1D array search

Create a class 'ArraySearch' with the following methods:

BinarySearch(): Using *binary search*, this method should find the specified element (say x) in a given 1-D array. It is given that the elements of the array are sorted in ascending order. If the element (x) is present in the array, the method should return the largest possible index of the element in the array. If the element is not present, it should return the largest index i such that vec[i]$< x$. If no such index i can be found, the method should return -1.

The method should have following arguments:

- vec: The input vector.
- x: The element to be searched.
- start,end: The search for the element (x) in the vector must be limited in [start,end)
- flag: A boolean flag is to be set to true if the element is found, false otherwise

LinearSearch(): This method should find the specified element in a given 1-D array using *sequential search* and return the largest possible index of the element, if present. If the element is not present, it should return largest index i, such that vec[i] $< x$. If no such index i can be found, it should return -1.

The method should have following arguments:

- vec: The input vector.
- x: The element to be searched.
- start,end: The search for the element (x) in the vector must be limited in [start,end)
- flag: A boolean flag is to be set to true if the element is found, false otherwise

Create a class 'Client' with the following methods:

lower_bound(): Given a sorted 1-D array in ascending order, this method should use the *BinarySearch()* method to find the largest index i such that vec[i] $\leq$ x in a given range. If no such index i is found, return -1.

The method should have following arguments:

- vec: The input vector.
- x: The element to be searched.
- start,end: The search for the element (x) in the vector must be limited in [start,end)
- flag: A boolean flag is to be set to true if the element is found, false otherwise

upper_bound(): Given a sorted 1-D array in ascending order, this method should use the *BinarySearch()* method to find the smallest index i such that vec[i]$> x$ in a given range. If no such index i is found, return -1.

The method should have following arguments:

- vec: The input vector.
- x: The element to be searched.
- start,end: The search for the element (x) in the vector must be limited in [start,end)
- flag: A boolean flag is to be set to true if the element is found, false otherwise

Input Format

```
N
vec
BS element start end
LS element start end
LB element start end
UB element start end
```

'N' is the size of the vector.

'vec' is a vector sorted in ascending order.

BS : use BinarySearch()

LS : use LinearSearch()

LB : use lower_bound()

UB : use upper_bound()

Constraints

$1 \leq N \leq 10^6$

$-2^{31} \leq vec[i] \leq 2^{31}$

Output Format

Print the output of each query in a separate line.

Sample Input 0

```
18
10 10 11 13 18 23 23 48 50 54 68 77 77 77 84 98 98 99
BS 68 0 18
LS 84 2 18
LB 77 5 13
UB 98 0 18
```

Sample Output 0

```
10
14
12
17
```