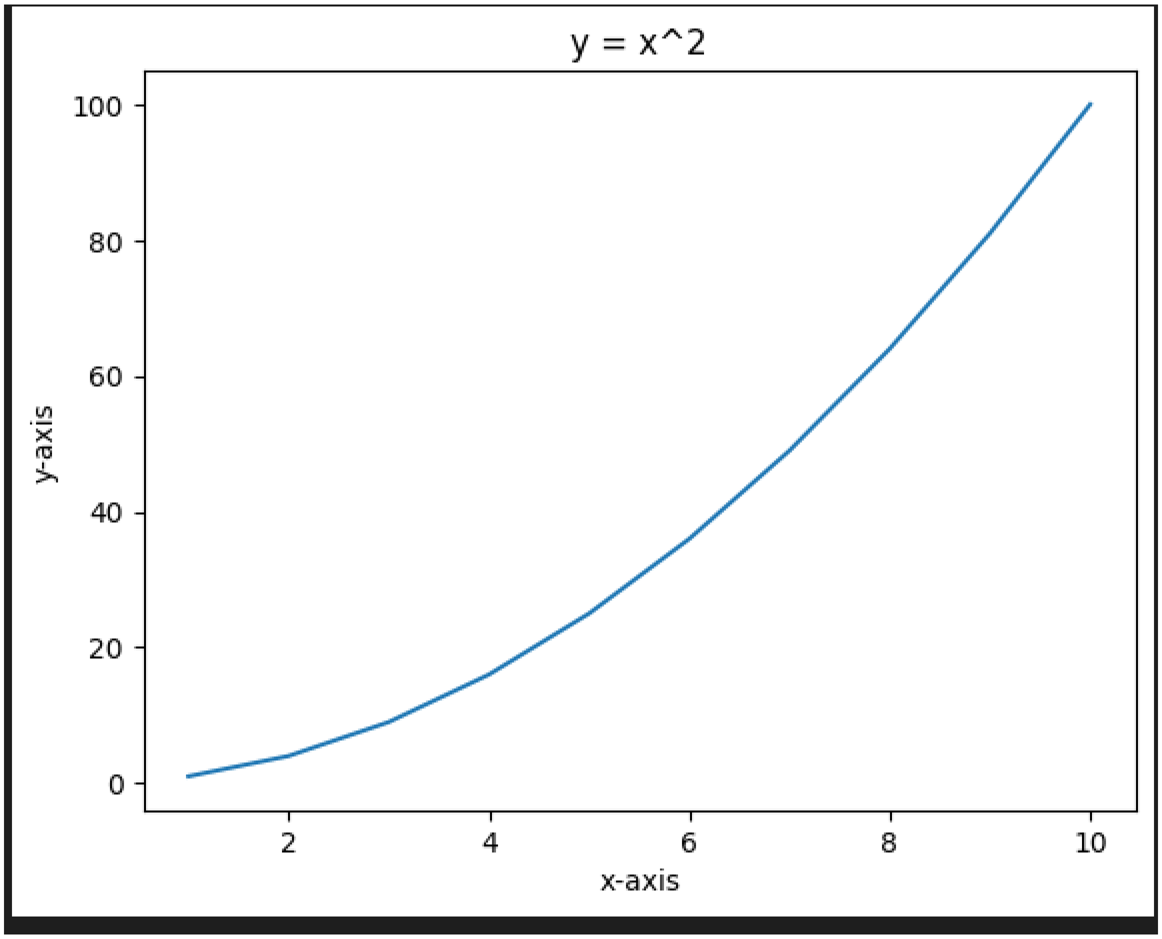


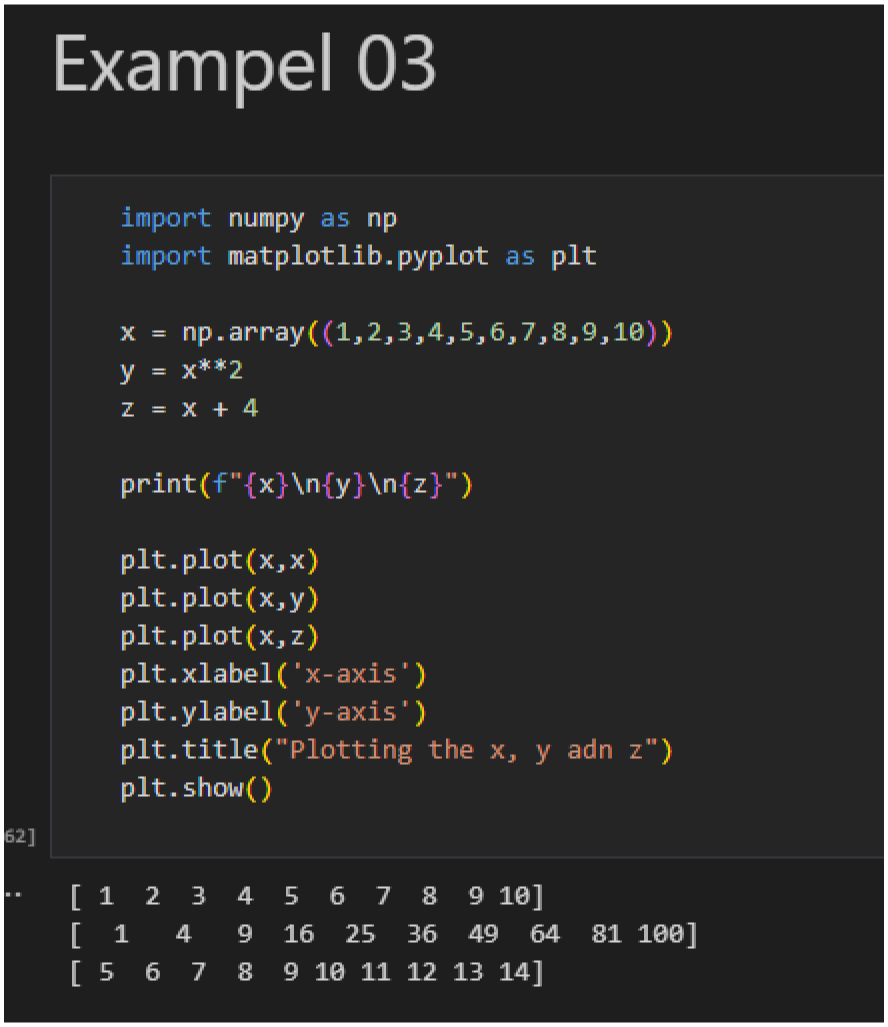
Example 02

```
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()
```

[61]

