NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:14

SOURCE CODE:

```
multiply = lambda x, y: x * y
def printTable(number):
    print(f"Table of ",number)
    for i in range(1, 11):
        result = multiply(number, i)
        print(number,"*",i,":",result)


number = int(input("Enter a number: "))
printTable(number)
```

OUTPUT:

Enter a number: 4

Table of 4

4 * 1 : 4

4 * 2 : 8

4 * 3 : 12

4 * 4 : 16

4 * 5 : 20

4 * 6 : 24

4 * 7 : 28

4 * 8 : 32

4 * 9 : 36

4 * 10 : 40

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:15

SOURCE CODE:

```
numbers = [1, 2, 3, 4, 5]

square = lambda x: x ** 2
cube = lambda x: x ** 3

squared_numbers = list(map(square, numbers))
 cubed_numbers = list(map(cube, numbers))

print("Original Numbers:", numbers)
print("Squared Numbers:", squared_numbers)
print("Cubed Numbers:", cubed_numbers)
```

OUTPUT:

Original Numbers: [1, 2, 3, 4, 5]

Squared Numbers: [1, 4, 9, 16, 25]

Cubed Numbers: [1, 8, 27, 64, 125]

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:16

SOURCE CODE:

```
class rec:

    def _init_(self):
        self.length = eval(input("Enter length of a rectangle: "))
        self.breadth = eval(input("Enter breadth of a rectangle: "))

    def sides(self):
        if self.length == self.breadth :
            print("It is a square.")
        else:
            print("It is not a square.")

obj = rec()
obj._init_()
obj.sides()
```

OUTPUT:

Enter length of a rectangle: 4

Enter breadth of a rectangle: 5

It is not a square.

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:17

SOURCE CODE:

```
class equi:
    def values(self,side1,side2,side3):
        if (side1==side2) and (side2==side3):
            print("The triangle is an Equilateral triangle.")
        else:
            print("The triangle is not an equilateral triangle.")

side1 = eval(input("Enter first side of a triangle: "))
side2 = eval(input("Enter second side of a triangle: "))
side3 = eval(input("Enter third side of a triangle: "))

obj = equi()
obj.values(side1,side2,side3)
```

OUTPUT:

Enter first side of a triangle: 3

Enter second side of a triangle: 3

Enter third side of a triangle: 3

The triangle is an Equilateral triangle.

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:18

SOURCE CODE:

```
class addition:
    def add(self,n1,n2):
        add=n1+n2
        print(n1,"+",n2,":",add)
class subtraction:
    def sub(self,n1,n2):
        sub=n1-n2
        print(n1,"-",n2,":",sub)
class multiplication:
    def multiply(self,n1,n2):
        multiply=n1*n2
        print(n1,"*",n2,":",multiply)
class division:
    def divide(self,n1,n2):
        divide=n1/n2
        print(n1,"/",n2,":",divide)
n1 = eval(input("Enter first number: "))
n2 = eval(input("Enter second number: "))

add_obj = addition()
add_obj.add(n1,n2)

sub_obj = subtraction()
sub_obj.sub(n1,n2)

multiply_obj=multiplication()
multiply_obj.multiply(n1,n2)

divide_obj = division()
divide_obj.divide(n1,n2)
```

OUTPUT:

Enter first number: 40

Enter second number: 50

40 + 50 : 90

40 - 50 : -10

40 * 50 : 2000

40 / 50 : 0.8

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:19

SOURCE CODE:

```
class printNum:
    def printEven(self):
        print("Even numbers : ")
        for i in range(1,101):
            if i%2 == 0:
                print(i,end = ' ')


    def printOdd(self):
        print("\nOdd numbers : ")
        for i in range(1,101):
            if i%2 != 0:
                print(i,end = ' ')

obj = printNum()
obj.printEven()
obj.printOdd()
```

OUTPUT:

Even numbers :

2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100

Odd numbers :

