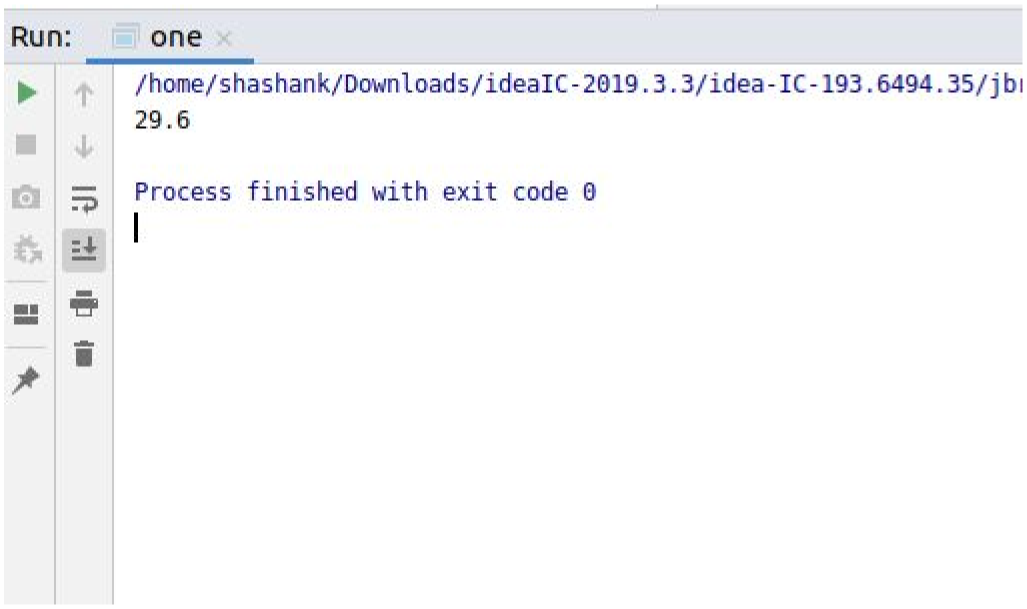
JAVA EXERCISE 4 (collection)

1. **Write Java code to define List . Insert 5 floating point numbers in List, and using an iterator, find the sum of the numbers in List.**

```
import java.util.*;


class one {
  public static void main(String[] args)
  {
     List<Double> list = new ArrayList<Double>();


     list.add(1.5);
     list.add(5.7);
     list.add(6.3);
     list.add(7.6);
     list.add(8.5);
     System.out.println(sum(list));
  }


  public static Double sum(List<Double> list)
  {
     Iterator<Double> it = list.iterator();


     double res = 0;
     while (it.hasNext()) {


        double num = it.next();
        res += num;


     }
```

```
    return res;
  }
}
```

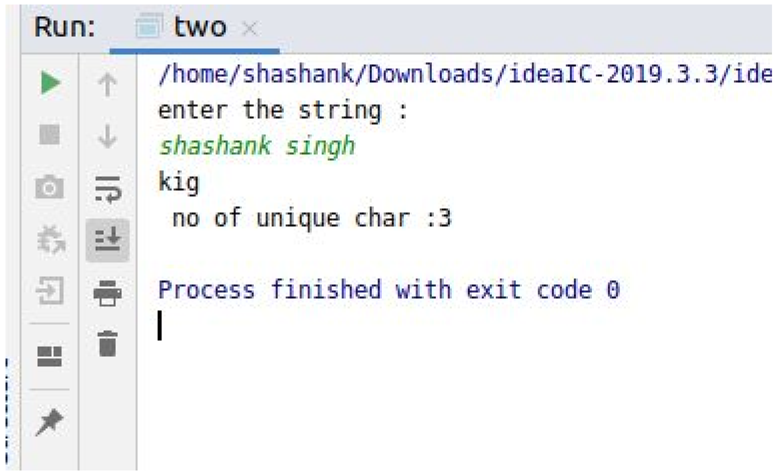

Run: one

```
/home/shashank/Downloads/ideaIC-2019.3.3/idea-IC-193.6494.35/jb
29.6

Process finished with exit code 0
```