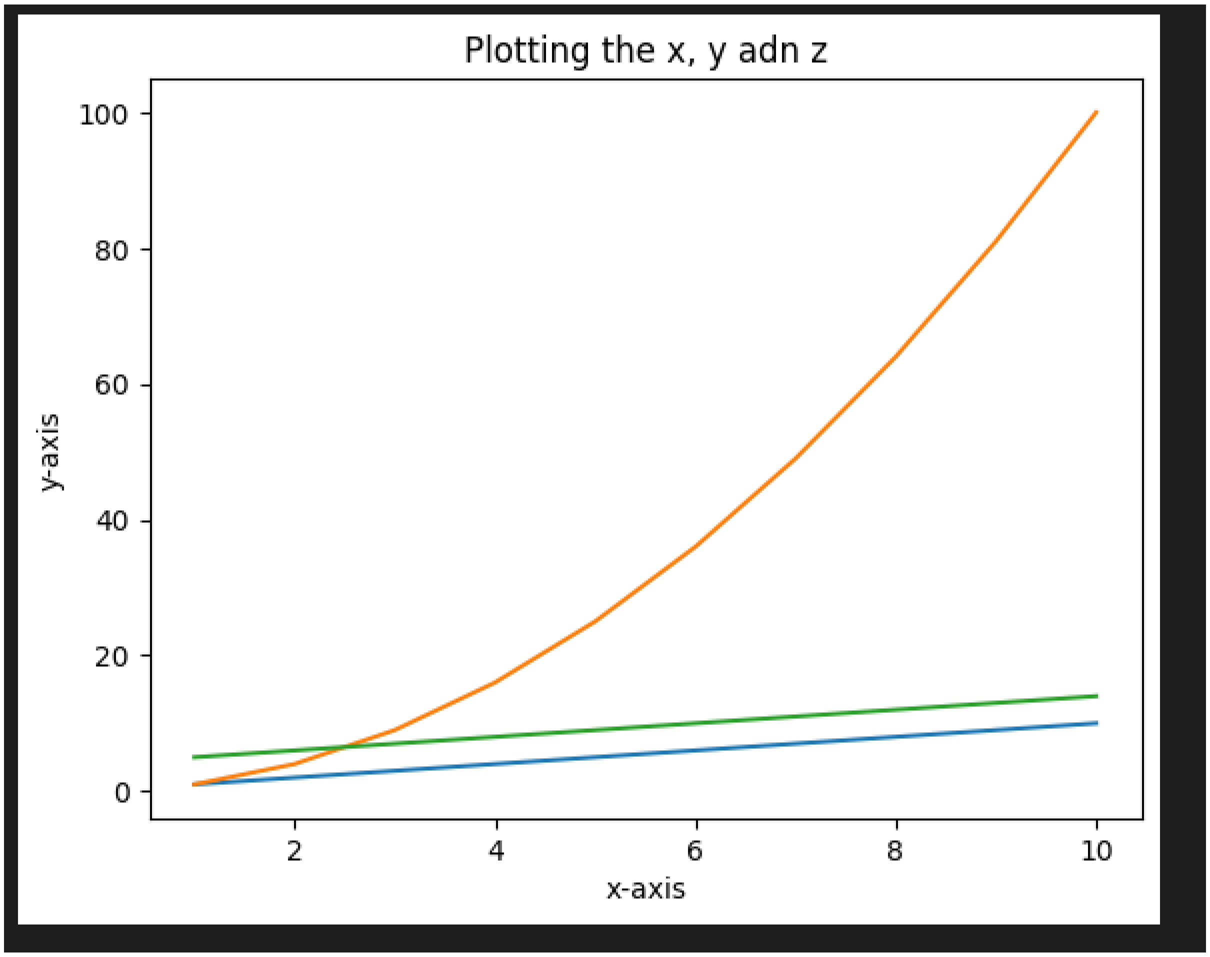
```
[ 1  2  3  4  5  6  7  8  9 10]
```

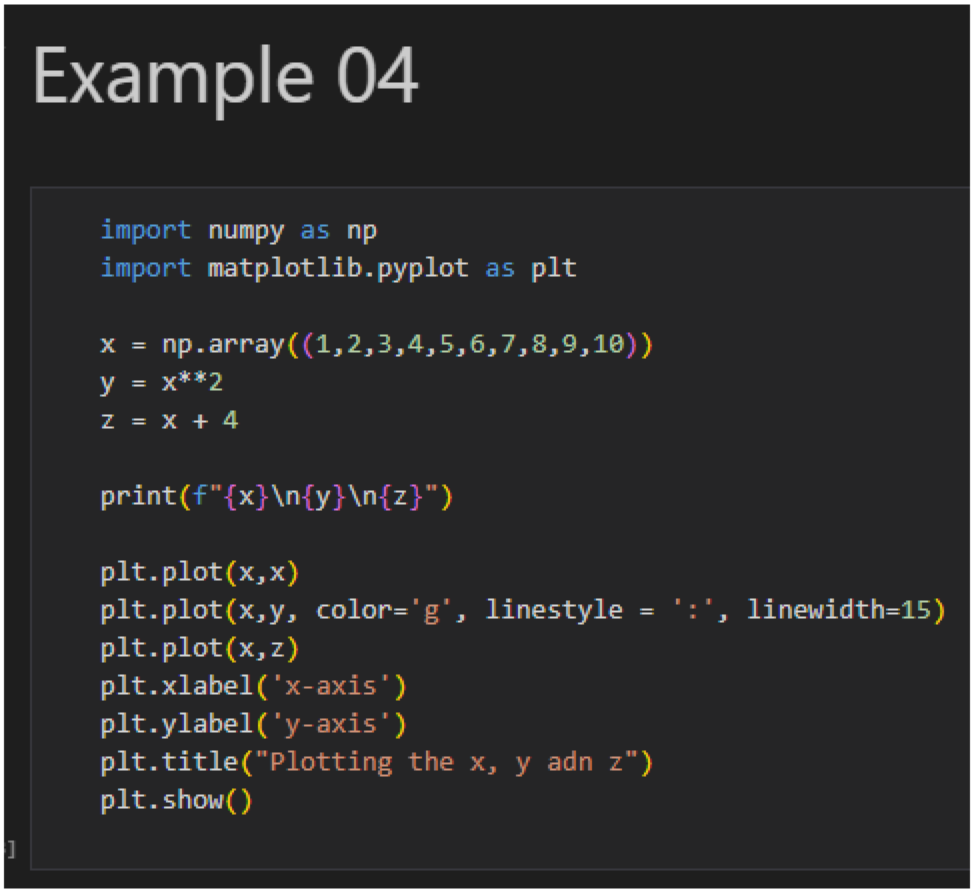

y = x^2
y-axis
x-axis
0
20
40
60
80
100
2
4
6
8
10


Exampel 03

```
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()
```

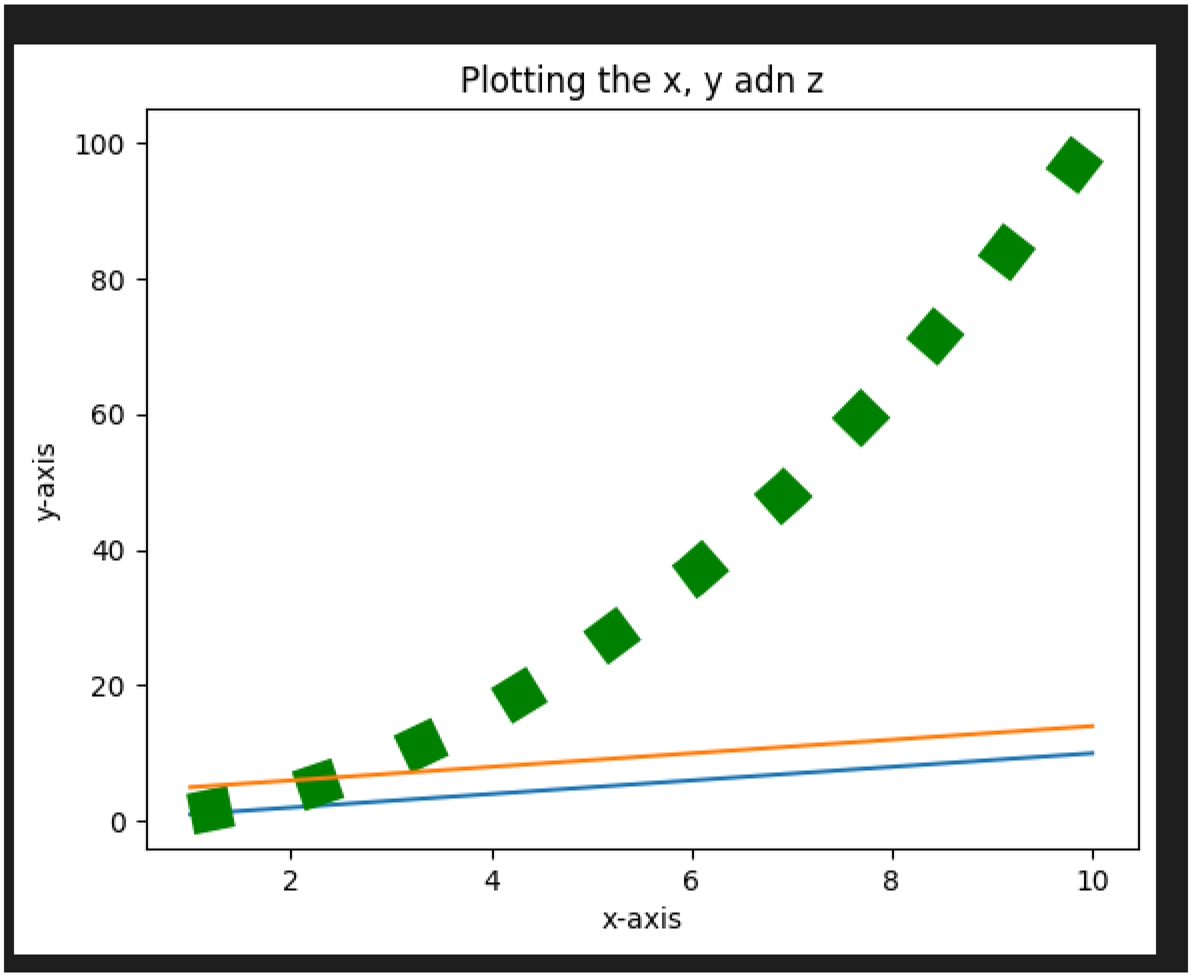
```
[ 1  2  3  4  5  6  7  8  9 10]
[  1   4   9  16  25  36  49  64  81 100]
[ 5  6  7  8  9 10 11 12 13 14]
```

