Design a data-structure**SpecialStack**that supports all the stack operations like **push()**, **pop()**,**isEmpty()**, **isFull()** and an additional operation **getMin()** which should return **minimum**element from the SpecialStack. Your task is to **complete all the functions**, using stack data-Structure.

**Example 1:**

**Input:**

Stack: 18 19 29 15 16

**Output:** 15

**Explanation:**

The minimum element of the stack is 15.

**Expected Time Complexity:**O(N) for getMin, O(1) for remaining all 4 functions.  
**Expected Auxiliary Space:**O(1) for all the 5 functions.

**Constraints:**  
1 ≤ N ≤ 104

**Company Tags**

[**Paytm**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Paytm) [**VMWare**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=VMWare) [**Flipkart**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Flipkart) [**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) [**FactSet**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=FactSet) [**MakeMyTrip**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=MakeMyTrip) [**Walmart**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Walmart) [**Goldman Sachs**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Goldman%20Sachs)[**Intuit**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Intuit) [**Adobe**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Adobe)[**SAP Labs**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=SAP%20Labs) [**Linkedin**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Linkedin)

**Topic Tags**

[**Stack**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Stack)[**STL**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=STL)[**Data Structures**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Data%20Structures)

Java code

import java.util.Scanner;

import java.util.Stack;

class SpeStack{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int t=sc.nextInt();

while(t-->0){

int n=sc.nextInt();

Stack<Integer> s=new Stack<>();

CodingMaxima g=new CodingMaxima ();

while(!g.isEmpty(s)){

g.pop(s);

}

while(!g.isFull(s,n)){

g.push(sc.nextInt(),s);

}

System.out.println(g.min(s));

}

}

}

class CodingMaxima{

public int min=Integer.MAX\_VALUE;

public void push(int a,Stack<Integer> s)

{

min=Math.min(min ,a);

s.push(a);

}

public int pop(Stack<Integer> s)

{

if(!s.isEmpty())

return s.pop();

else

return -1;

}

public int min(Stack<Integer> s)

{

return min;

}

public boolean isFull(Stack<Integer>s, int n)

{

return (s.size()>=n);

}

public boolean isEmpty(Stack<Integer>s)

{

return s.isEmpty();

}

}