2. Write a method that takes a string and returns the number of unique characters in the string.

```
import java.util.Arrays;

import java.util.Scanner;


public class two{


  static final int MAX_CHAR = 256;

  static void uniqchar(String str)

  {

    int n = str.length();

    int c=0;

    int[] count = new int[MAX_CHAR];

    int[] index = new int[MAX_CHAR];
```

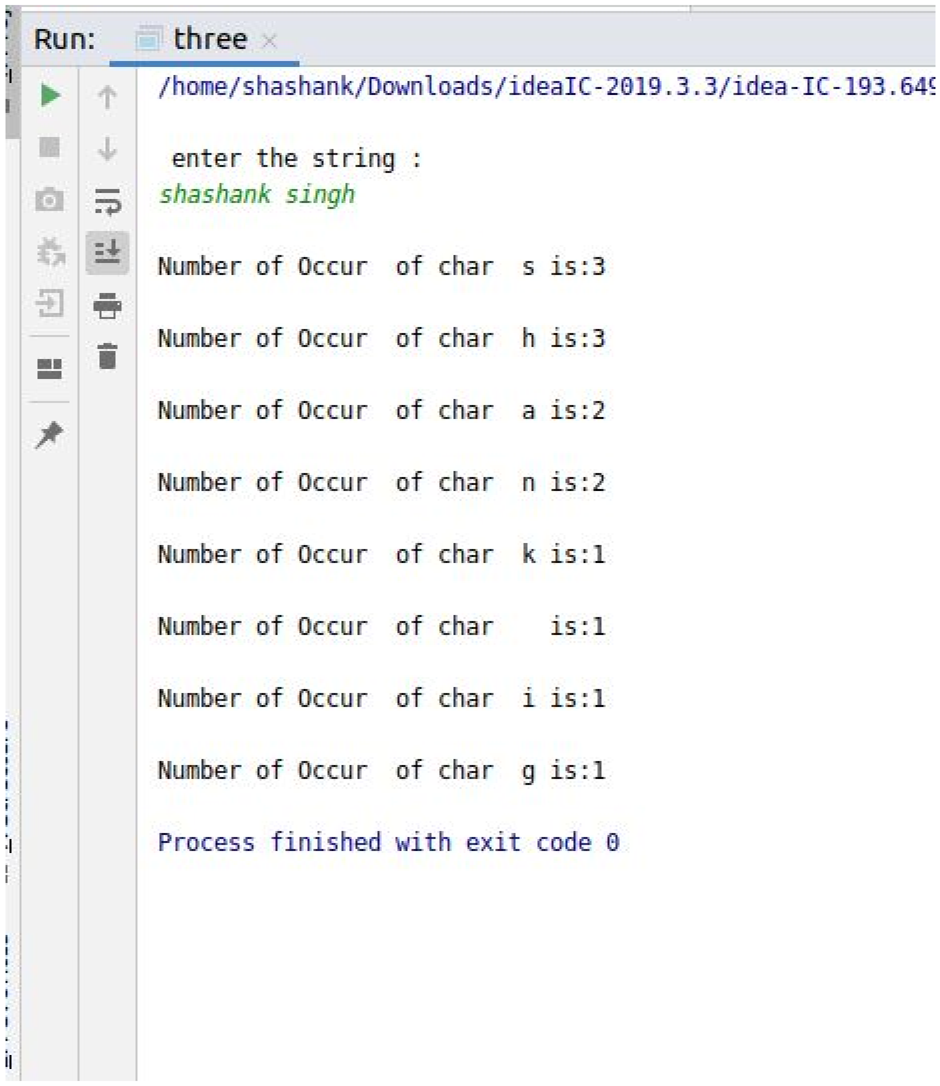
```
        for (int i = 0; i < MAX_CHAR; i++)

        {

            count[i] = 0;

            index[i] = n;

        }

        for (int i = 0; i < n; i++)

        {

            char x = str.charAt(i);

            ++count[x];

            if (count[x] == 1 && x !=' ')

                index[x] = i;

            if (count[x] == 2)

                index[x] = n;

        }

        Arrays.sort(index);


        for (int i = 0; i < MAX_CHAR && index[i] != n; i++) {

            System.out.print(str.charAt(index[i]));

            c++;

        }

        System.out.println("\n no of unique char :"+c);


    }

    public static void main(String args[])

    {

        Scanner sc=new Scanner(System.in);

        System.out.println("enter the string :");

        String str = sc.nextLine();

        uniqchar(str);

    }

}
```



```
/home/shashank/Downloads/ideaIC-2019.3.3/ide
enter the string :
shashank singh
kig
 no of unique char :3

Process finished with exit code 0
```