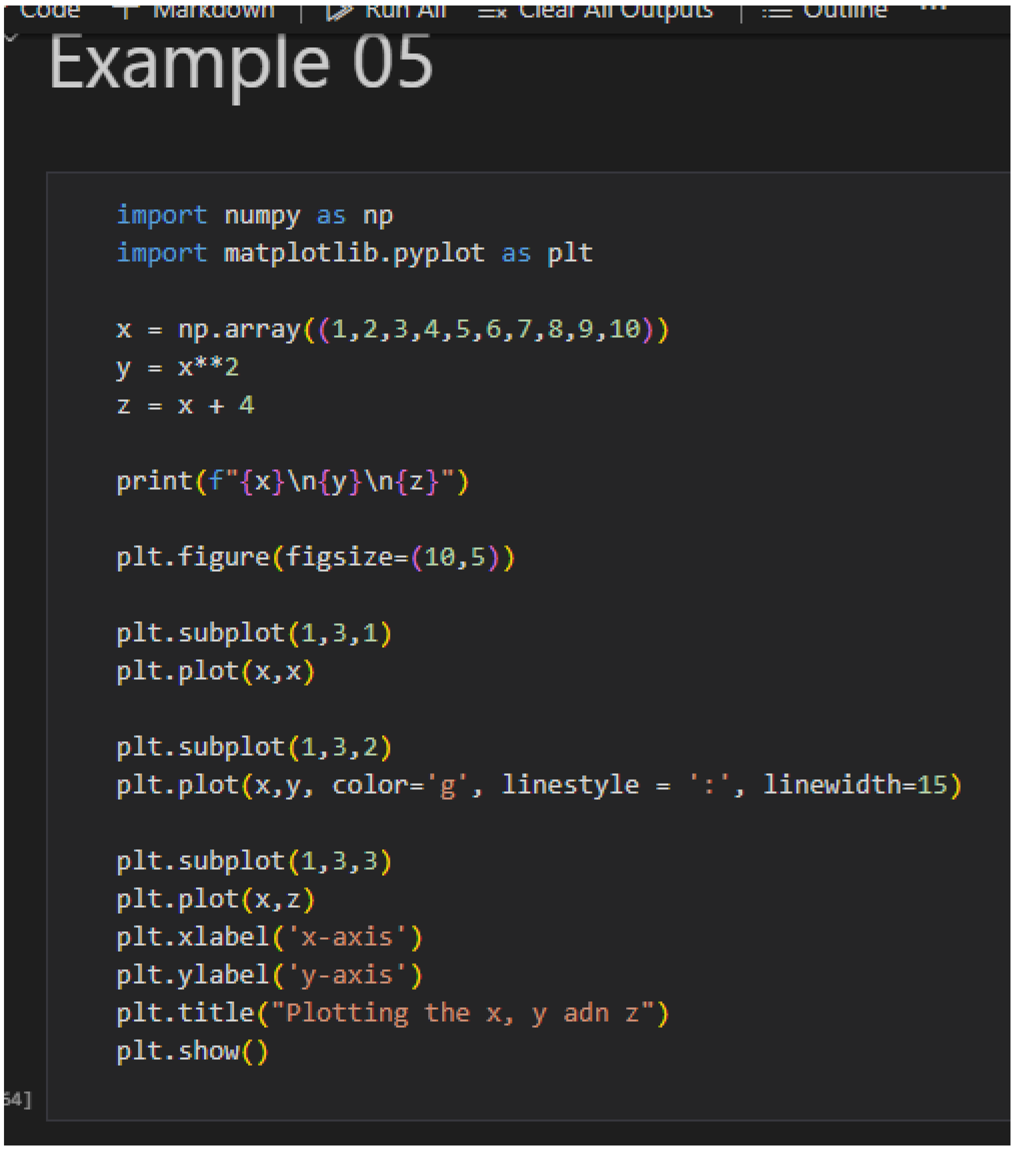

Plotting the x, y adn z
y-axis
x-axis
100
80
60
40
20
0
2
4
6
8
10



Example 04

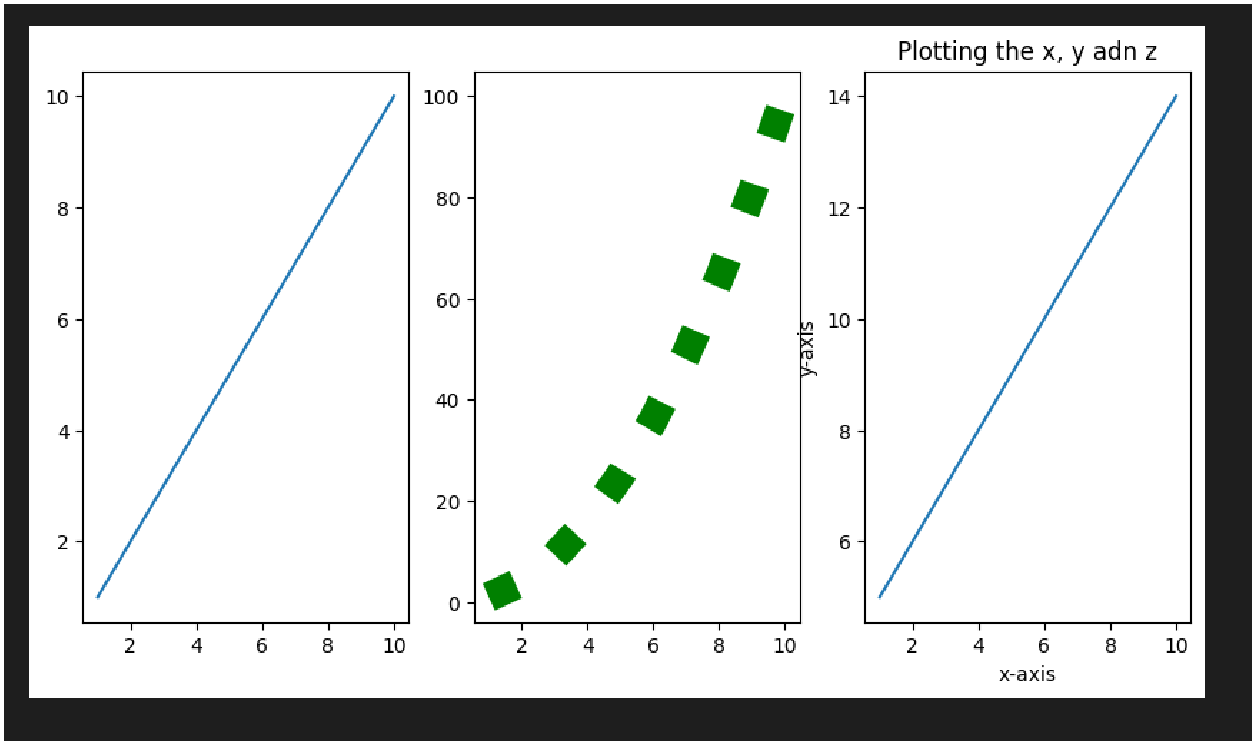
```
import numpy as np
import matplotlib.pyplot as plt

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()
```


Plotting the x, y adn z
y-axis
x-axis
0
20
40
60
80
100
2
4
6
8
10

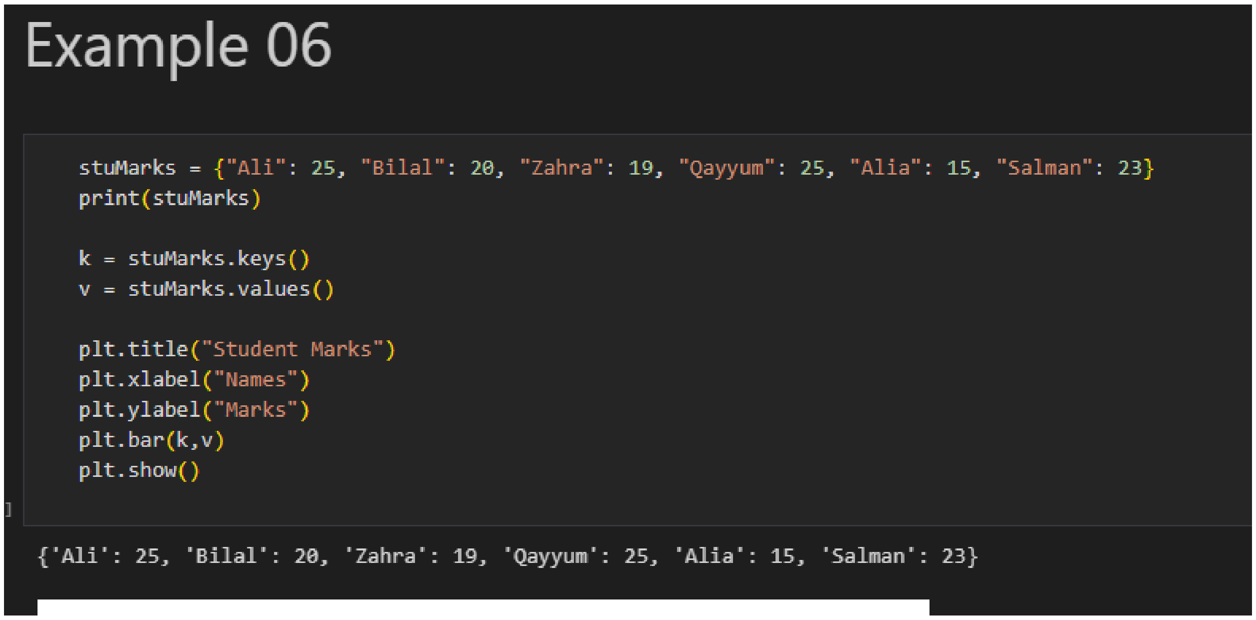


Example 05

```python
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()
```

