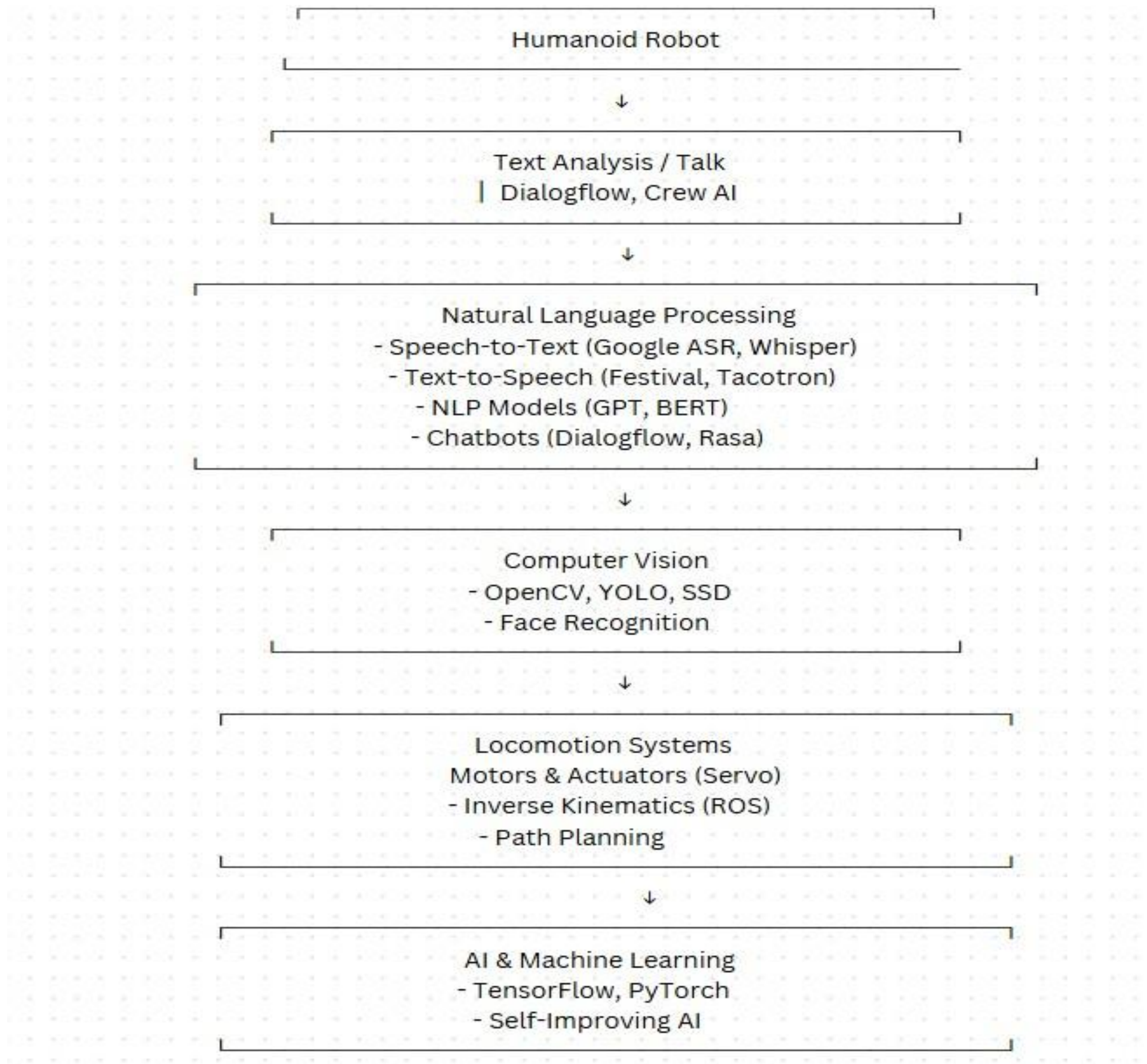
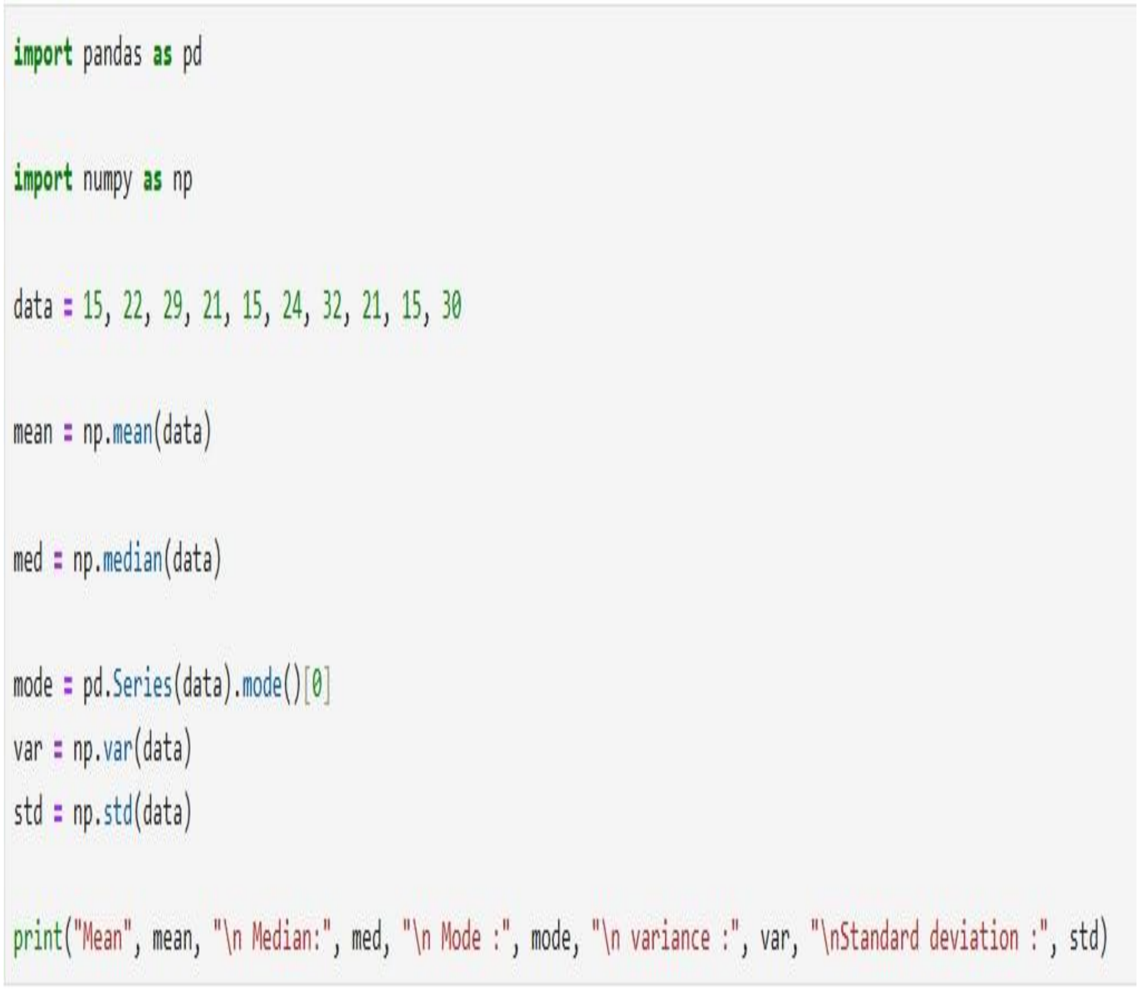
1.Imagine you are tasked with designing a humanoid robot to assist in a home or office environment. The robot must be capable of interacting with people by talking and listening, walking to different locations, seeing and recognizing objects, and learning from its surroundings to adapt its behaviour. What technologies, tools, and frameworks would you need to build such a robot? Give as flow chart.

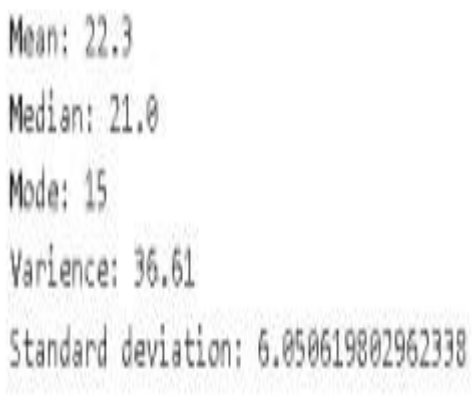


2. Calculate and interpret mean, median, mode, variance and standard deviation for a given dataset. Data = [15,21,29,21,15,24,32,21,15,30]
Source code:



```python
import pandas as pd


import numpy as np


data = 15, 22, 29, 21, 15, 24, 32, 21, 15, 30


mean = np.mean(data)


med = np.median(data)


mode = pd.Series(data).mode()[0]
var = np.var(data)
std = np.std(data)


print("Mean", mean, "\n Median:", med, "\n Mode :", mode, "\n variance :", var, "\nStandard deviation :", std)
```

