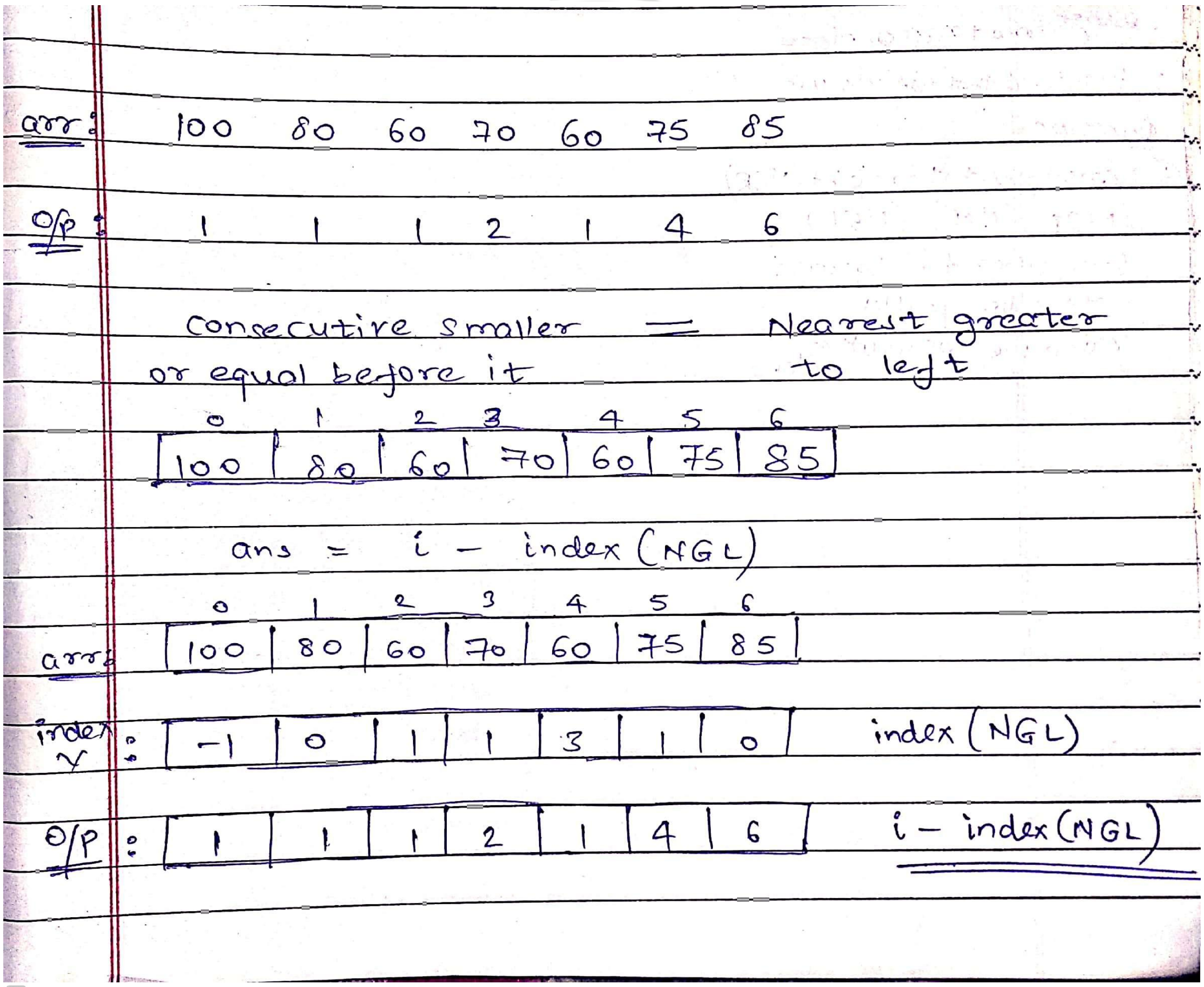
Stock Span Problem

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 the span of stocks price for all n days.