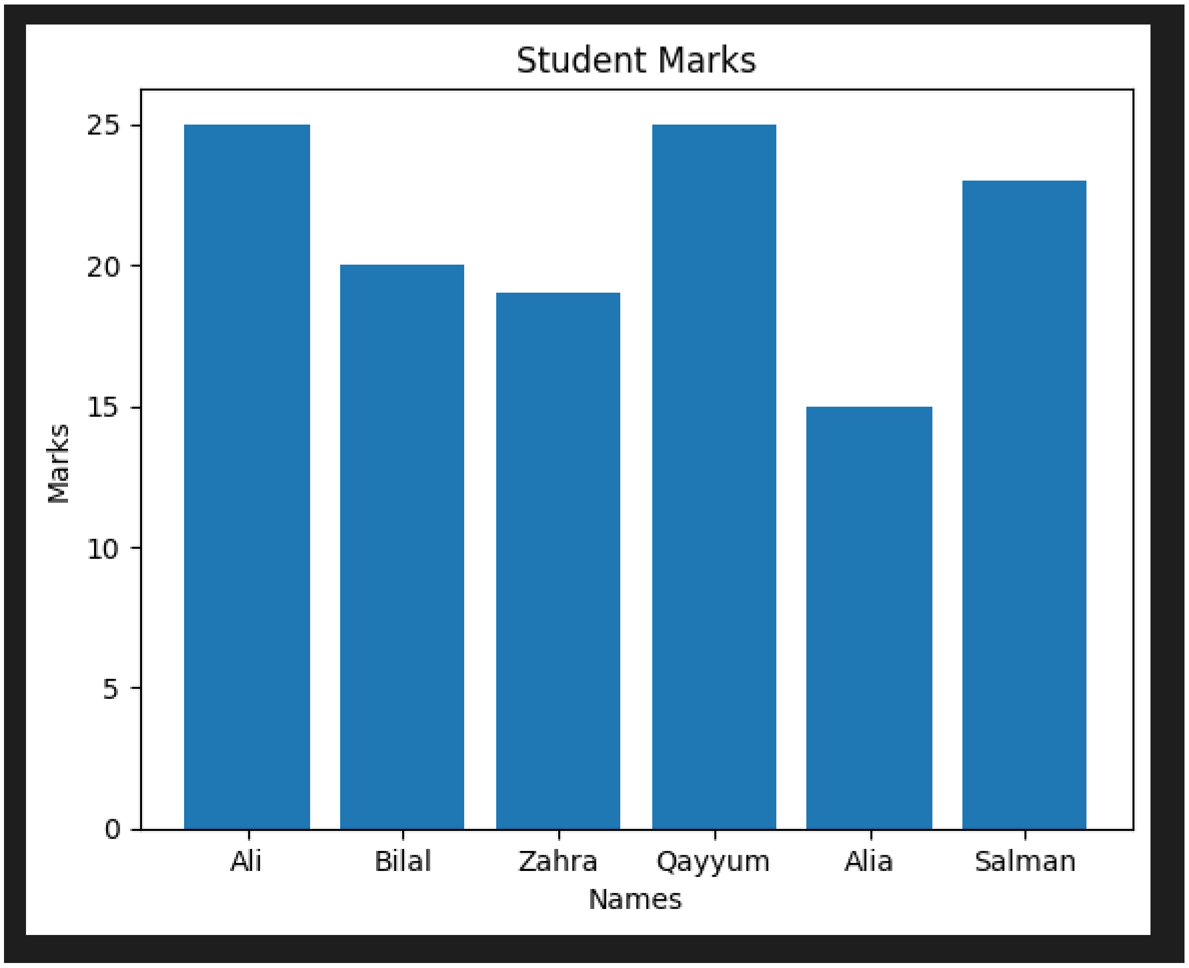


Example 06

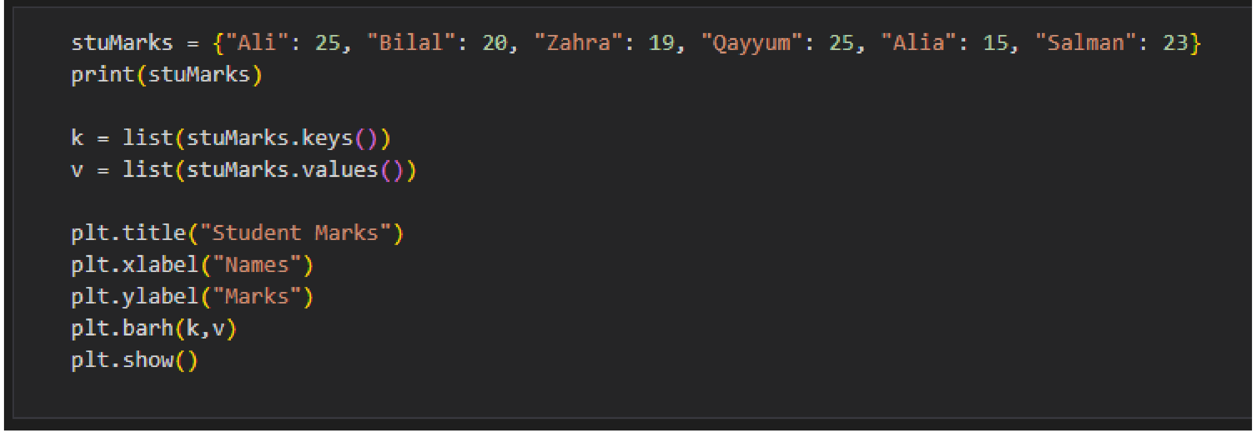


```
stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

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

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

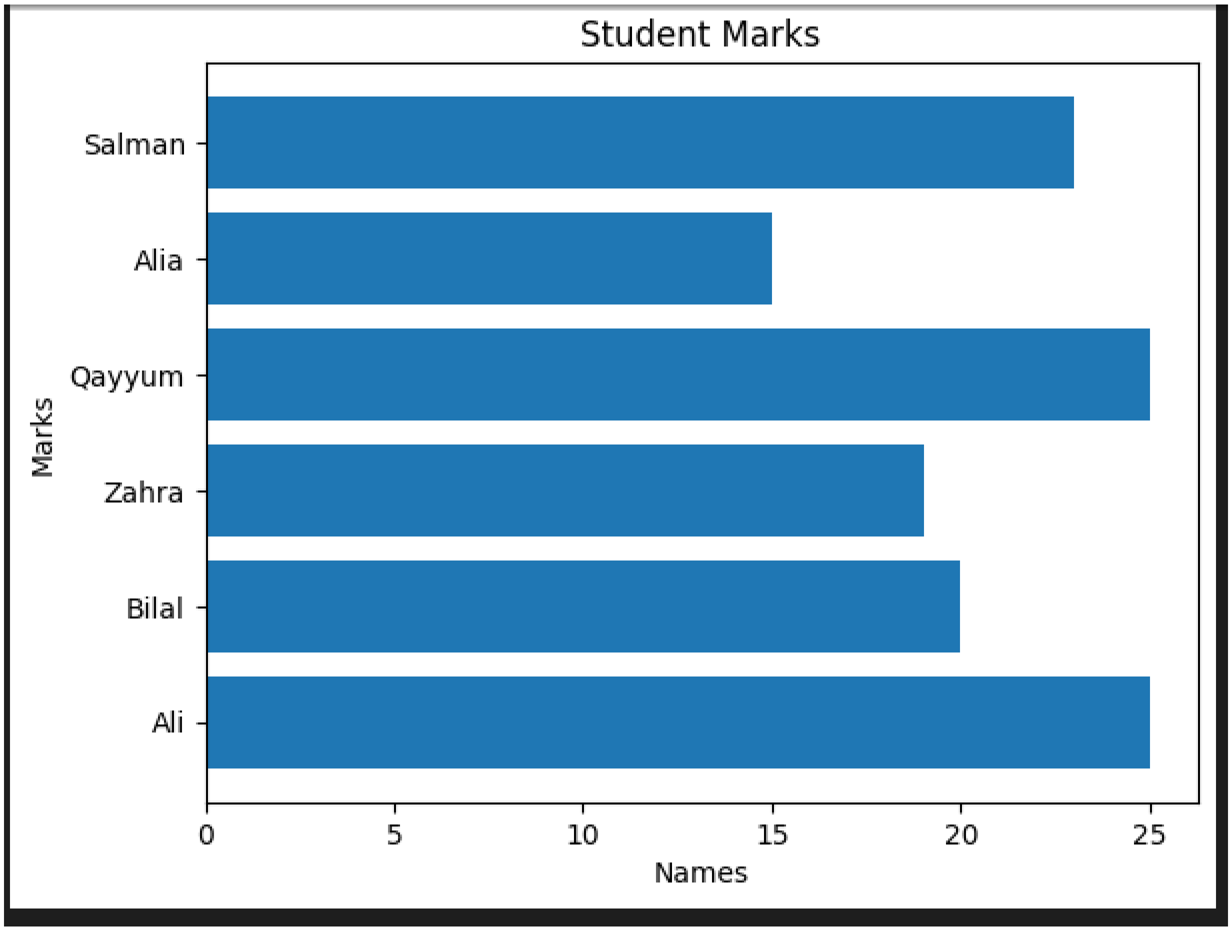
```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```