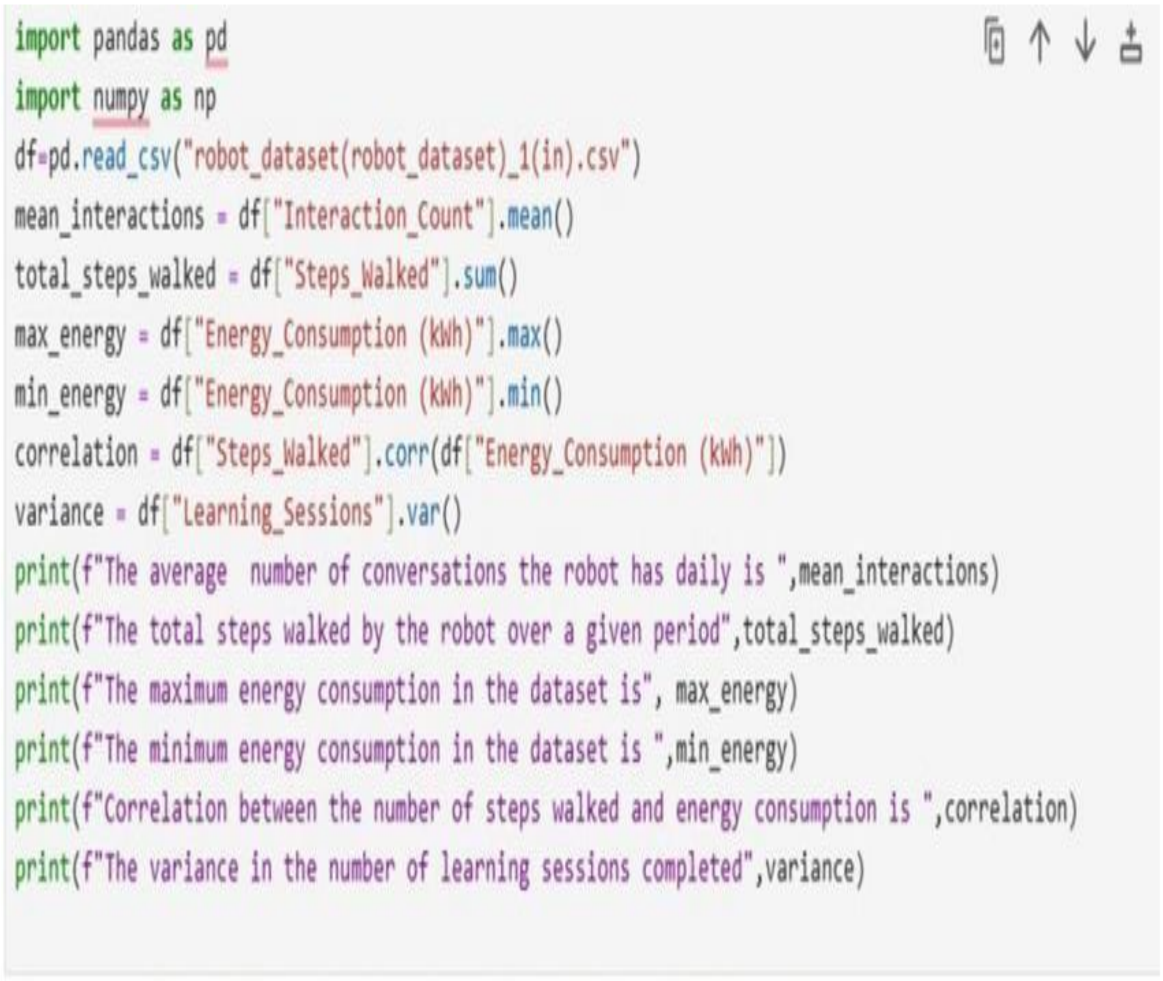
Output:



```
Mean: 22.3
Median: 21.0
Mode: 15
Varience: 36.61
Standard deviation: 6.050619802962338
```

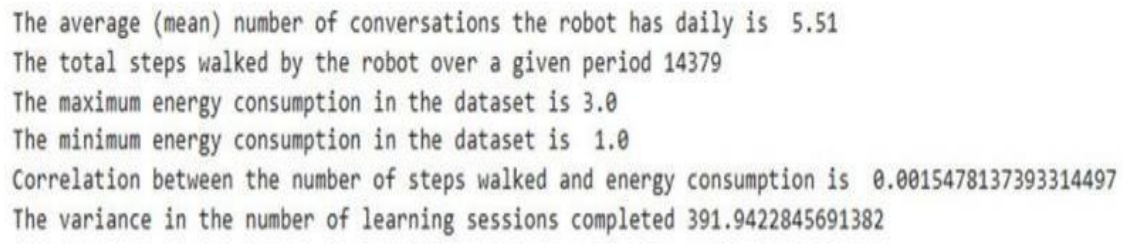
3. You are analyzing dataset that captures the adily performance and activity of a humanoid robot in a simulated environment.The database link robot _dataset(robot_dataset)_1.csv includes the following attributes.

Source Code:



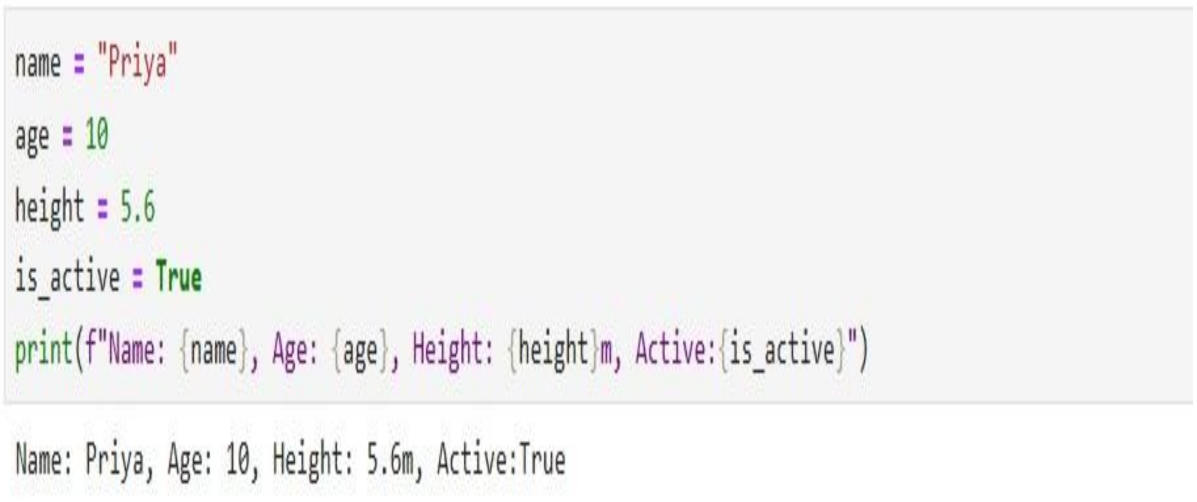
```python
import pandas as pd
import numpy as np
df=pd.read_csv("robot_dataset(robot_dataset)_1(in).csv")
mean_interactions = df["Interaction_Count"].mean()
total_steps_walked = df["Steps_Walked"].sum()
max_energy = df["Energy_Consumption (kWh)"].max()
min_energy = df["Energy_Consumption (kWh)"].min()
correlation = df["Steps_Walked"].corr(df["Energy_Consumption (kWh)"])
variance = df["Learning_Sessions"].var()
print(f"The average  number of conversations the robot has daily is ",mean_interactions)
print(f"The total steps walked by the robot over a given period",total_steps_walked)
print(f"The maximum energy consumption in the dataset is", max_energy)
print(f"The minimum energy consumption in the dataset is ",min_energy)
print(f"Correlation between the number of steps walked and energy consumption is ",correlation)
print(f"The variance in the number of learning sessions completed",variance)
```

Output:



```
The average (mean) number of conversations the robot has daily is  5.51
The total steps walked by the robot over a given period 14379
The maximum energy consumption in the dataset is 3.0
The minimum energy consumption in the dataset is  1.0
Correlation between the number of steps walked and energy consumption is  0.0015478137393314497
The variance in the number of learning sessions completed 391.9422845691382
```

