Stock span Problem.

Problem Statement:-

The stock span problem is a financial problem where we have a series of n daily price quotes for a stock and we need to calculate span of stock's price for all n days

The span S_i of the stock's price on a given day i is defined as the maximum number of consecutive day just before the given day, for which the price of stock on the current day is less than or equal to its price on the given day.

For example if an array of 7 days price is given as {100, 80, 60, 70, 60, 75, 85}

then span values for corresponding 7 days are {1, 1, 1, 2, 1, 4, 6}

arr : | 100 | 80 | 60 | 70 | 60 | 75 | 85 |

4

- for any day check the smaller (consequtive) before day

100	80	60	70	60	75	85
1	1	1	2	1	4	6

op

vector v

Consequtive Smaller or Equal to

arr[]: 100 80 60 70 60 75 85

- Nearest greater to left,

0	1	2	3	4	5	6
100	80	60	70	60	75	85
-1	0	1	1	3	1	0

1 2-1=1 3-1=2 4-3=1 5-1=4 6-0=6

100 80 80 70 80 100

NGL index - i

- we need a vector to store index

0	1	2	3	4	5	6
100	80	60	70	60	75	85

Index v

-1	0	1	1	3	1	0

op 1 1 1 2 1 4 6

Code NGL Index.

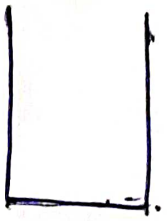
```
vector<int> v;
stack<pair<int, int>> s;

for(int i=0; i<size; i++)
{
    if(s.size()==0)
    {
        v.push_back(-1);
    }
    else if(s.size()>0 && s.top().first > arr[i])
    {
        v.push_back(st.top().second);
    }
    else if(s.size()>0 && s.top().first <= arr[i])
    {
        while(s.size()!=0 && s.top().first <= arr[i])
        {
            s.pop();
        }
        if(s.size()==0)
            v.push_back(-1)
        else
            v.push_back(s.top().second);
    }
    s.push({arr[i], i});
}
return v;
```

→ NGL Index (use in that way)

	0	1	2	3	4	5	6
Arr:	100	80	60	70	60	75	85
vector v	-1	0	1	1	3	1	0
Output i-v	1	1	1	2	1	4	6

Nearest greater to left → index



[Stock span Problem → Consequtive smaller or equal Before it → NGL Index → i-NGL Index]

after getting index of nearest greater left

	0	1	2	3	4	5	6
V:	-1	0	1	1	3	1	0
i-V[i]	1	1	1	2	1	4	6

```
for (int i=0; i < V.size(); i++)
{
    V[i] = i - V[i];
}
return V;
```