3. Write a method that takes a string and print the number of occurrence of each character characters in the string.

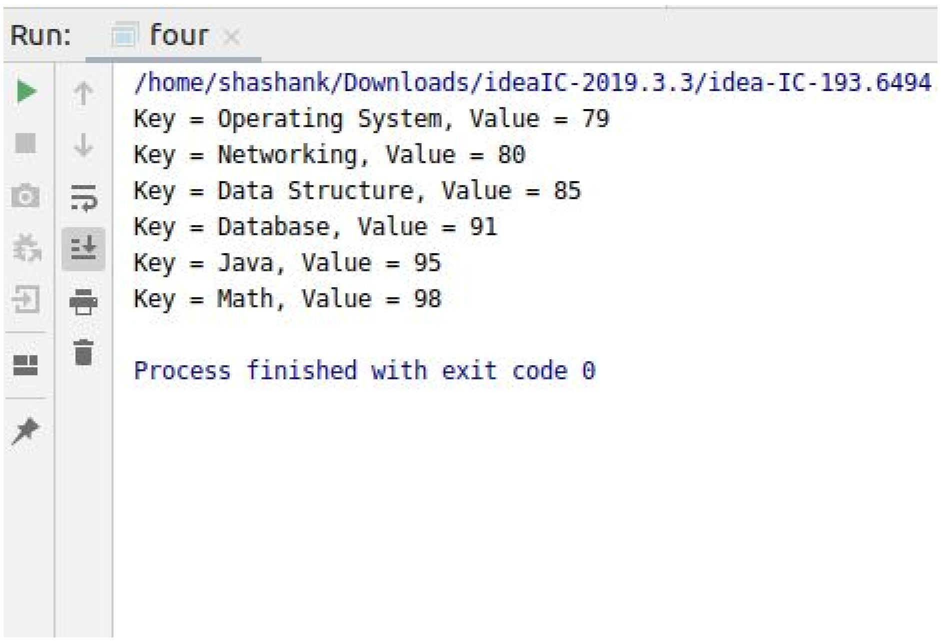
```
import java.util.Scanner;
public class three{
  static final int MAX_CHAR = 256;
  static void occ(String str)
  { int count[] = new int[MAX_CHAR];
     int len = str.length();
     for (int i = 0; i < len; i++)
        count[str.charAt(i)]++;
     char ch[] = new char[str.length()];
     for (int i = 0; i < len; i++) {
        ch[i] = str.charAt(i);
        int find = 0;
        for (int j = 0; j <= i; j++) {
           if (str.charAt(i) == ch[j])
              find++;
        }
        if (find == 1)
           System.out.println("\nNumber of Occur  of char  " + str.charAt(i) + " is:" + count[str.charAt(i)]);
        }
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("\n enter the string :");
        String str = sc.nextLine();
        occ(str);
    }
}
```



```
/home/shashank/Downloads/ideaIC-2019.3.3/idea-IC-193.649

 enter the string :
shashank singh

Number of Occur  of char  s is:3

Number of Occur  of char  h is:3

Number of Occur  of char  a is:2

Number of Occur  of char  n is:2

Number of Occur  of char  k is:1

Number of Occur  of char    is:1

Number of Occur  of char  i is:1

Number of Occur  of char  g is:1

Process finished with exit code 0
```