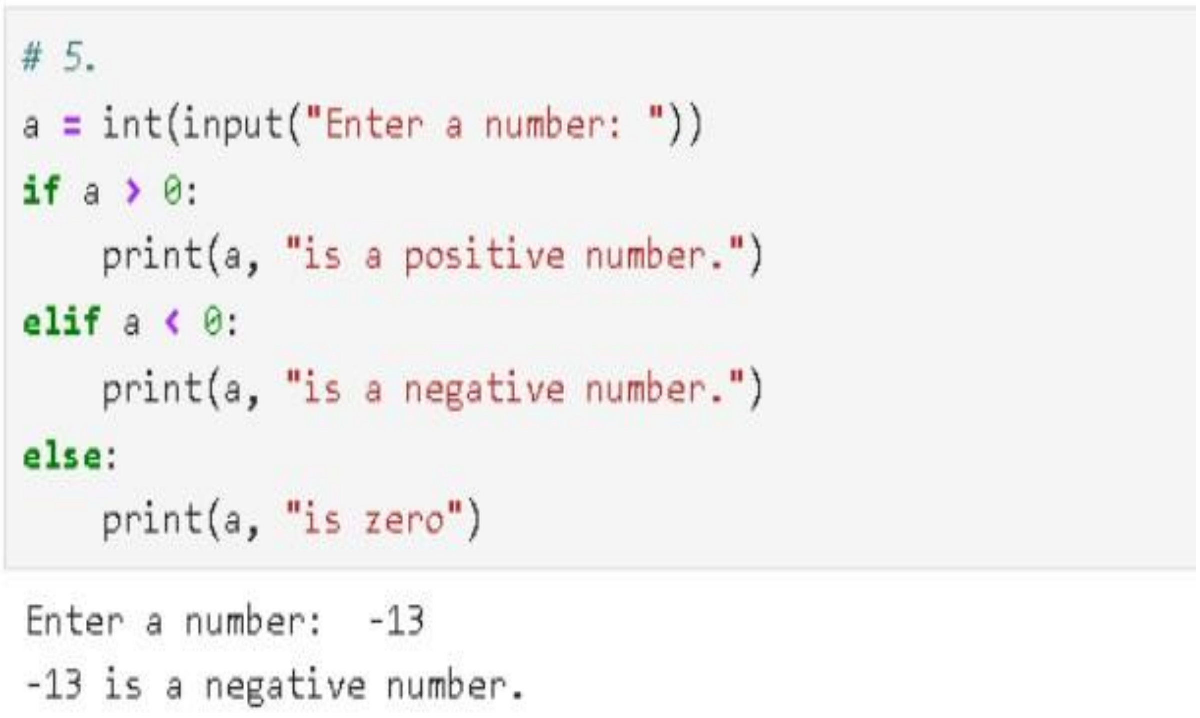
4.Write a Python program that declares variables of different data types (e.g., string, integer, float, and boolean). Output the variables in a sentence format using print() and f-strings.



```python
name = "Priya"
age = 10
height = 5.6
is_active = True
print(f"Name: {name}, Age: {age}, Height: {height}m, Active:{is_active}")
```

```
Name: Priya, Age: 10, Height: 5.6m, Active:True
```

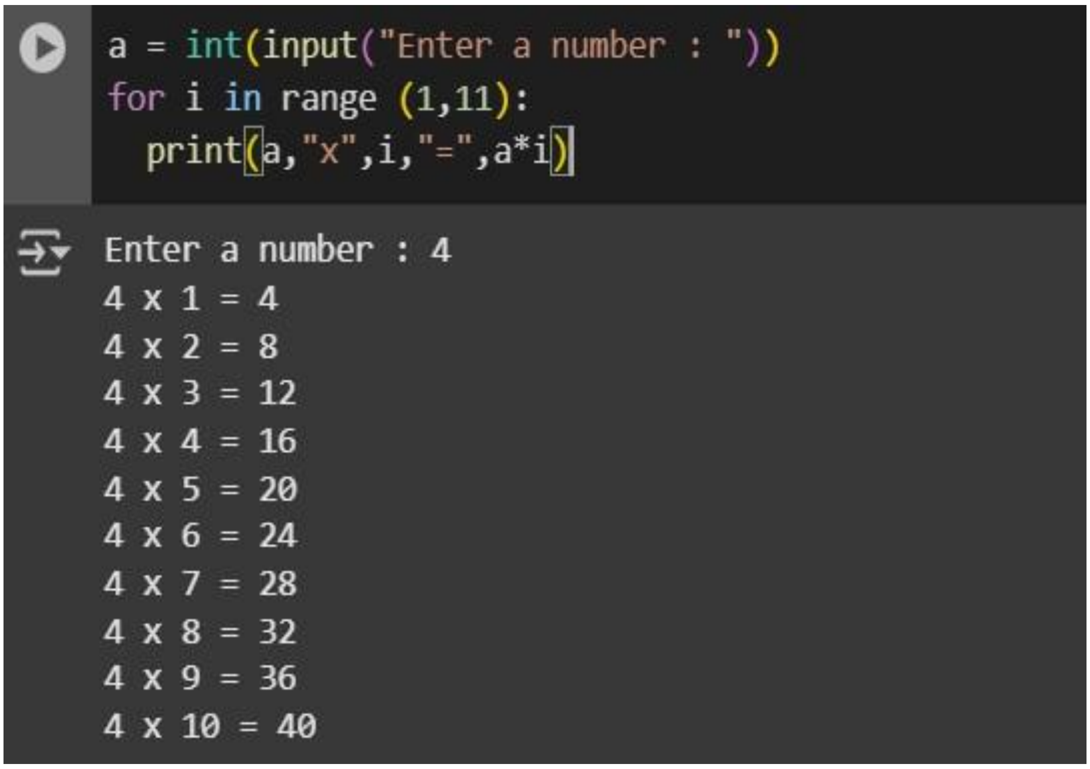
5. Write a Python program that takes an integer input and checks whether the number is positive, negative, or zero using conditional statements (if-else).



```python
# 5.
a = int(input("Enter a number: "))
if a > 0:
    print(a, "is a positive number.")
elif a < 0:
    print(a, "is a negative number.")
else:
    print(a, "is zero")
```

```
Enter a number:  -13
-13 is a negative number.
```

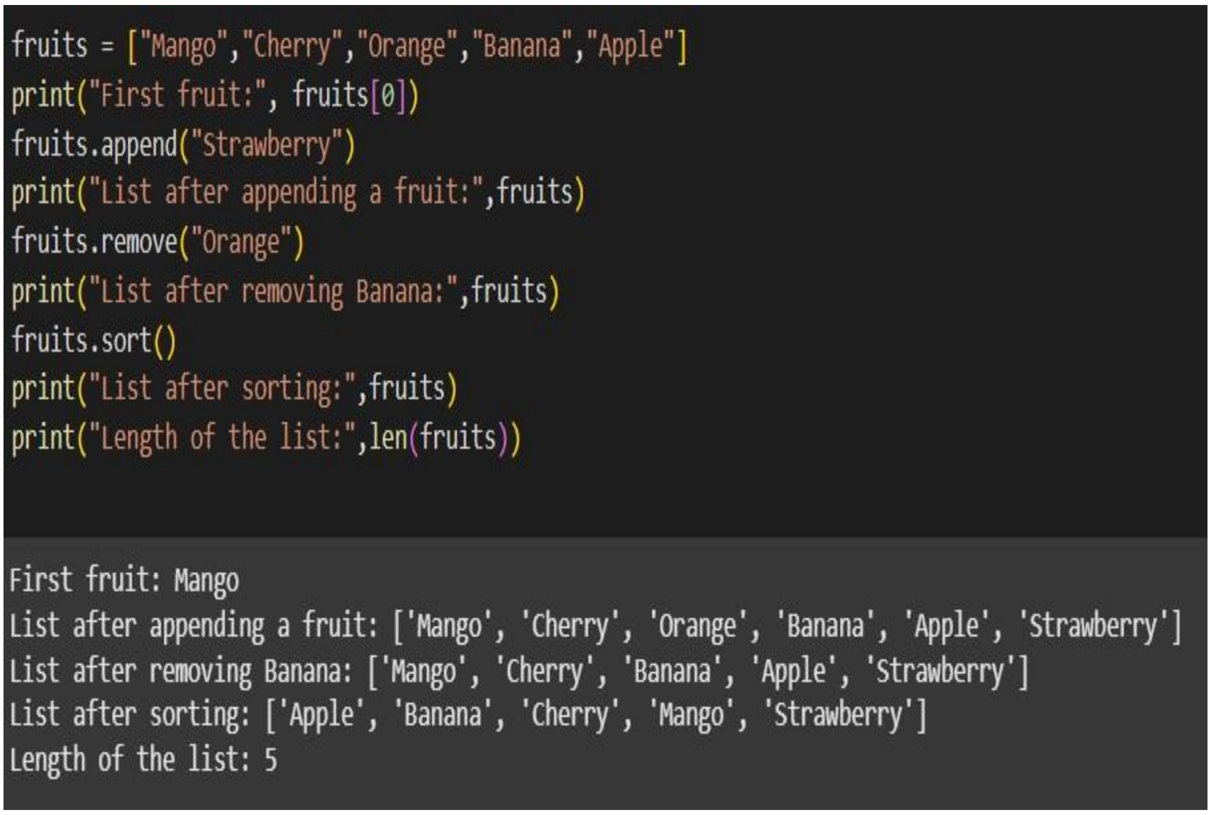
6.Write a Python program that takes a number as input and prints the multiplication table for that number (from 1 to 10).



```
a = int(input("Enter a number : "))
for i in range (1,11):
  print(a,"x",i,"=",a*i)
```

```
Enter a number : 4
4 x 1 = 4
4 x 2 = 8
4 x 3 = 12
4 x 4 = 16
4 x 5 = 20
4 x 6 = 24
4 x 7 = 28
4 x 8 = 32
4 x 9 = 36
4 x 10 = 40
```

