Deepanshu

2019CS50427

Page No.
Date

Homework 2.1

- Methods defined: New, Insert, Delete, IsIn
- Other methods :- 1. IsEmpty (S: ADT) : Boolean
  2. size (S: ADT) : integer
- Additional methods (taken from C++ STL):
  lowerBound, upperBound, largest, smallest

Homework 2.2

Q what are the drawbacks of using arrays for the implementation of stack?

Ans 1. we have to be concerned about StackFullException while implementing push(). Other implementations may not require such exception handling.

2. we have to specify size beforehand.

2a. If predicted/expected N is small, then we may have to reallocate memory to a new array of sufficient size which is an inefficient method.

2b. If predicted/ expected N is large, then memory is getting wasted.

3. Even after changing array size once (as in point 2a), we may have to repeat that procedure in future.
we do not have flexibility on size.

<u>Homework 2.3</u>

10 6 2 4 2 5 8 (consistent with diagram)
(1) (1) (1) (2) (1) (4) (6) Dr. Naveen

~~Algorithm int findspan(~~ [illegible]

```
Algorithm findspan( int ar[], int n)
  // 0 based indexing [illegible] int span[n];
  i ← 0
while i<n
    do j ← i-1
        while( ar[j] <= ar[i] and j>=0)
                j--
        [illegible].
        span[i] = i - j
return span
```

```
for i in range n:
    while (j >0 and
        ar[j] <= ar[i])
        j--
    i-j .
```

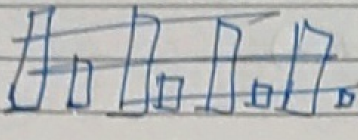
complexity = $O(n^2)$

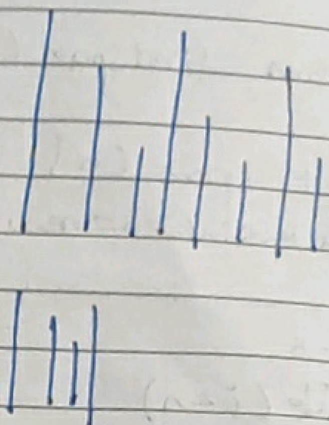
Homework 2.4

```
Algorithm findspan(int ar[], int n)
        int span[n], count=0
      Stack <int> s = new stack()
      i <- 0
      while (i<n and not(s.isEmpty()))
         do
             if (ar[s.top()] > ar[i])
                     break          // got the larger bar
             s.pop()
     if s.isEmpty()                // this is the maximum height
           span[i] = i+1                bar
     else :
             span[i] = i - s.top()
               s.push(i)              // pushing to stack
     return span
```

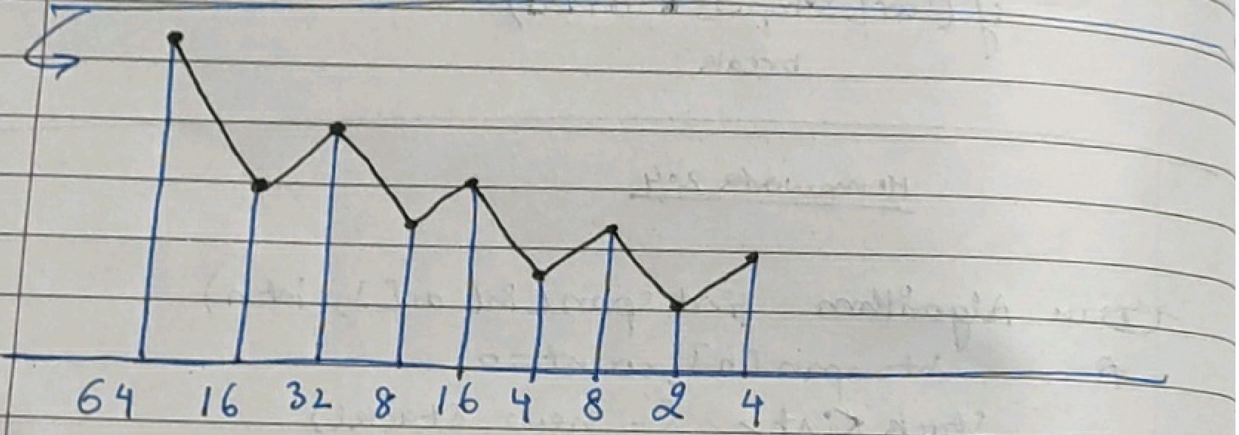
Note : Define Stack<int> s = new stack() as statement which creates a new stack that can store integers.

Homework ~~2-4~~ 2-5





1024 1014 1004... 514 1020 1010 .....



Some thing of this form will make the inner loop execu
roughly $\frac{n}{2}$ times i.e O(n)