4. Write a program to sort HashMap by value

```
import java.util.*;

import java.lang.*;

public class four{

  public static HashMap<String, Integer> sortByValue(HashMap<String, Integer> hm)

  { List<Map.Entry<String, Integer> > list = new LinkedList<Map.Entry<String, Integer> >(hm.entrySet());

     Collections.sort(list, new Comparator<Map.Entry<String, Integer> >() {

        public int compare(Map.Entry<String, Integer> o1, Map.Entry<String, Integer> o2)

        {

           return (o1.getValue()).compareTo(o2.getValue());

        }

     });

     HashMap<String, Integer> temp = new LinkedHashMap<String, Integer>();

     for (Map.Entry<String, Integer> aa : list) {

        temp.put(aa.getKey(), aa.getValue());

     }

     return temp;

  }

  public static void main(String[] args)

  {


     HashMap<String, Integer> hm = new HashMap<String, Integer>();

     hm.put("Math", 98);

     hm.put("Data Structure", 85);

     hm.put("Database", 91);

     hm.put("Java", 95);

     hm.put("Operating System", 79);

     hm.put("Networking", 80);

     Map<String, Integer> hm1 = sortByValue(hm);

     for (Map.Entry<String, Integer> en : hm1.entrySet()) {
            System.out.println("Key = " + en.getKey() + ", Value = " + en.getValue());
        }
    }
}
```



```
/home/shashank/Downloads/ideaIC-2019.3.3/idea-IC-193.6494
Key = Operating System, Value = 79
Key = Networking, Value = 80
Key = Data Structure, Value = 85
Key = Database, Value = 91
Key = Java, Value = 95
Key = Math, Value = 98

Process finished with exit code 0
```

5. Write a program to sort Employee objects based on highest salary using Comparator. Employee class{ Double Age; Double Salary; String Name

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Comparator;


class EmployeeSortBysal implements Comparator< Employee > {
  public int compare(Employee emp1, Employee emp2) {
      int value = 0;
      if (emp1.empSalary> emp2.empSalary)
          value = 1;
```

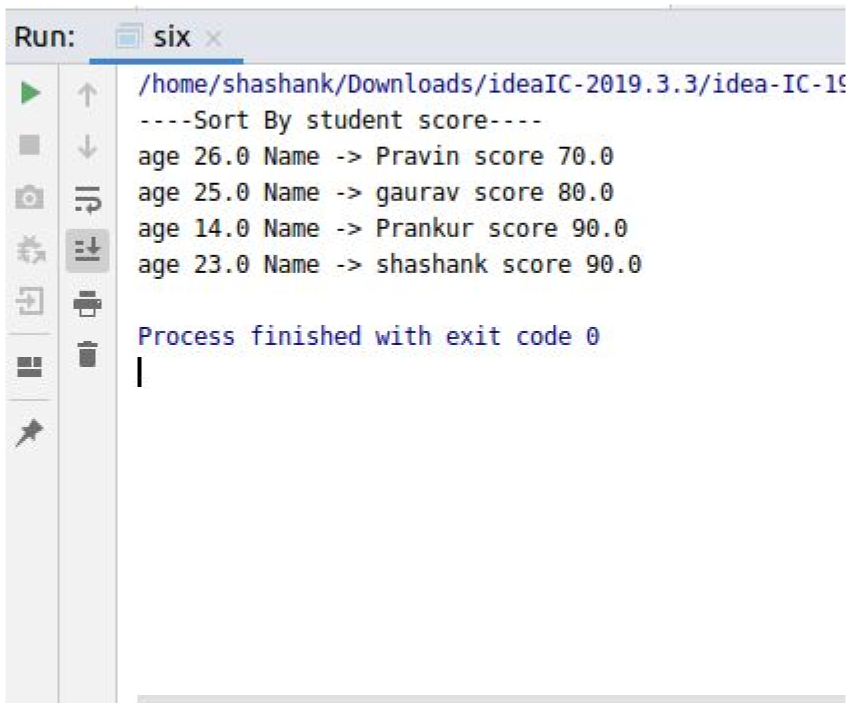
```
        else if (emp1.empSalary< emp2.empSalary)
            value = -1;
        else if (emp1.empSalary == emp2.empSalary)
            value = 0;

        return value;
    }
}
class Employee {
    public int empage;
    public String empName;
    public double empSalary;

    Employee(int empage, String empName, double empSalary) {
        this.empage = empage;
        this.empName = empName;
        this.empSalary = empSalary;
    }

}
public class five{

    public static void main(String[] args) {
        List <Employee> employees = new ArrayList <Employee> ();
        employees.add(new Employee(23, "shashank", 15000));
        employees.add(new Employee(14, "Prankur", 160000));
        employees.add(new Employee(25, "gaurav", 14000));
        employees.add(new Employee(26, "Pravin", 22000));

        System.out.println("----Sort By Employee sal----");
        Collections.sort(employees, new EmployeeSortBysal());
        printEmployees(employees);