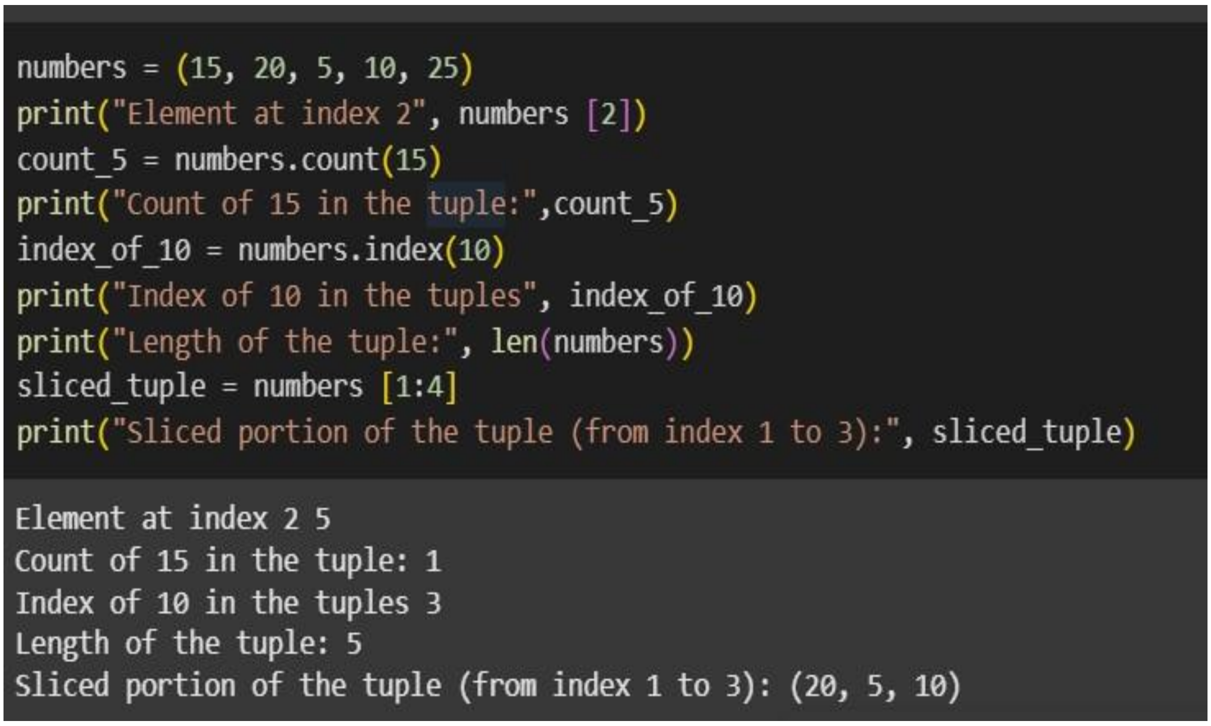
7.Create a Python list that contains the names of 5 different fruits. Perform the given operations on the list.



```
fruits = ["Mango","Cherry","Orange","Banana","Apple"]
print("First fruit:", fruits[0])
fruits.append("Strawberry")
print("List after appending a fruit:",fruits)
fruits.remove("Orange")
print("List after removing Banana:",fruits)
fruits.sort()
print("List after sorting:",fruits)
print("Length of the list:",len(fruits))
```

```
First fruit: Mango
List after appending a fruit: ['Mango', 'Cherry', 'Orange', 'Banana', 'Apple', 'Strawberry']
List after removing Banana: ['Mango', 'Cherry', 'Banana', 'Apple', 'Strawberry']
List after sorting: ['Apple', 'Banana', 'Cherry', 'Mango', 'Strawberry']
Length of the list: 5
```

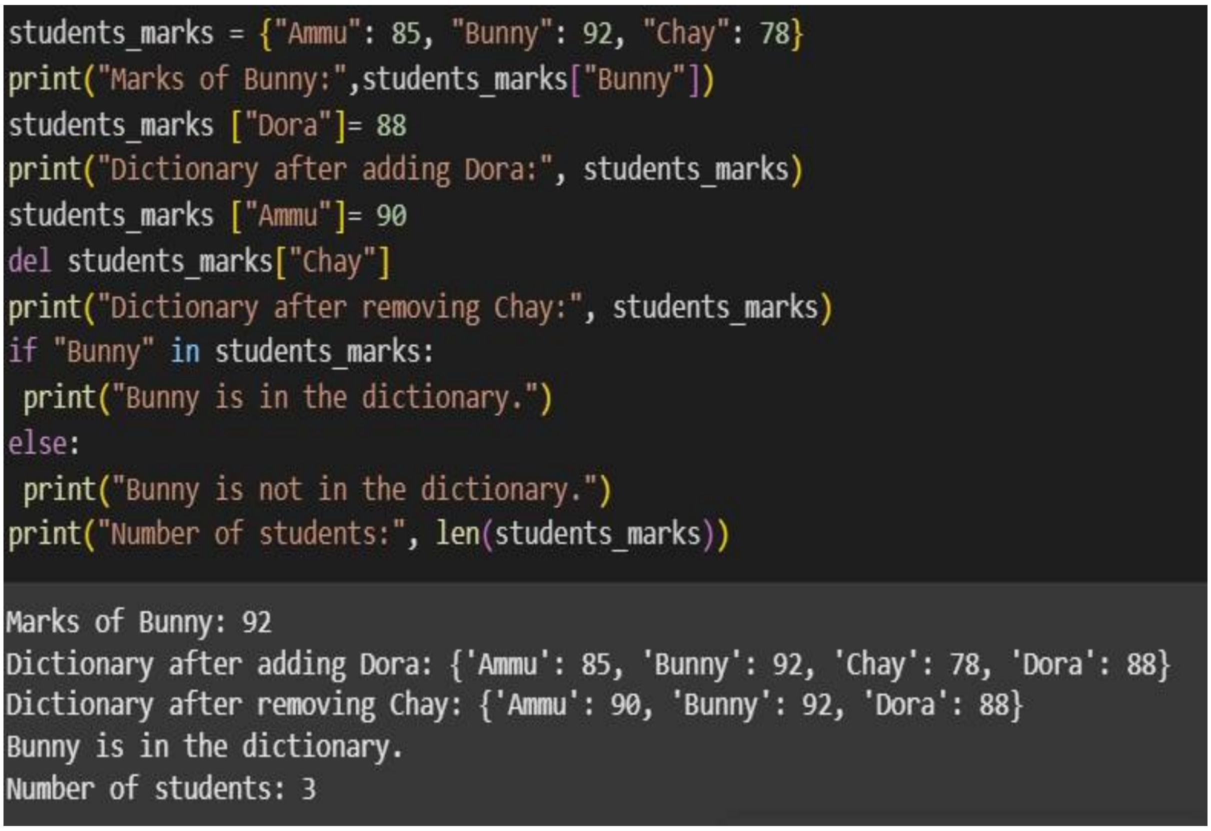
8.Write a Python program that creates a tuple containing 5 numbers. Perform the given operations on the tuple.



```python
numbers = (15, 20, 5, 10, 25)
print("Element at index 2", numbers [2])
count_5 = numbers.count(15)
print("Count of 15 in the tuple:",count_5)
index_of_10 = numbers.index(10)
print("Index of 10 in the tuples", index_of_10)
print("Length of the tuple:", len(numbers))
sliced_tuple = numbers [1:4]
print("Sliced portion of the tuple (from index 1 to 3):", sliced_tuple)
```

```
Element at index 2 5
Count of 15 in the tuple: 1
Index of 10 in the tuples 3
Length of the tuple: 5
Sliced portion of the tuple (from index 1 to 3): (20, 5, 10)
```

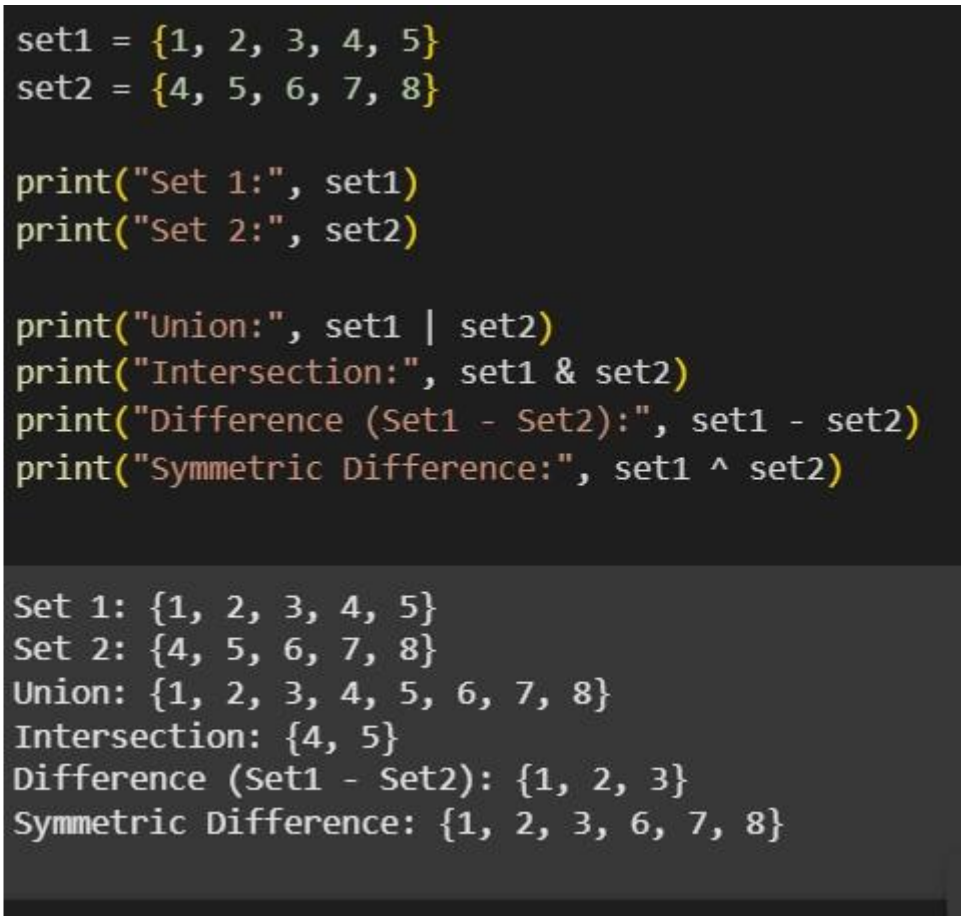
9.Create a dictionary that stores the names of 3 students as keys and their marks in mathematics as values. Perform the given operations.



```python
students_marks = {"Ammu": 85, "Bunny": 92, "Chay": 78}
print("Marks of Bunny:",students_marks["Bunny"])
students_marks ["Dora"]= 88
print("Dictionary after adding Dora:", students_marks)
students_marks ["Ammu"]= 90
del students_marks["Chay"]
print("Dictionary after removing Chay:", students_marks)
if "Bunny" in students_marks:
 print("Bunny is in the dictionary.")
else:
 print("Bunny is not in the dictionary.")
print("Number of students:", len(students_marks))
```

```
Marks of Bunny: 92
Dictionary after adding Dora: {'Ammu': 85, 'Bunny': 92, 'Chay': 78, 'Dora': 88}
Dictionary after removing Chay: {'Ammu': 90, 'Bunny': 92, 'Dora': 88}
Bunny is in the dictionary.
Number of students: 3
```