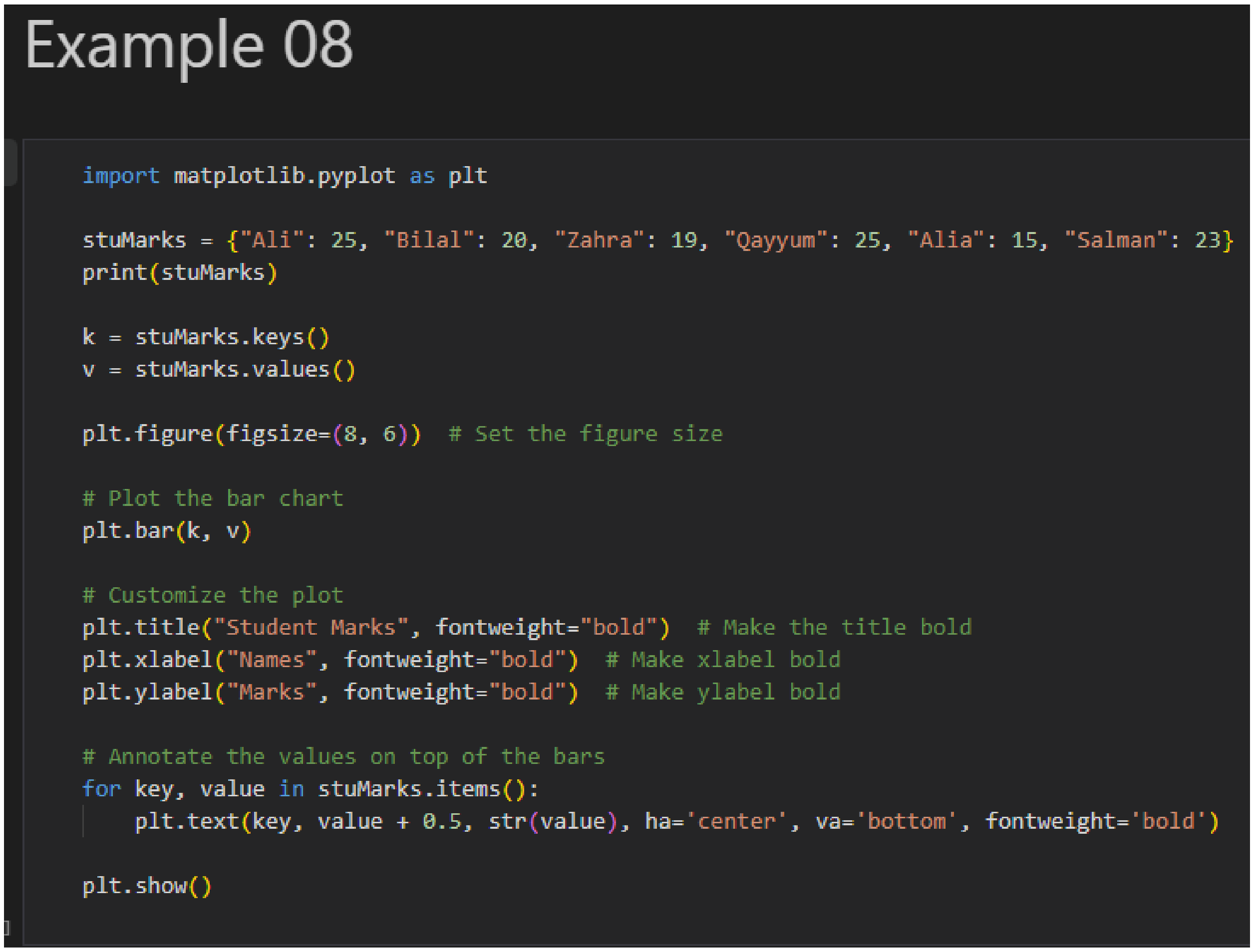


Example 07



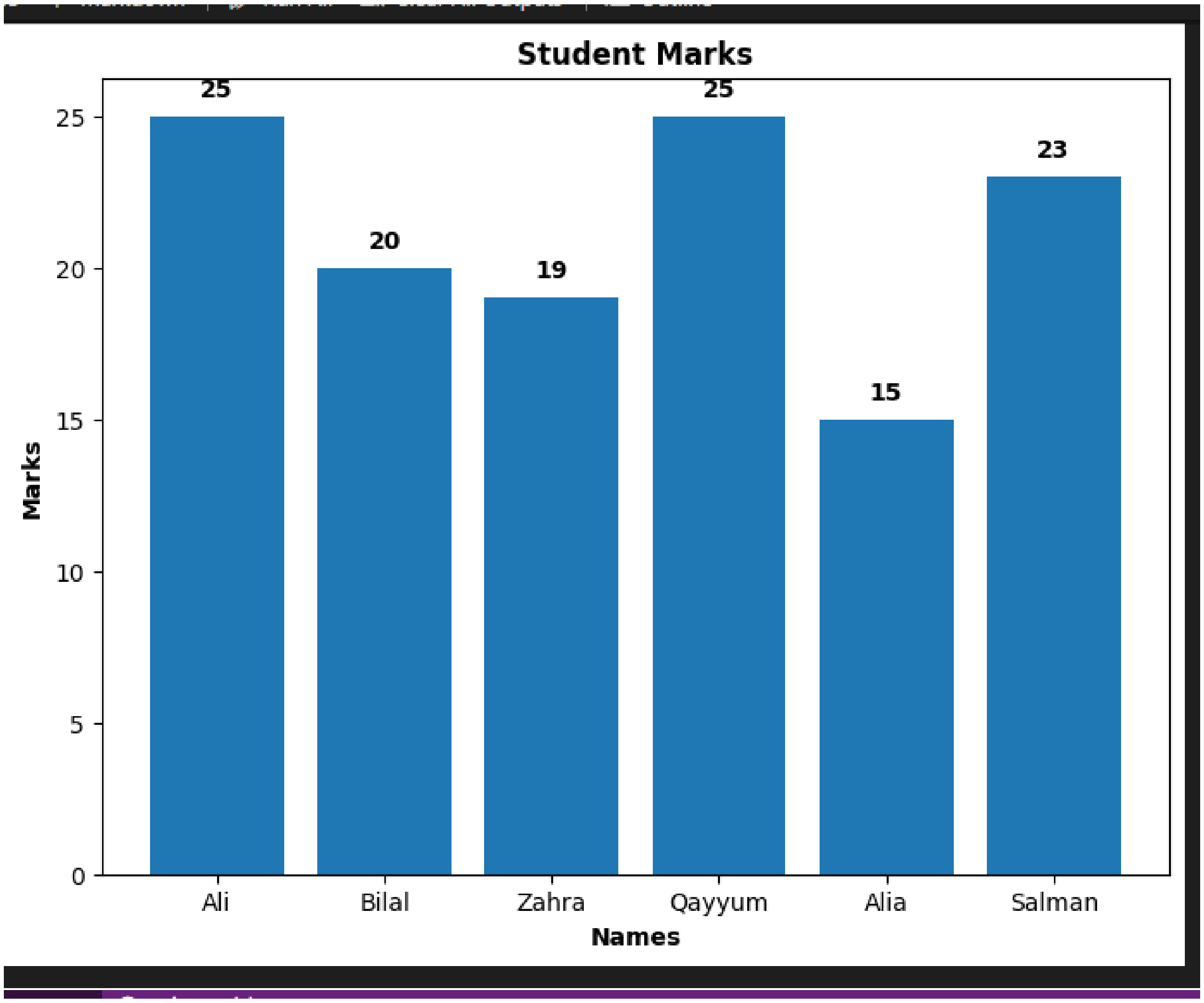
```python
stuMarks = {"Ali": 25, "Bilal": 20, "Zahra": 19, "Qayyum": 25, "Alia": 15, "Salman": 23}
print(stuMarks)

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

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

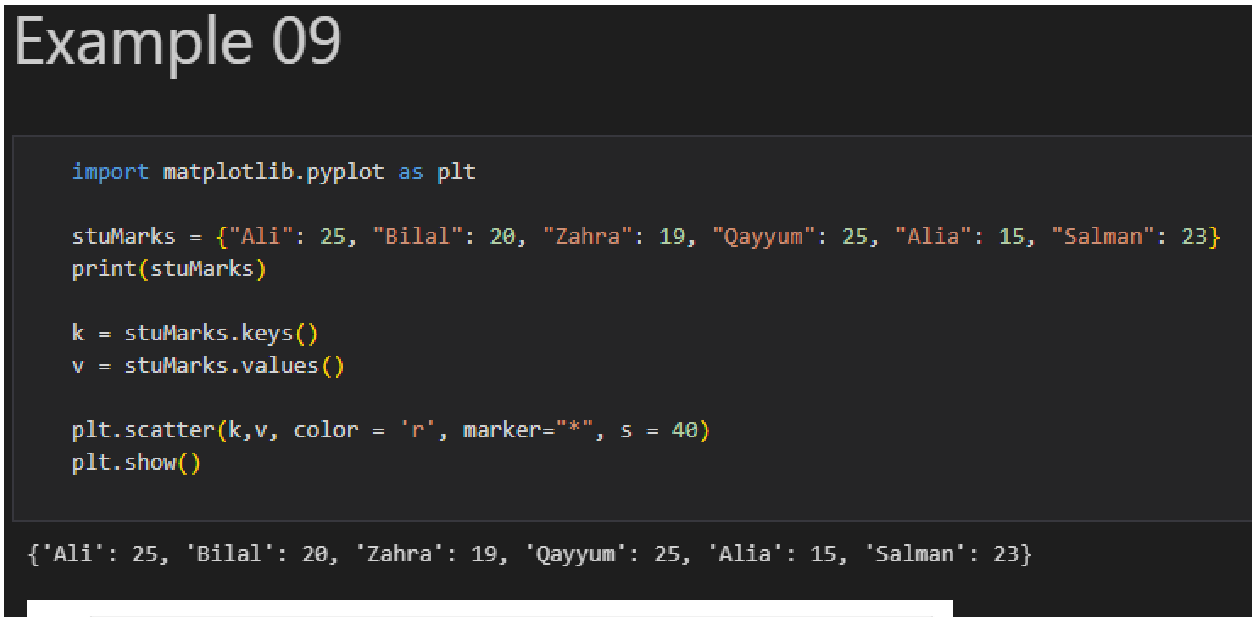

Student Marks
Salman
Alia
Qayyum
Zahra
Bilal
Ali
Marks
0
5
10
15
20
25
Names



Example 08

