STOCK SPANNER

In this problem, we are supposed to implement a Stock Span class with a next function. Everytime the next function is called, we have to take in a stock price and output the no. of consecutive days stock price previously was less than or equal to current price.

Brute force solution involves using a dynamic array such that in case next is called, we can just reverse iterate to check. Time Complexity is O(N) and so is the space

Optimal Solution involves using logic of previous greater element and a stack. Everytime new element arrives, we just subtract previous greater element's index from its index.

Pseudocode :

```
class stockSpanner {
  public :
  stack < pair < int , int >> st ;
  int i ;
  stockSpanner () {
      i = 0 ;
  }
  next(int x) {
```

```
        while (!st.empty() && x >= st.top().first){
            st.pop();
        }
        if (st.empty()) {
            pair<int, int> y = (x, i++);
            st.push(y);
            return i;
        } else {
            pair<int, int> y = (x, i);
            st.push(y);
            return i - st.top().second;
        }
    }
};
```