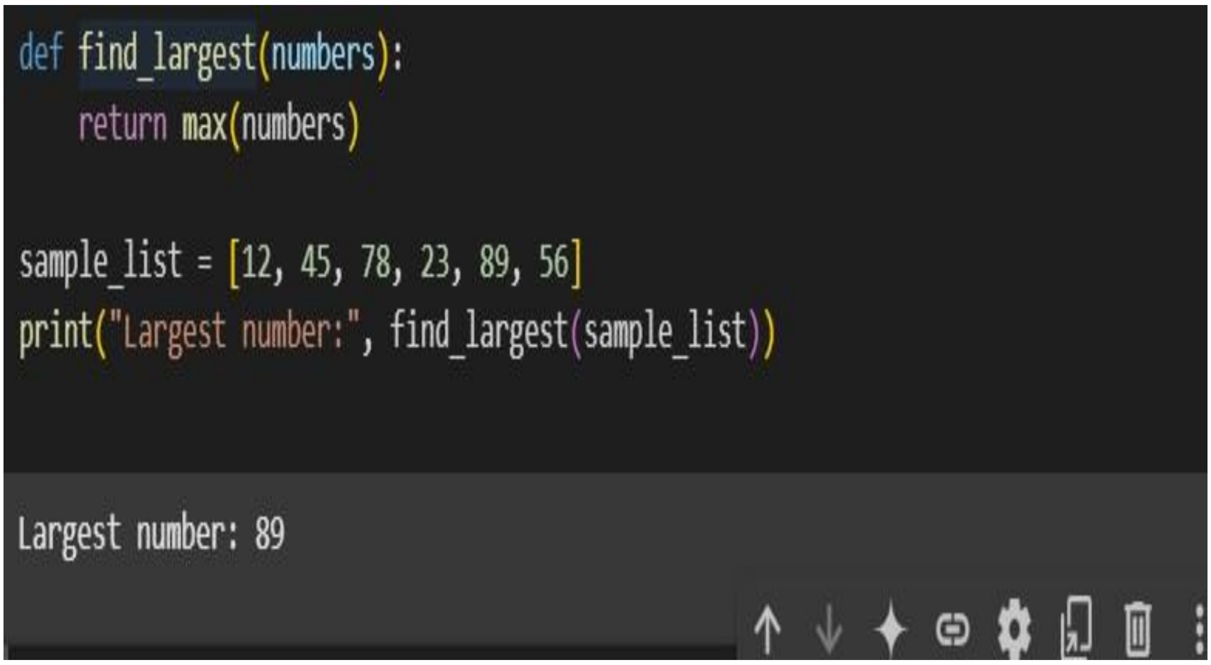
10. Create two sets of integers. Perform the given set operations.



```
set1 = {1, 2, 3, 4, 5}
set2 = {4, 5, 6, 7, 8}

print("Set 1:", set1)
print("Set 2:", set2)

print("Union:", set1 | set2)
print("Intersection:", set1 & set2)
print("Difference (Set1 - Set2):", set1 - set2)
print("Symmetric Difference:", set1 ^ set2)
```

```
Set 1: {1, 2, 3, 4, 5}
Set 2: {4, 5, 6, 7, 8}
Union: {1, 2, 3, 4, 5, 6, 7, 8}
Intersection: {4, 5}
Difference (Set1 - Set2): {1, 2, 3}
Symmetric Difference: {1, 2, 3, 6, 7, 8}
```

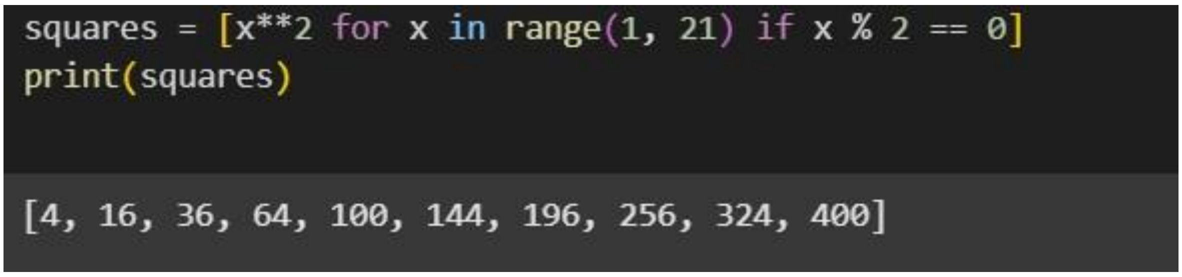
11.Write a Python function called find_largest() that takes a list of numbers as input and returns the largest number from the list. Test the function with a sample list.



```
def find_largest(numbers):
    return max(numbers)

sample_list = [12, 45, 78, 23, 89, 56]
print("Largest number:", find_largest(sample_list))
```

```
Largest number: 89
```

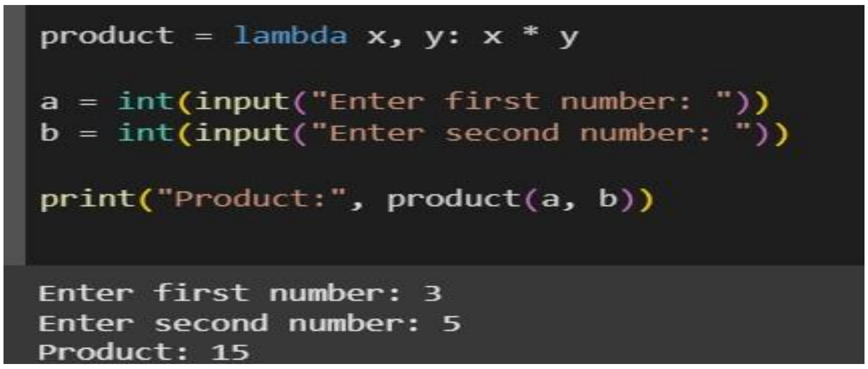
12.   12.Use list comprehension to create a list of squares of all even numbers between 1 and 20.



```
squares = [x**2 for x in range(1, 21) if x % 2 == 0]
print(squares)
```

```
[4, 16, 36, 64, 100, 144, 196, 256, 324, 400]
```

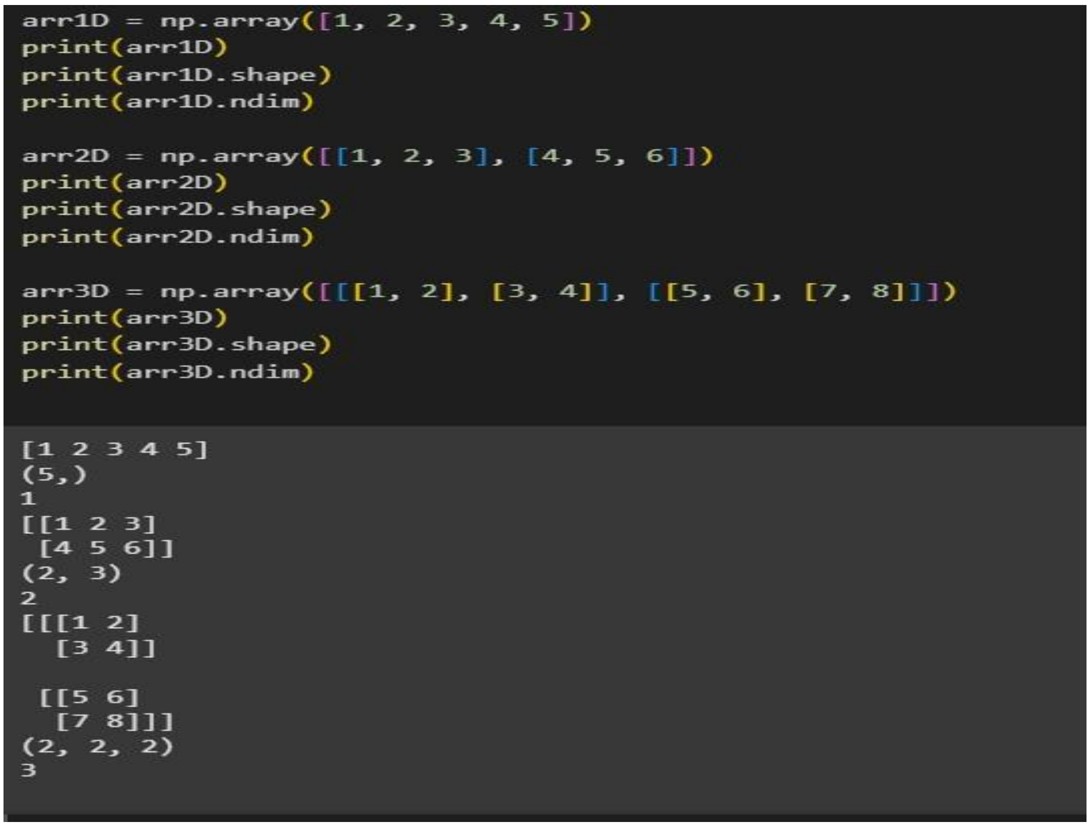
13.	13. Write a Python script that uses a lambda function to calculate the product of two numbers provided by the user.



```
product = lambda x, y: x * y

a = int(input("Enter first number: "))
b = int(input("Enter second number: "))

print("Product:", product(a, b))
```

```
Enter first number: 3
Enter second number: 5
Product: 15
```

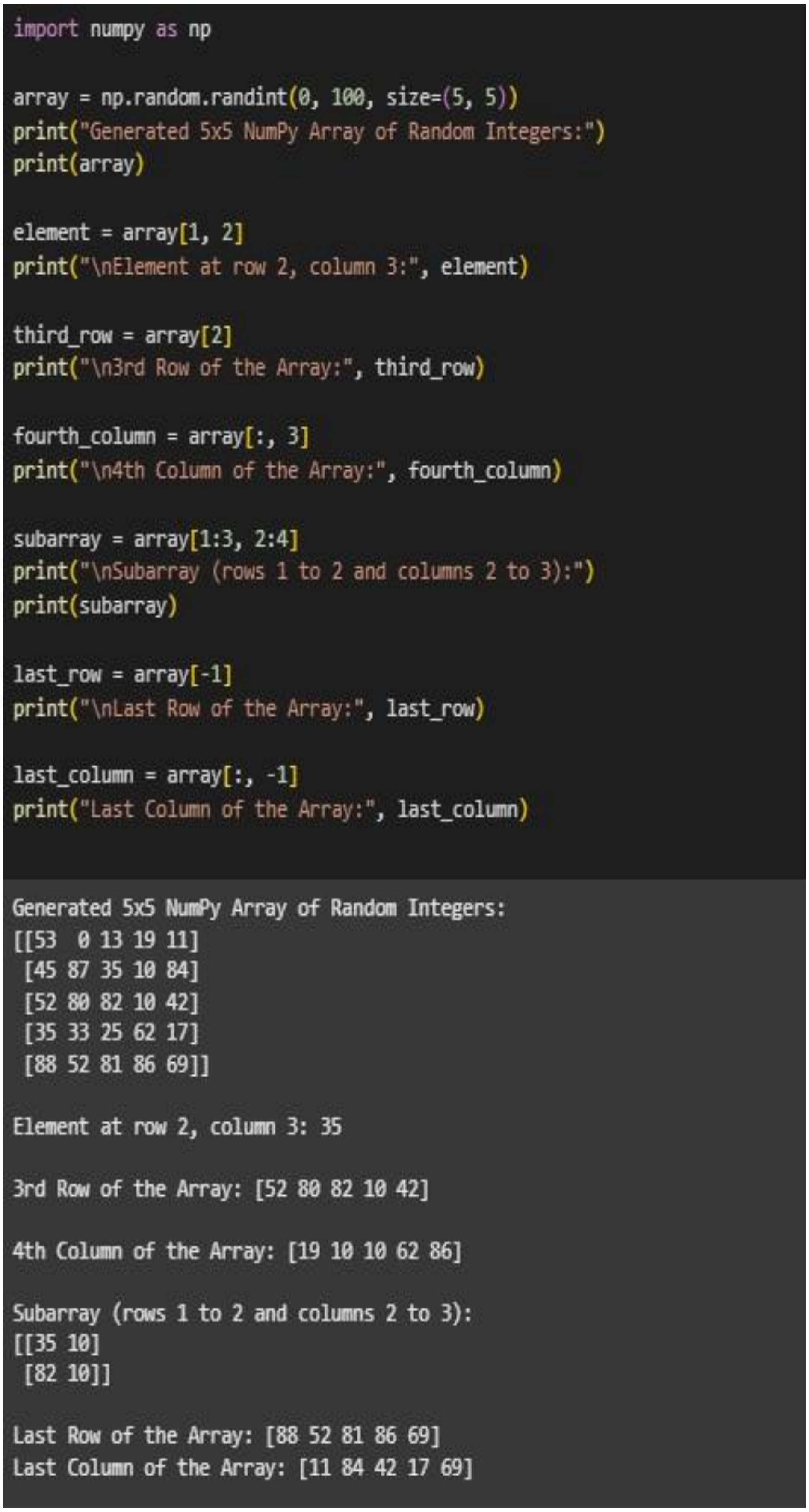
14.Write a Python program create a one-dimensional, two-dimensional, and three-dimensional NumPy array. Print the shape and dimensions of each array.



```
arr1D = np.array([1, 2, 3, 4, 5])
print(arr1D)
print(arr1D.shape)
print(arr1D.ndim)

arr2D = np.array([[1, 2, 3], [4, 5, 6]])
print(arr2D)
print(arr2D.shape)
print(arr2D.ndim)

arr3D = np.array([[[1, 2], [3, 4]], [[5, 6], [7, 8]]])
print(arr3D)
print(arr3D.shape)
print(arr3D.ndim)
```

```
[1 2 3 4 5]
(5,)
1
[[1 2 3]
 [4 5 6]]
(2, 3)
2
[[[1 2]
  [3 4]]

 [[5 6]
  [7 8]]]
(2, 2, 2)
3
```