The span Si of the stock's price on a given day i is defined as the maximum number of consecutive days just before the given day, for which the price of the 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 prices is given as {100, 80, 60, 70, 60, 75, 85}, then the span values for corresponding 7 days are {1, 1, 1, 2, 1, 4, 6}.



arr:	100	80	60	70	60	75	85
O/P:	1	1	1	2	1	4	6

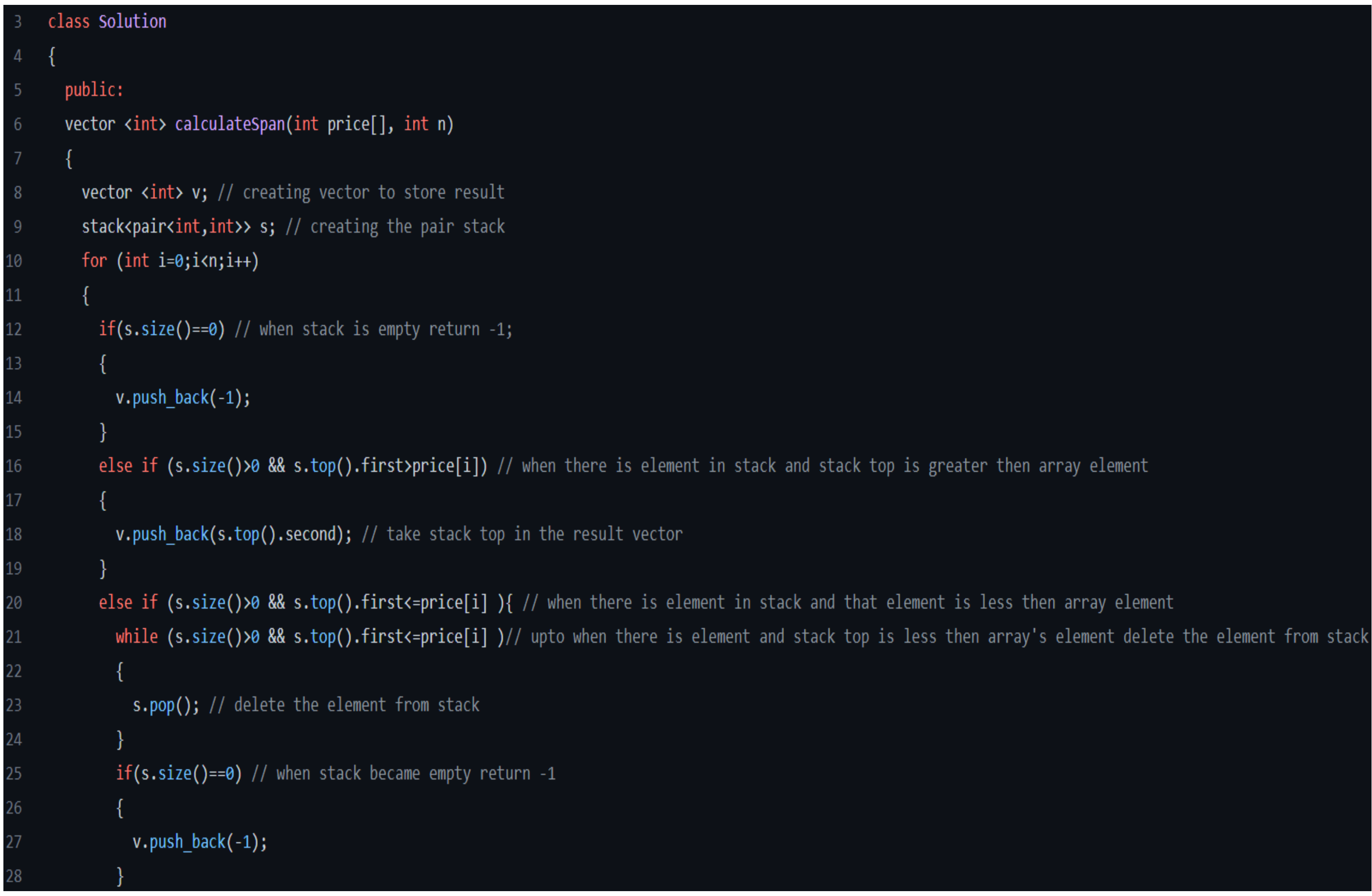
consecutive smaller or equal before it = Nearest greater to left