    }
    public static void printEmployees(List <Employee> employees) {
        for (Employee e: employees) {
            System.out.println("Id->" + e.empage + " Name -> " + e.empName + " Salary-> " + e.empSalary);
        }
    }


}
```

6. <u>Write a program to sort the Student objects based on Score , if the score are same then sort on First Name . Class Student{ String Name; Double Score; Double Age</u>

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.Comparator;
class studSortByscore implements Comparator< stud > {
  public int compare(stud s1, stud s2) {
      int value = 0;
      if (s1.studscore> s2.studscore)
          value = 1;
      else if (s1.studscore<s2.studscore)
          value = -1;
      else if (s1.studscore == s2.studscore)
      {
          value= s1.studName.compareTo(s2.studName);
      }

      return value;
```

```
    }
}
class stud {
    public double studage;
    public String studName;
    public double studscore;

    stud(double studage, String studName, double studscore) {
        this.studage = studage;
        this.studName = studName;
        this.studscore = studscore;
    }
}
public class six{

    public static void main(String[] args) {
        List<stud> studs = new ArrayList<stud>();
        studs.add(new stud(23, "shashank ", 90));
        studs.add(new stud(14, "Prankur ", 90));
        studs.add(new stud(25, "gaurav ", 80));
        studs.add(new stud(26, "Pravin ", 70));

        System.out.println("----Sort By student score----");
        Collections.sort(studs, new studSortByscore());

        printstud(studs);

    }
    public static void printstud(List <stud>studs) {
        for (stud s:studs) {
            System.out.println("age " + s.studage + " Name -> " + s.studName + "score " + s.studscore);
        }
```

```
  }
}
```



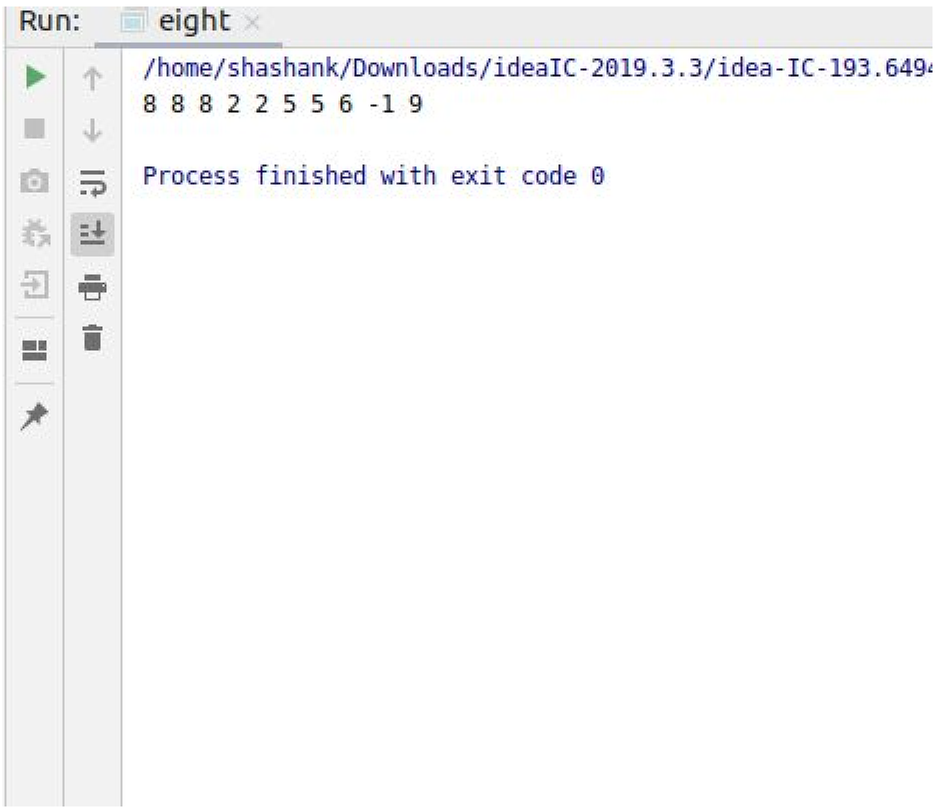
Run: six

```
/home/shashank/Downloads/ideaIC-2019.3.3/idea-IC-1
----Sort By student score----
age 26.0 Name -> Pravin score 70.0
age 25.0 Name -> gaurav score 80.0
age 14.0 Name -> Prankur score 90.0
age 23.0 Name -> shashank score 90.0

Process finished with exit code 0
```