1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:20

SOURCE CODE:

```
class printTable:
    def table(self,number):
        for i in range(1,11):
            print(number,'*',i,"=",number*i)

num = int(input("Enter number to print table : "))
obj = printTable()
obj.table(num)
```

OUTPUT:

Enter number to print table : 6
6 * 1 = 6
6 * 2 = 12
6 * 3 = 18
6 * 4 = 24
6 * 5 = 30
6 * 6 = 36
6 * 7 = 42
6 * 8 = 48
6 * 9 = 54
6 * 10 = 60

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:21

SOURCE CODE:

```
import numpy as np
arr = np.array([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 9]])

sumOfElements = np.sum(arr)

print("Sum of all elements:", sumOfElements)
```

OUTPUT:

Sum of all elements: 45

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:22

SOURCE CODE:

```
import numpy as np

x = np.array([1, 2, 3, 4, 5])
y = np.array([5, 2, 3, 8, 5])

same_positions = np.where(x == y)[0]

print("Positions where elements of x and y are the same:", same_positions)
```

OUTPUT:

Positions where elements of x and y are the same: [1 2 4]

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:23

SOURCE CODE:

```
import numpy as np

array = np.arange(13, 40).reshape(9, 3)
sub_arrays = np.split(array, 3)
print("Original 9x3 array:")
print(array)

print("\nThree equal-sized sub-arrays:")
for i, sub_array in enumerate(sub_arrays):
    print(f"Sub-array {i+1}:")
    print(sub_array)
```

OUTPUT:

Original 9x3 array:

```
[[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]]
```

Three equal-sized sub-arrays:

```
Sub-array 1:
[[13 14 15]
 [16 17 18]
 [19 20 21]]
Sub-array 2:
[[22 23 24]
 [25 26 27]
 [28 29 30]]
Sub-array 3:
[[31 32 33]
 [34 35 36]
 [37 38 39]]
```

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:24

SOURCE CODE:

```
import pandas as pd
import numpy as np

data = {
   'A': [1, 2, 3],
   'B': [4, 5, 6],
   'C': [7, 8, 9]
}
df = pd.DataFrame(data)

numpy_array = df.to_numpy()

print("\nOriginal DataFrame:")
print(df)
print("\nConverted NumPy array:")
print(numpy_array)
```

OUTPUT:

Original DataFrame:

A B C

0 1 4 7

1 2 5 8

2 3 6 9

Converted NumPy array:

[[1 4 7]

[2 5 8]

[3 6 9]]

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

Problem statement:25

SOURCE CODE:

```
import pandas as pd

data = {
        ' Name': ['Alice', 'Bob', 'Charlie'],
        ' Age': [25, 30, 35],
        ' City': ['New York', 'Los Angeles', 'Chicago']
    }

    df = pd.DataFrame(data)
    print("\n",df)

    csv_file_path = 'output.csv'
    df.to_csv(csv_file_path)

    print(f"\nDataFrame has been written to {csv_file_path}")
```

OUTPUT:

Name Age City

0 Alice 25 New York

1 Bob 30 Los Angeles

2 Charlie 35 Chicago

OUTPUT.csv

,Name,Age,City

0,Alice,25,New York

1,Bob,30,Los Angeles

2,Charlie,35,Chicago

NAME: Ashish Kothari
SECTION: D1
ROLL NO: 17

Sub Code: PBC 401
Course: BCA
University roll.no:2221283

PROBLEM STATEMENT:26

SOURCE CODE:

```
import matplotlib.pyplot as plt

# Step 1: Create sample data
x = [1, 2, 3, 4, 5]
y = [2, 3, 5, 7, 11]

# Step 2: Plot the data using Matplotlib
plt.plot(x, y, marker='o', linestyle='-', color='b', label='Sample Data')

# Step 3: Customize the plot (optional)
plt.title('Sample Line Chart')
plt.xlabel('X-axis Label')
plt.ylabel('Y-axis Label')
plt.legend()
plt.grid(True)

# Step 4: Display the plot
plt.show()
```

OUTPUT: