What will be the output for the following code?

```
import numpy as np
a=np.arange(6).reshape(3,-1)
print(a.shape)
```

What will be output for the following code?

```
class A:
    def hello(self):
        print("1",end=",")
    def hello(self):
        print("2",end=",")
class B(A):
    def __init__(self):
        self.hello()
    def hello(self):
         print("3",end=",")
         super().hello()
b=B()
b.hello()
```

What will be output for the following code?

```
class A:
    money=5000
    def __init__(self):
        self.money+=A.money
class B(A):
    def display(self):
        self.money+=2000
        print(self.money)
obj=B()
obj.display()
```

What will be output for the following code?

```
class A:
    a=20
    def __init__(self):
        self.a+=10
class B(A):
    a=15
    def buy(self):
        pass
class C(B,A):
    a=30
    def buy(self):
        print(self.a)
obj=C()
obj.buy()
```

What will be output for the following code?

```
class A:
    money=100000
    def buy(self):
        super().buy()
        self.money=5000
class B(A):
    def buy(self):
        self.money=3000
class C(A):
```

money=30000
    def buy(self):
        self.money=10000
class D(B,C):
    B.money+=C.money
    def buy(self):
        return self.money
obj=D()
print(obj.buy())	

What will be output for the following code?
class A:
    def __init__(self):
        super().__init__()
        print("1",end=",")
class B(A):
    def __init__(self):
        super().__init__()
        print("2",end=",")
class C(object):
    def __init__(self):
        super().__init__()
        print("3",end=",")
class D(C,B):
    def __init__(self):
        super().__init__()
        print("4",end=",")
obj=D()	

What will be output for the following code?
import numpy as np
a=np.array(5)
b=np.array([5,7,8])
print(b+a)	

What will be output for the following code?
import numpy as np
arr = np.array([[1,2,3,4,5], [6,7,8,9,10]])
print(arr[1, -3])	

What will be output for the following code?
import numpy as np
arr = np.array([[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]])
print(arr[1, 1, 2])	

rom abc import ABC,abstractmethod
class Father(ABC):
 	
    def display(self):
        pass
    def play(self):
        print('Abstract class')
class Son(Father):
    def play(self):
        print('Child class')
A=Son()
A.play()	

There is an array of scores of 5 Batsmen in 4 T20 Matches. Which is given below.
Scores= [[13, 10, 9, 33],
[63, 46, 90, 42],
[39, 76, 13, 29],
[82, 9, 29, 78],
[67, 61, 59, 36]]
Further you are asked to perform below tasks.
(i). Add scores of every batsman of 5th Match given below in the same array and print the array.
Match_6= [41, 87, 72, 36, 92]

(ii). Add two new batsmen's scores in respective 5 T20 Matches in the array created in task (i) above and print the array.
Batsman_6= [77, 83, 98, 95, 89]
Batsman_7= [92, 71, 52, 61, 53]

(iii). Add extra column with sum of all 5 T20 Matches' scores of each batsman in the array created in task (ii) and print the final array.

Note: Use Numpy module for all the Arrays given above.

Using the final array created in task(iii) above, generate graphs mentioned below:

(a). Make a line chart of Total Scores of each batsman which is stored in last column of final array v/s No. of Batsman. Use dashed line in graph, with black color. Give label on x-axis as "No. of Batsman" and label on y-axis as "Scores". Give title to the chart as "Leader Board" with bold fonts.

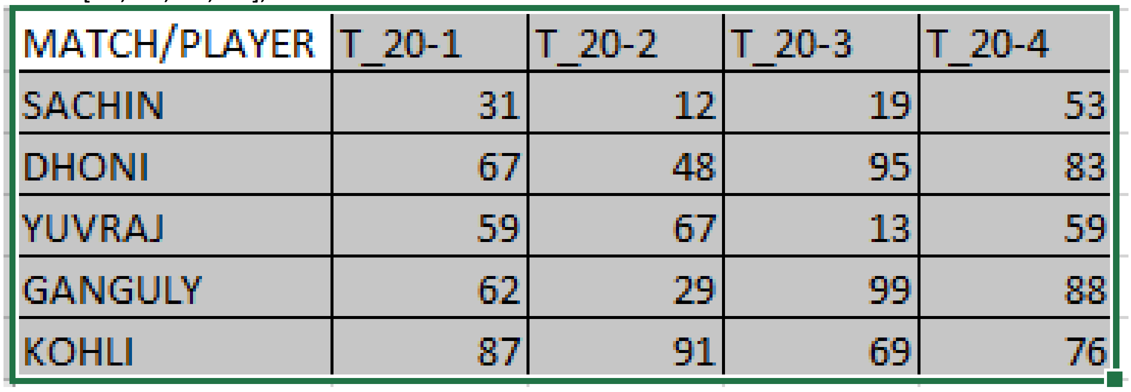
(b). Make one Bar chart of scores of Batsman_1 and Batsman_2 for all 5 T20 matches. Give color for bars of Batsman_1 as Purple and for Batsman_2 Dark red. Also show required legend in bar chart.

(c). Make a pie chart of Total Scores of each batsman which is stored in last column of final array. Show the pie chart with exploded view of all pieces with 0.1 amount. Also display percentage in the pie chart. Also show required legend for pie chart.

Note: Passing the values using numpy Array Slicing from array created in task(iii) for creating graphs above is compulsory.

There is an array of scores of 5 Batsmen in 4 T20 Matches. Which is given below.

```
Scores= [[31, 12, 19, 53],
         [67, 48, 95, 83],
         [59, 67, 13, 59],
```



MATCH/PLAYER	T_20-1	T_20-2	T_20-3	T_20-4
SACHIN	31	12	19	53
DHONI	67	48	95	83
YUVRAJ	59	67	13	59
GANGULY	62	29	99	88
KOHLI	87	91	69	76

1. Find the maximum score in T_20-3 and print it (use only the numpy module)
2. Find the minimum score of YUVRAJ and print it (use only the numpy module)
3. Add an extra column with the sum of all 4 T20 Matches' scores of each batsman in the array created and print it. (use only the numpy module)

Create a program using the instructions given below:

1. Create a constructor in all three classes (Respondent, Manager and Director) which takes the id and name as input and initializes two additional variables, rank and free. rank should be equal to 3 for Respondent, 2 for Manager and 1 for Director. free should be a boolean variable with value True initially. (1 mark)

2. Implement rest of the methods in all three classes in the following way: (2 marks)

a. receive_call(): prints the message, "call received by (name of the employee)" and sets the free variable to False.

b. end_call(): prints the message, "call ended" and sets the free variable to True.

c. is_free(): returns the value of the free variable

d. get_rank(): returns the value of the rank variable

3. Create a class Call, with a constructor that accepts id and name of the caller and initializes a variable called assigned to False. (0.5 marks)

4. Create a class CallHandler, with three lists, respondents, managers and directors as class variables. (0.5 marks)

5. Create an add_employee() method in CallHandler class that allows you to add an employee (an object of Respondent/Manager/Director) into one of the above lists according to their rank. (1 mark)

6. Create a dispatch_call() method in CallHandler class that takes a call object as a parameter. This method should find the first available employee starting from rank 3, then rank 2 and then rank 1. If a free employee is found, call its receive_call() function and change the call's assigned variable value to True. If no free employee is found, print the message: "Sorry! All employees are currently busy." (2 marks)

7. Create 3 Respondent objects, 2 Manager objects and 1 Director object and add them into the list of available employees using the CallHandler's add_employee() method. (1 mark)

8. Create a Call object and demonstrate how it is assigned to an employee. (1 mark)