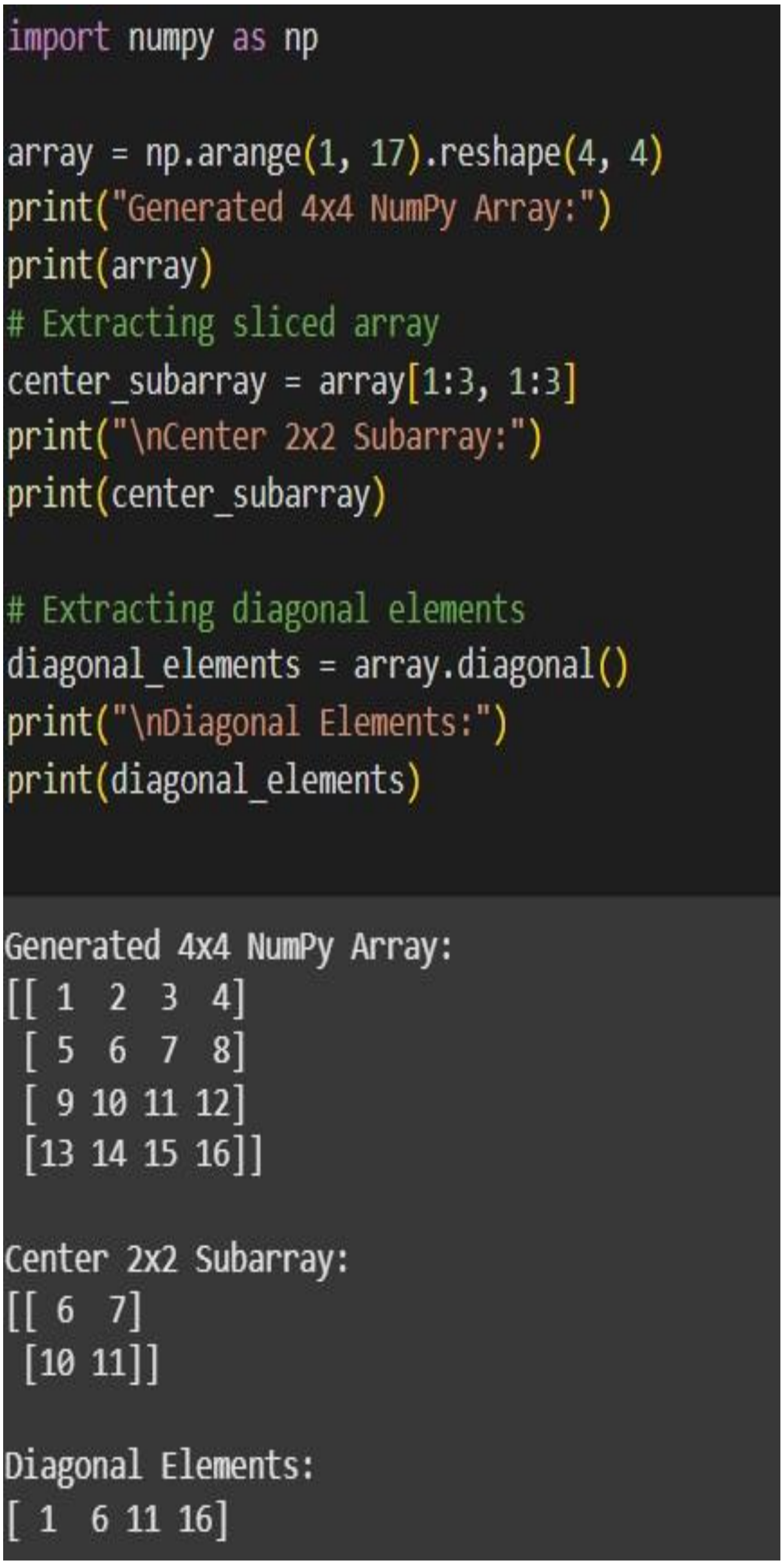
15.Write a Python program to create a 5x5 NumPy array of random integers and Perform array indexing as given.



```
import numpy as np

array = np.random.randint(0, 100, size=(5, 5))
print("Generated 5x5 NumPy Array of Random Integers:")
print(array)

element = array[1, 2]
print("\nElement at row 2, column 3:", element)

third_row = array[2]
print("\n3rd Row of the Array:", third_row)

fourth_column = array[:, 3]
print("\n4th Column of the Array:", fourth_column)

subarray = array[1:3, 2:4]
print("\nSubarray (rows 1 to 2 and columns 2 to 3):")
print(subarray)

last_row = array[-1]
print("\nLast Row of the Array:", last_row)

last_column = array[:, -1]
print("Last Column of the Array:", last_column)
```

```
Generated 5x5 NumPy Array of Random Integers:
[[53  0 13 19 11]
 [45 87 35 10 84]
 [52 80 82 10 42]
 [35 33 25 62 17]
 [88 52 81 86 69]]

Element at row 2, column 3: 35

3rd Row of the Array: [52 80 82 10 42]

4th Column of the Array: [19 10 10 62 86]

Subarray (rows 1 to 2 and columns 2 to 3):
[[35 10]
 [82 10]]

Last Row of the Array: [88 52 81 86 69]
Last Column of the Array: [11 84 42 17 69]
```

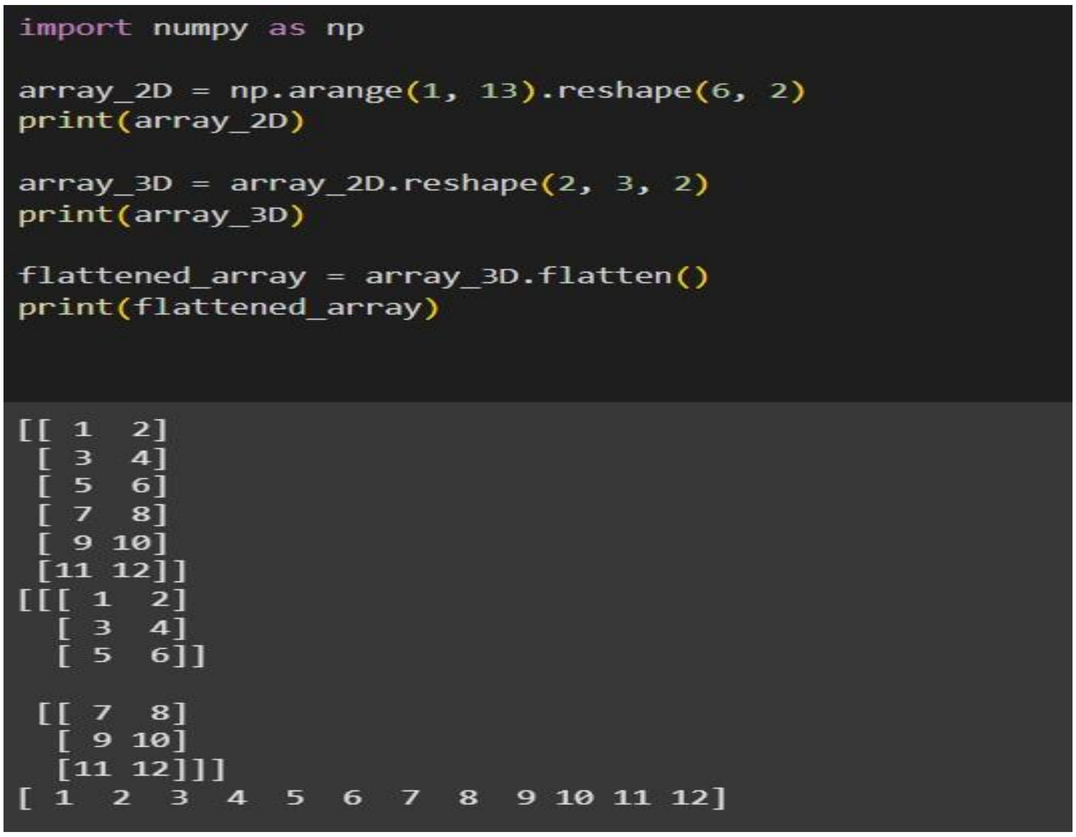
16.Create a NumPy array of shape (4, 4) containing numbers from 1 to 16. Use slicing to extract for the given conditions



```python
import numpy as np

array = np.arange(1, 17).reshape(4, 4)
print("Generated 4x4 NumPy Array:")
print(array)
# Extracting sliced array
center_subarray = array[1:3, 1:3]
print("\nCenter 2x2 Subarray:")
print(center_subarray)

# Extracting diagonal elements
diagonal_elements = array.diagonal()
print("\nDiagonal Elements:")
print(diagonal_elements)
```

```
Generated 4x4 NumPy Array:
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15 16]]

Center 2x2 Subarray:
[[ 6  7]
 [10 11]]

Diagonal Elements:
[ 1  6 11 16]
```

17. 17.Write a Python program that creates a 2D array of shape (6, 2) using np.arange() and then reshapes it into a 3D array of shape (2, 3, 2). Flatten the reshaped array and print the result.

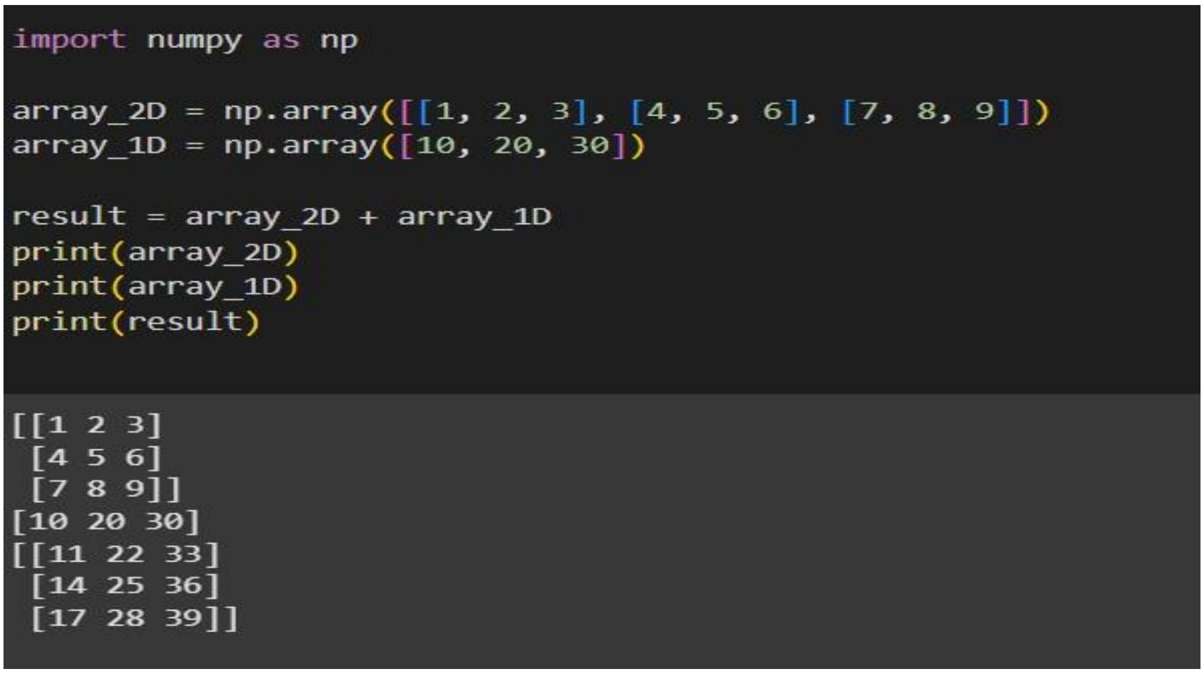


```
import numpy as np

array_2D = np.arange(1, 13).reshape(6, 2)
print(array_2D)

array_3D = array_2D.reshape(2, 3, 2)
print(array_3D)

flattened_array = array_3D.flatten()
print(flattened_array)
```

```
[[ 1  2]
 [ 3  4]
 [ 5  6]
 [ 7  8]
 [ 9 10]
 [11 12]]
[[[ 1  2]
  [ 3  4]
  [ 5  6]]

 [[ 7  8]
  [ 9 10]
  [11 12]]]
[ 1  2  3  4  5  6  7  8  9 10 11 12]
```

.