7. Print the elements of an array in the decreasing frequency if 2 numbers have same frequency then print the one which came first.

```
import java.util.*;


class eight {
  public static StringBuffer sortbyfreq(int arr1[], int l1) {
      Map<Integer, Integer> countMap = getCountMap(arr1, l1);
      StringBuffer result = new StringBuffer();
      countMap.entrySet().stream()
              .sorted(Map.Entry.<Integer, Integer> comparingByValue().reversed())
              .forEach(e -> {
                  int key = e.getKey();
                  int val = e.getValue();
                  for (int i = 0; i < val; i++) {
                      result.append(key + " ");
```

```
                }
            });

        return result;
    }
    private static Map<Integer, Integer> getCountMap(int[] arr, int l1) {
        Map<Integer, Integer> countMap = new LinkedHashMap<>();
        for (int i = 0; i < l1; i++) {
            if (countMap.containsKey(arr[i])) {
                countMap.put(arr[i], countMap.get(arr[i]) + 1);
            } else {
                countMap.put(arr[i], 1);
            }
        }
        return countMap;
    }
    public static void main(String[] args){
        int a[] = { 2, 5, 2, 6, -1, 9, 5, 8, 8, 8 };
        System.out.println(sortbyfreq(a, a.length));
    }
}
```

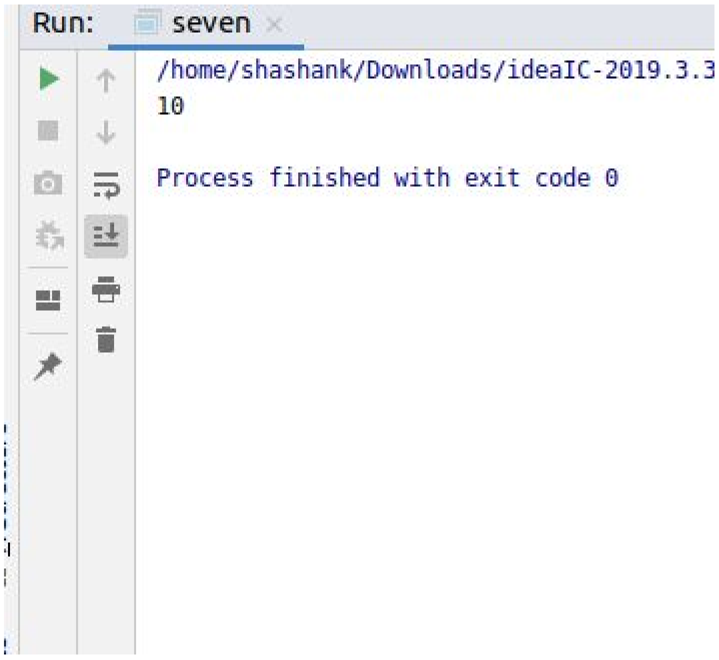


```
/home/shashank/Downloads/ideaIC-2019.3.3/idea-IC-193.649
8 8 8 2 2 5 5 6 -1 9

Process finished with exit code 0
```

8. **<u>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. (Expected complexity O(1))</u>**

```
import java.util.Stack;
class seven extends Stack<Integer>
{
  Stack<Integer> min = new Stack<>();
  void push(int x)
  {
     if(isEmpty() == true)
     {
        super.push(x);
        min.push(x);
     }
     else
     {
        super.push(x);
        int y = min.pop();
        min.push(y);
        if(x < y)
           min.push(x);
        else
```

```
            min.push(y);
        }
    }
    public Integer pop()
    {
        int x = super.pop();
        min.pop();
        return x;
    }
    int getMin()
    {
        int x = min.pop();
        min.push(x);
        return x;
    }
    public static void main(String[] args)
    {
        seven s = new seven();
        s.push(10);
        s.push(20);
        s.push(30);
        System.out.println(s.getMin());
    }
}
```



Run: seven

```
/home/shashank/Downloads/ideaIC-2019.3.3
10

Process finished with exit code 0
```