```python
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()
```

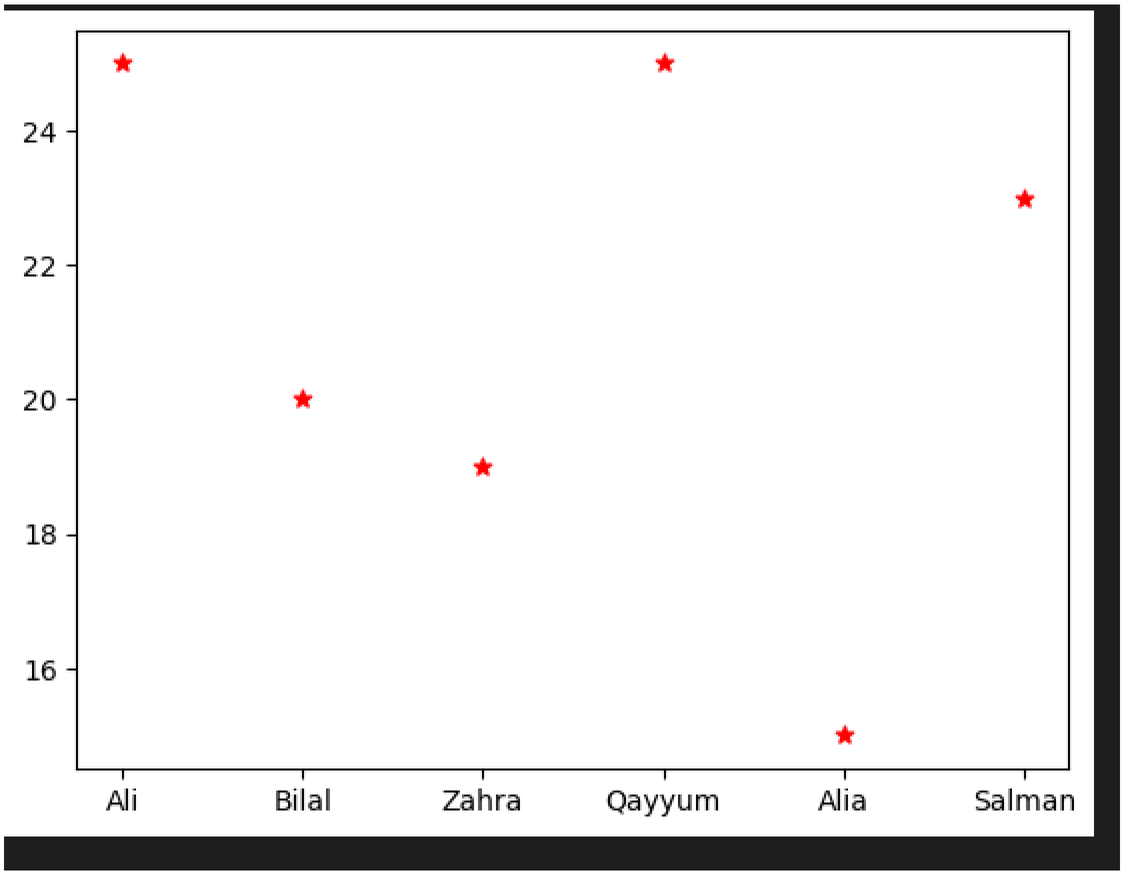


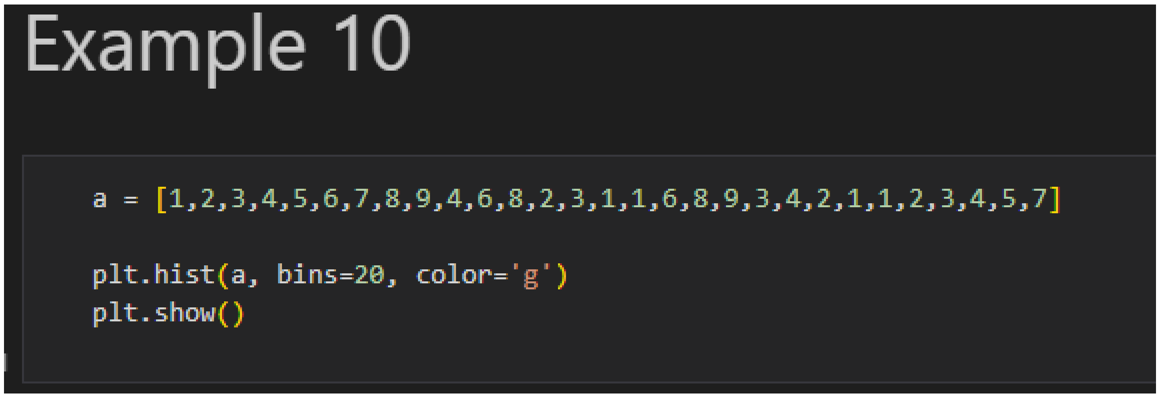


Example 09

```
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()
```

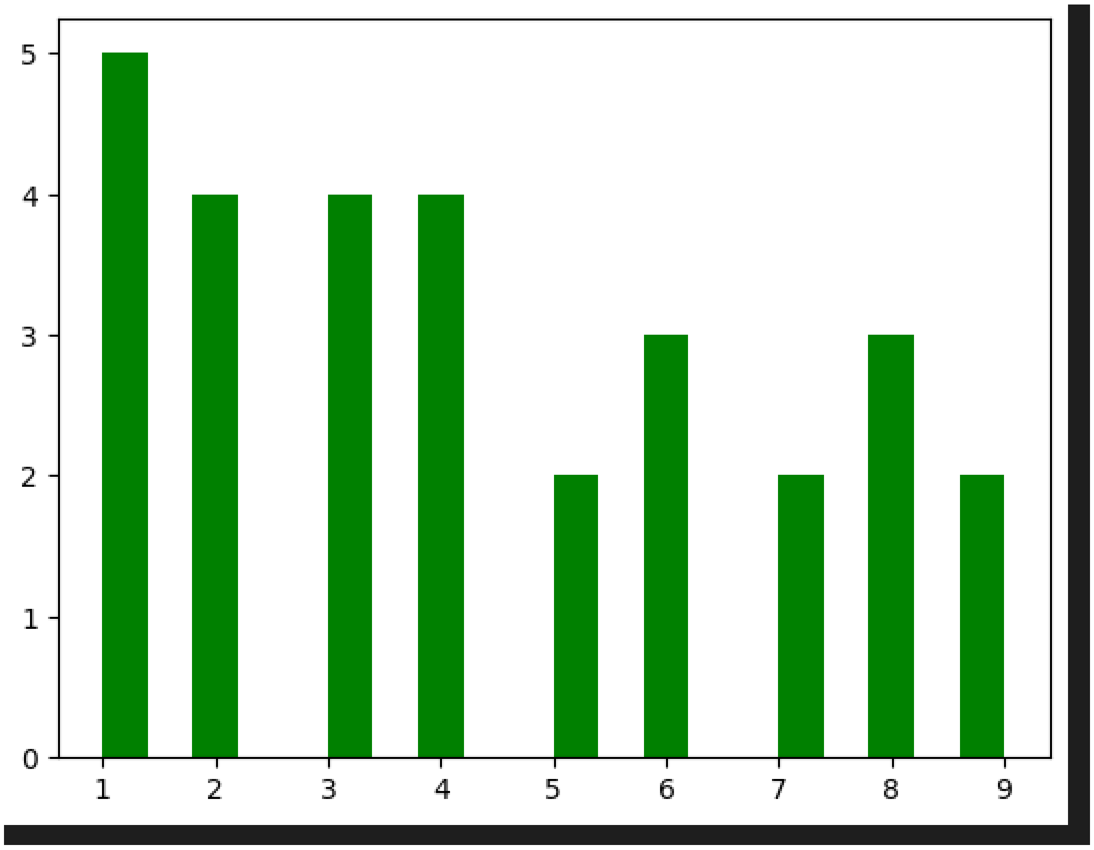
```
{'Ali': 25, 'Bilal': 20, 'Zahra': 19, 'Qayyum': 25, 'Alia': 15, 'Salman': 23}
```

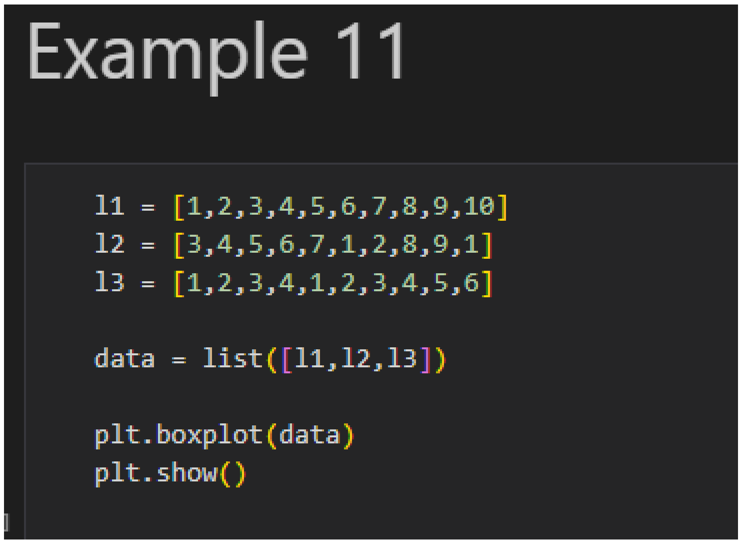




Example 10

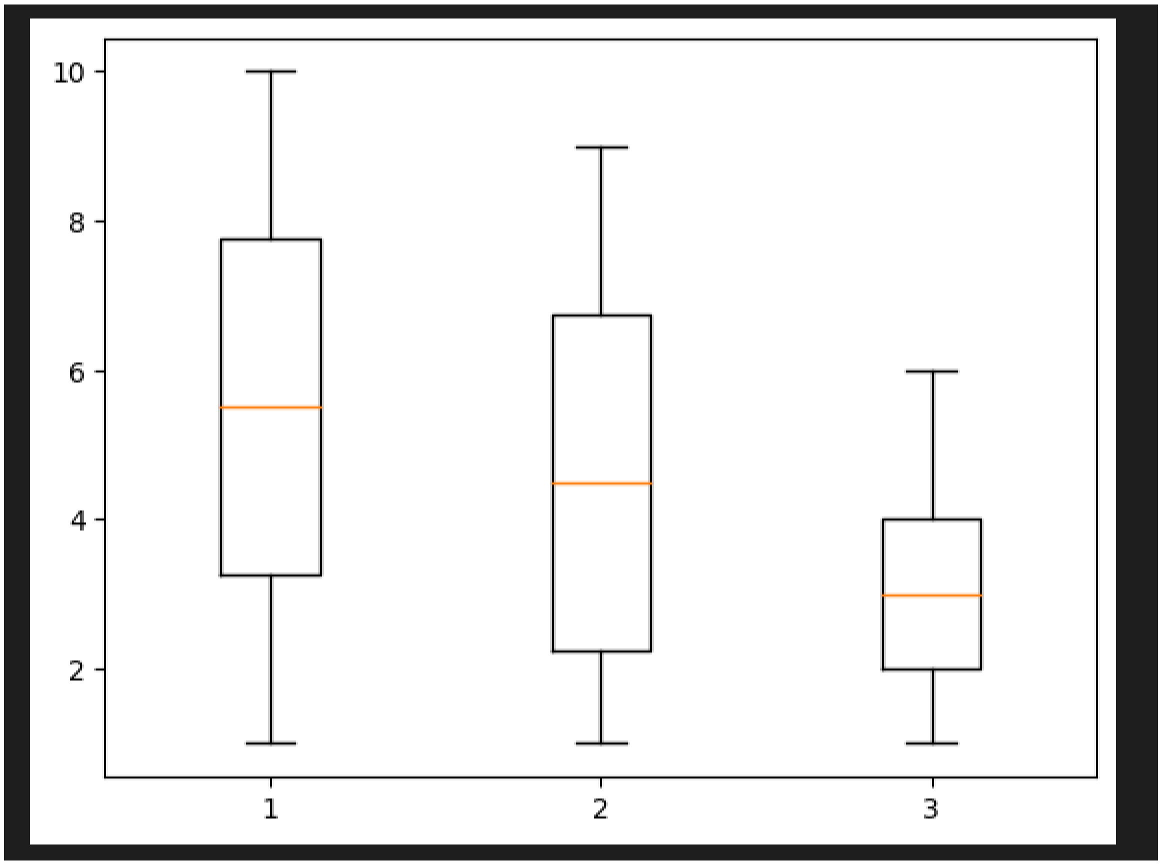
```
a = [1,2,3,4,5,6,7,8,9,4,6,8,2,3,1,1,6,8,9,3,4,2,1,1,2,3,4,5,7]

