SHEET OF EXERCISES:

Part A- Vectors:

Ex 1: Write a program that does the following:

- Declares a vector V1 of size n (n is given as input from the user)
- Declares another vector V2 of size 0
- Stores in V2 the cumulative sum of V1
- Prints , using an iterator, the values stored in V2

Example :

Input:

```
4
1 2 4 2
```

Output:

```
1 3 7 9
```

Ex 2: A list of numbers is said to be Beautiful if the sum of the greatest 3 elements and the smallest 3 elements is strictly between a and b. It's guaranteed that a<b, and that the number of elements given by the user is at least 6.

Input format:

- An integer n (n>=6)
- n integers separated by spaces
- two integers a and b such that a<b

Output:

Print "Beautiful" or "Not Beautiful" depending on the input.

Example:

Input:

```
10

4 2 6 5 6 5 1 3 3 6

20 30
```

Output:

```
Beautiful
```

Explanation: The smallest and largest three elements are respectively (1,2,3) and (6,6,6). Their sum is 1+2+3+6+6+6=24 where 20<24<30.

Ex 3: Write a program that:

- stores in a vector n integers given by the user (where n is given by the user)
- accepts from the user an integer q representing the number of queries
- for each query, the user must enter an integer x and the program must report the following information:
    - If the element is found in the list or not
    - The smallest element in the list that is strictly greater than x
    - The smallest element in the list that is greater than or equal to x
    - The greatest element in the list that is strictly less than x
    - The greatest element in the list that is less than or equal to x

Remarks:

1- You may use another vector if needed

2- If some required information isn't in the list, print "Doesn't exist"

Example:

Input:

```
6

4 2 1 5 5 1

1

5
```

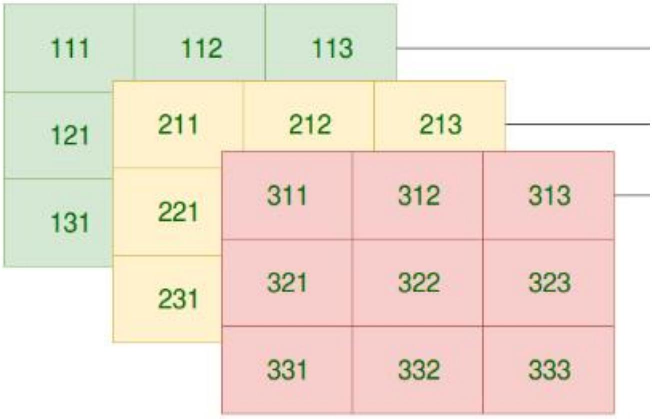
Output:

```
The element is found in the list

Doesn't exist

5

4

5
```

Ex 4: (Advanced) Write a program that :

- Creates 2 three-dimensional matrix using vectors
- Let the user fill the user fill the matrices as shown below
- Declare a third 3-dimensional matrix and store in it the sum of the previously defined matrices.
- Calculate, using iterators, the sum of all elements of the third matrix and display it on the screen.

Example:



Suppose that the two matrices given by the user are the same, and are equal to the one represented in the figure above. The input would be like this:

Input:

```
311 312 313
321 322 323
331 332 333

211 212 213
221 222 223
231 232 233

111 112 113
121 122 123
131 132 133

311 312 313
321 322 323
331 332 333

211 212 213
221 222 223
231 232 233

111 112 113
121 122 123
131 132 133
```

Output:

```
11,988
```

By Adel Haj Hassan