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

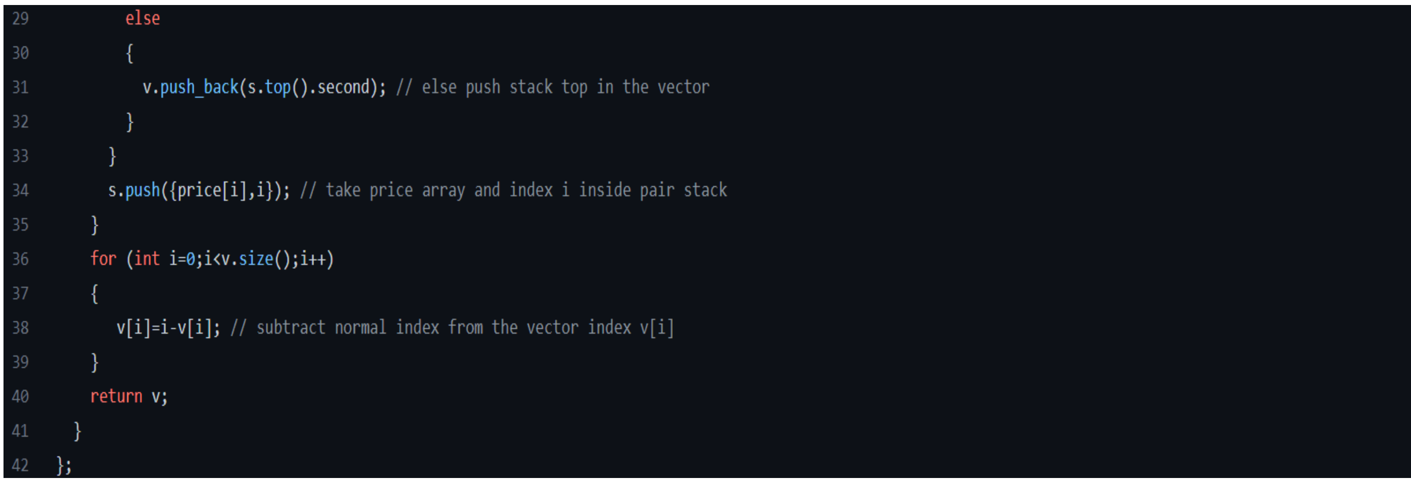
ans = i − index (NGL)

	0	1	2	3	4	5	6	
arr:	100	80	60	70	60	75	85	
index v :	−1	0	1	1	3	1	0	index (NGL)
O/P :	1	1	1	2	1	4	6	i − index (NGL)

CODE :



```cpp
class Solution
{
  public:
  vector <int> calculateSpan(int price[], int n)
  {
    vector <int> v; // creating vector to store result
    stack<pair<int,int>> s; // creating the pair stack
    for (int i=0;i<n;i++)
    {
      if(s.size()==0) // when stack is empty return -1;
      {
        v.push_back(-1);
      }
      else if (s.size()>0 && s.top().first>price[i]) // when there is element in stack and stack top is greater then array element
      {
        v.push_back(s.top().second); // take stack top in the result vector
      }
      else if (s.size()>0 && s.top().first<=price[i] ){ // when there is element in stack and that element is less then array element
        while (s.size()>0 && s.top().first<=price[i] )// upto when there is element and stack top is less then array's element delete the element from stack
        {
          s.pop(); // delete the element from stack
        }
        if(s.size()==0) // when stack became empty return -1
        {
          v.push_back(-1);
        }
```



```cpp
        else
        {
          v.push_back(s.top().second); // else push stack top in the vector
        }
      }
      s.push({price[i],i}); // take price array and index i inside pair stack
    }
    for (int i=0;i<v.size();i++)
    {
      v[i]=i-v[i]; // subtract normal index from the vector index v[i]
    }
    return v;
  }
};
```