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

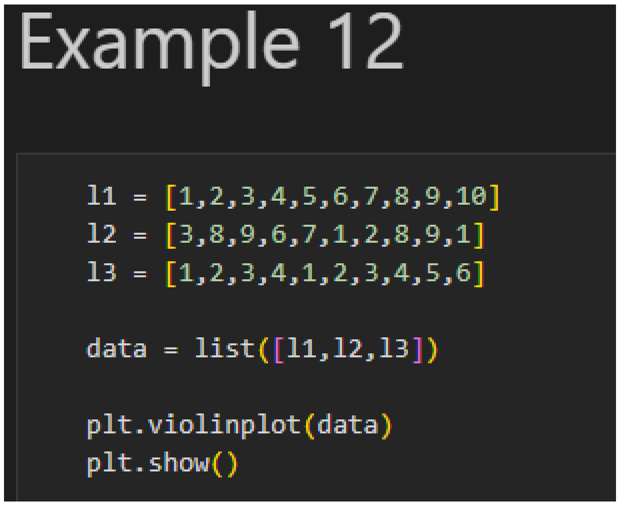




Example 11

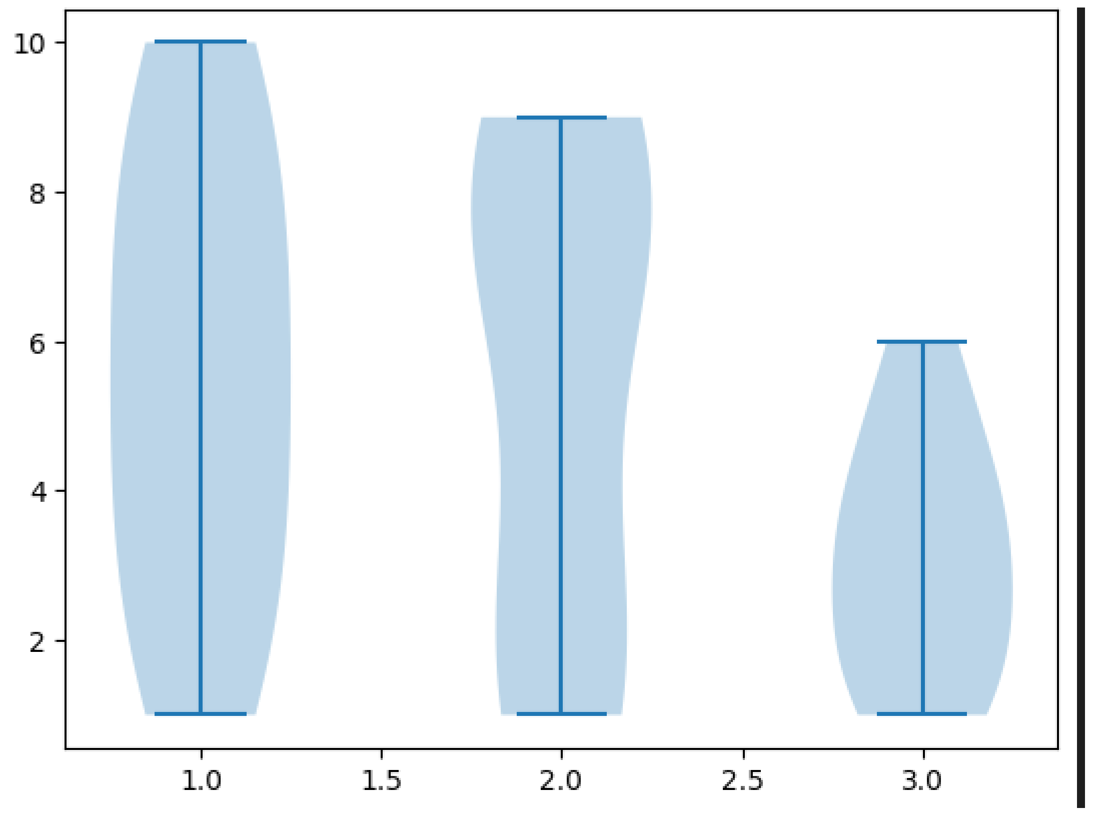
```
l1 = [1,2,3,4,5,6,7,8,9,10]
l2 = [3,4,5,6,7,1,2,8,9,1]
l3 = [1,2,3,4,1,2,3,4,5,6]

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

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

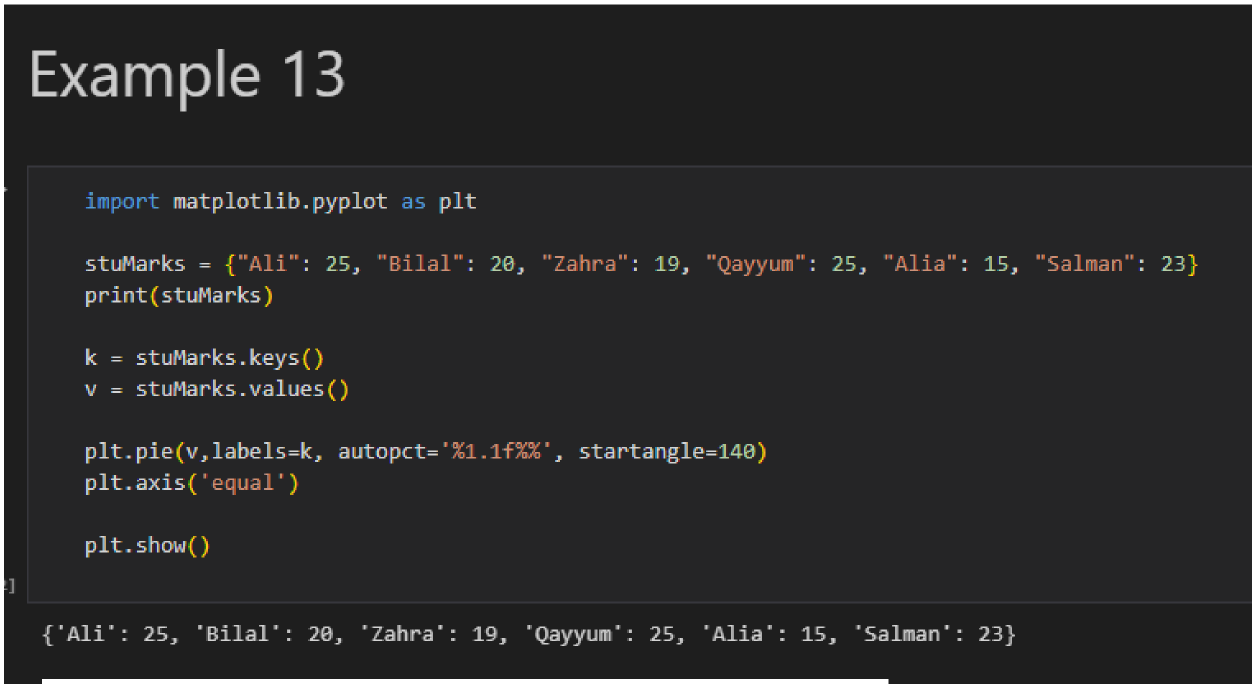




Example 12

```
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()
```

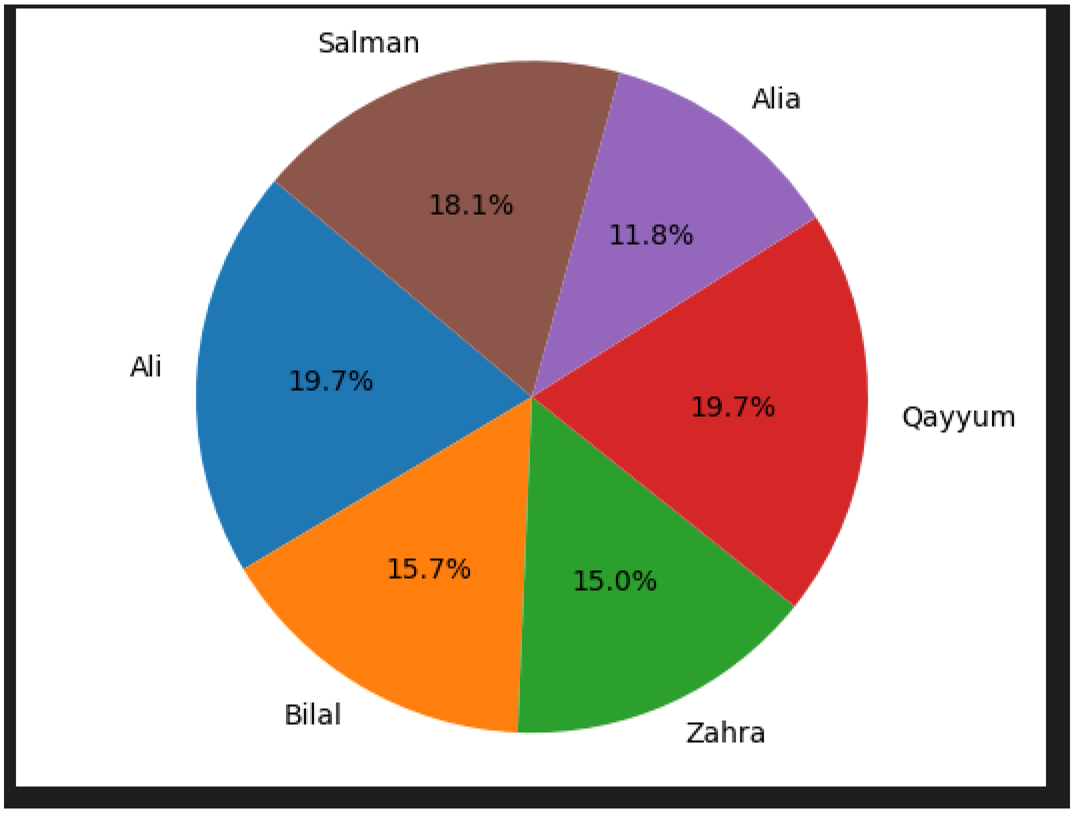




Example 13

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

Dr. A. Q. Khan Institute of Computer Sciences & Information Technology, (KICSIT)
Department of Computer Engineering

Artificial Intelligence Lab
All Rubrics of Microprocessor & Interfacing Lab

CLO 1

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 2

Marks	1	2	3	4
Model Implementation	The model is not implemented as per guidelines and requirements are not met	Some section of model is correctly implemented	Most section of model is correctly implemented and understands it well	The model is properly implemented, and have good understanding about it

CLO 3

Marks	1	2	3	4
Data Pre-processing	The data is not pre-processed as per guidelines and requirements are not met	Some section of data pre-processing is correct	Most section of data pre-processing is correct and understands it well	The data pre-processing is done properly, and have good understanding about it

CLO 4

Marks	1	2	3	4
Team Work	Rarely listens to, shares with, and supports the efforts of others. Often is not a good team member.	Often listen to, shares with and supports the efforts of others, but sometimes is not good team member.	Usually listen to, shares with, and supports the efforts of others. Usually, respectful and listening actively	Almost always listens to, shares with and supports the efforts of others. Tries to keep people working well together.

ab Report Rubric: *must be submitted in next lab***.**

Marks	1	2	3	4
Lab Report	The lab report does not follow the guidelines for formatting.	Presents some sections of the lab in the correct order. Three or more sections are not in the correct order; missing heading or title;	Presents most sections of the lab in the correct order, one or two sections may not be in the correct order; heading or title missing or not complete;	Presents all the sections of the lab in the correct order with correct formatting: includes correct heading, section headings and title of lab;