18. Write a Python program to demonstrate broadcasting. Create an array of shape (3, 3) and add a one-dimensional array of shape (1, 3) to it using broadcasting.



```
import numpy as np

array_2D = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
array_1D = np.array([10, 20, 30])

result = array_2D + array_1D
print(array_2D)
print(array_1D)
print(result)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[10 20 30]
[[11 22 33]
 [14 25 36]
 [17 28 39]]
```

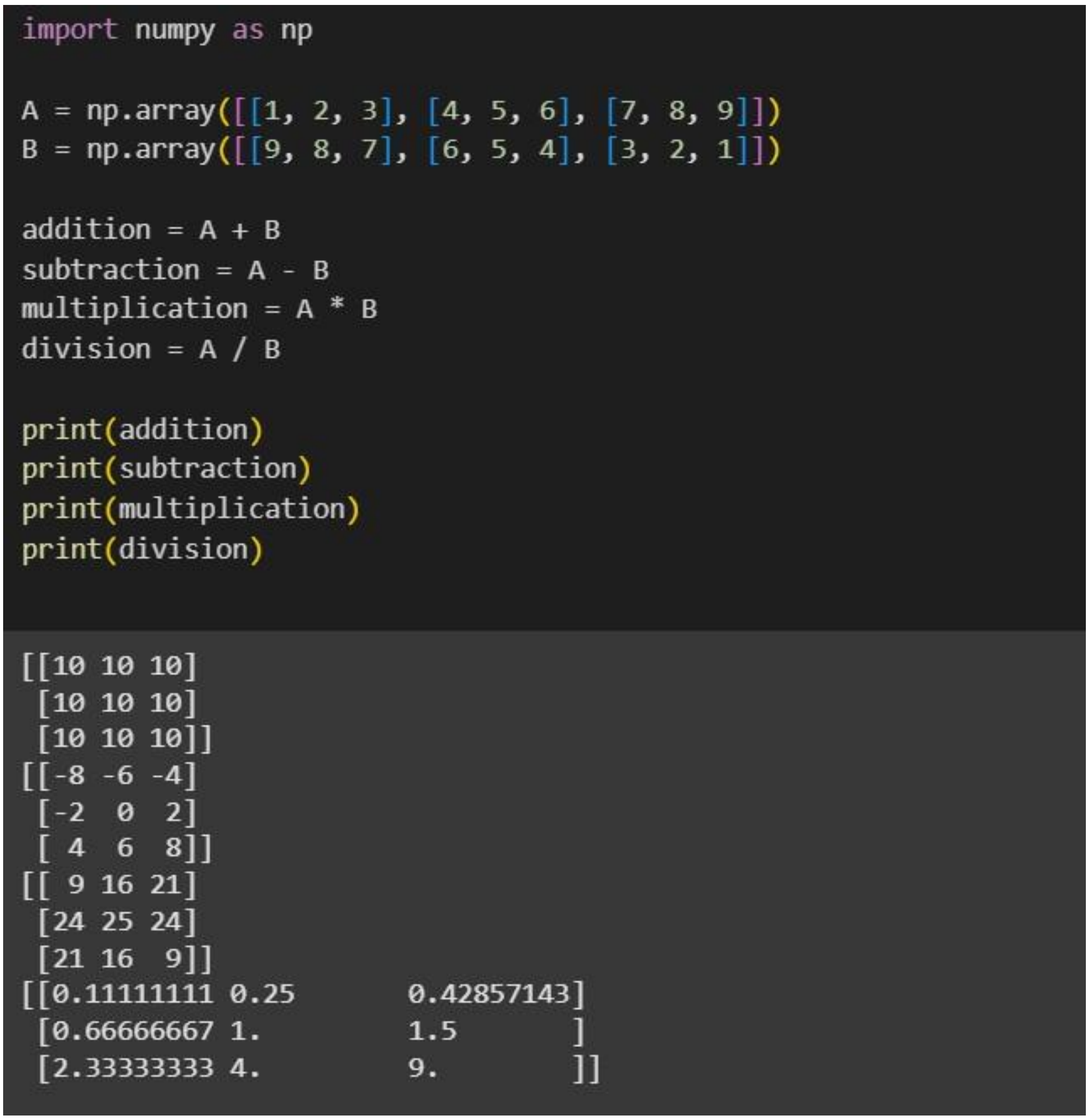
19.Create two NumPy arrays of the same shape, A and B. Perform the following arithmetic operations:

Element-wise addition.

Element-wise subtraction.

Element-wise multiplication.

Element-wise division

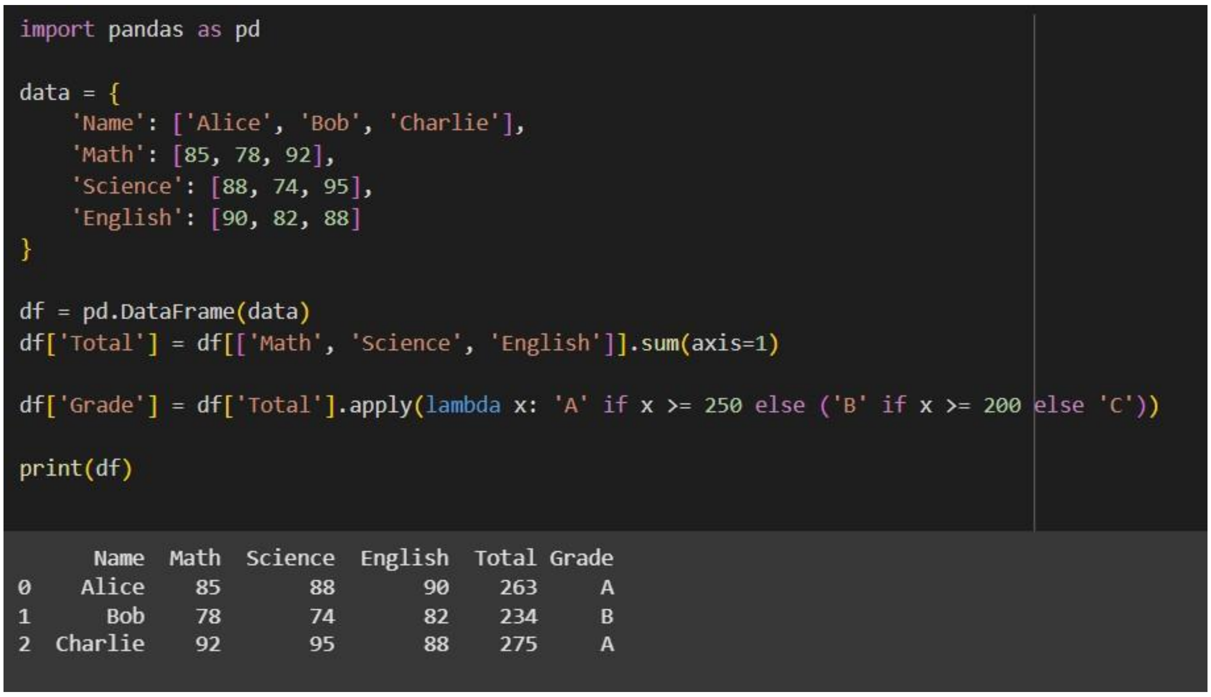


```python
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
B = np.array([[9, 8, 7], [6, 5, 4], [3, 2, 1]])

addition = A + B
subtraction = A - B
multiplication = A * B
division = A / B

print(addition)
print(subtraction)
print(multiplication)
print(division)
```

```
[[10 10 10]
 [10 10 10]
 [10 10 10]]
[[-8 -6 -4]
 [-2  0  2]
 [ 4  6  8]]
[[ 9 16 21]
 [24 25 24]
 [21 16  9]]
[[0.11111111 0.25       0.42857143]
 [0.66666667 1.         1.5       ]
 [2.33333333 4.         9.        ]]
```

20. Create a Pandas DataFrame with the given Name and marks of 3 courses:

Add a new column named 'Total' that represents the sum of all the courses. Add 'Grade' based on the values of the 'Total'. Print the updated DataFrame with the new 'Total' and 'Grade' column.



```python
import pandas as pd

data = {
    'Name': ['Alice', 'Bob', 'Charlie'],
    'Math': [85, 78, 92],
    'Science': [88, 74, 95],
    'English': [90, 82, 88]
}

df = pd.DataFrame(data)
df['Total'] = df[['Math', 'Science', 'English']].sum(axis=1)

df['Grade'] = df['Total'].apply(lambda x: 'A' if x >= 250 else ('B' if x >= 200 else 'C'))

print(df)
```

```
      Name  Math  Science  English  Total Grade
0    Alice    85       88       90    263     A
1      Bob    78       74       82    234     B
2  Charlie    92       95       88    275     A
```