Next Smaller Element To The Right

1. You are given a number n, representing the size of array a.

2. You are given n numbers, representing elements of array a.

3. You are required to "next smaller element on the right" for all elements of array

4. Input and output is handled for you.

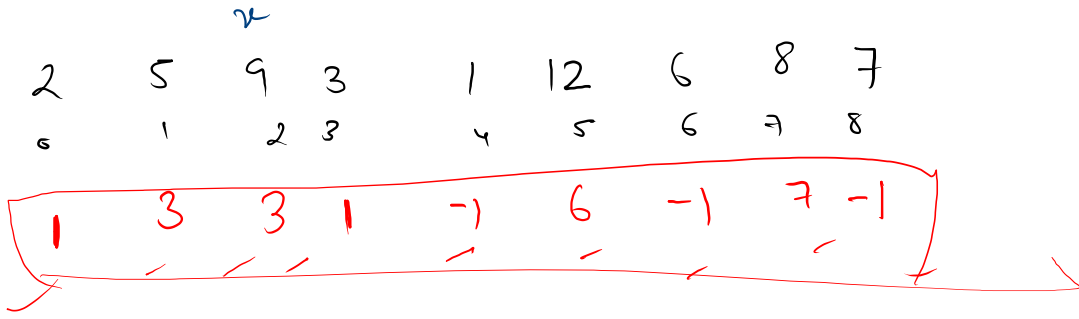
"Next smaller element on the right" of an element x is defined as the first element to right of x having value smaller than x. Note -> If an element does not have any element on it's right side smaller than it, consider -1 as it's "next smaller element on right"

Sample Input 0

```
9
2 5 9 3 1 12 6 8 7
```

Sample Output 0

```
1 3 3 1 -1 6 -1 7 -1
```



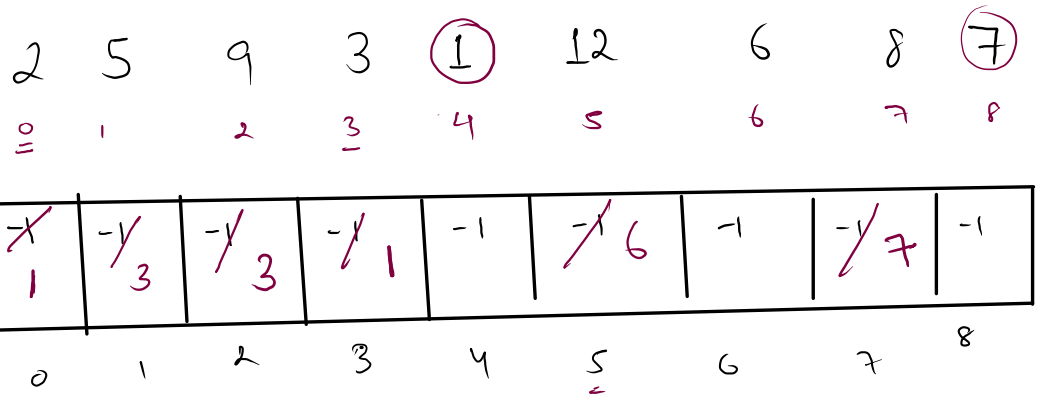
Sample Input 0

```
9
2 5 9 3 1 12 6 8 7
```

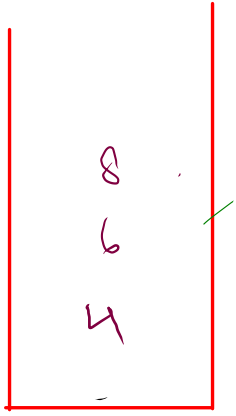
Sample Output 0

```
1 3 3 1 -1 6 -1 7 -1
```

ngor



idx for those whose ans not found



loop

if (A[st.peek()] > A[i])

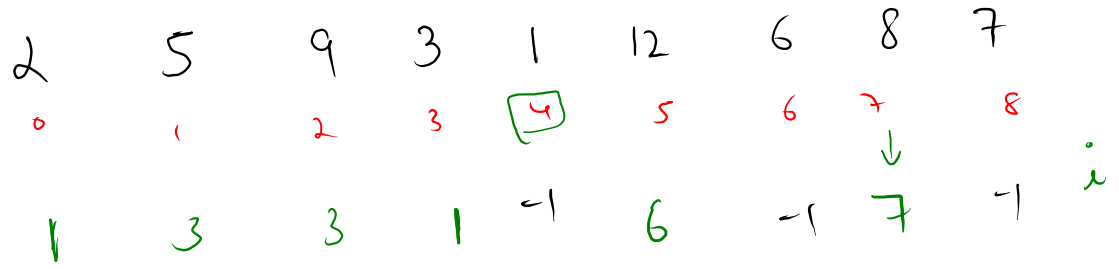
↳ ans[st.peek] = A[i]

Sample Input 0

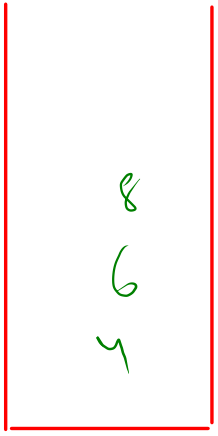
```
9
2 5 9 3 1 12 6 8 7
```

Sample Output 0

```
1 3 3 1 -1 6 -1 7 -1
```



idx (ans = 404)



A[7] > A[8]

loop

A[st.peek()] > A[i]

↳ then A[i] can become ans of st.peek

ans[st.peek] = A[i]

age = 52

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [] A = new int[n];
        int [] ans = new int[n];
        for(int i = 0; i < n; i++){
            A[i] = scn.nextInt();
            ans[i] = -1;
        }

        Stack<Integer> st = new Stack<>();   // idx of unresolved ele
        st.push(0);
        for(int i = 1; i < n; i++){
            while(st.size() != 0 && A[st.peek()] > A[i]){
                int idx = st.pop();
                ans[idx] = A[i];
            }
            st.push(i);
        }

        for(int ele : ans){
            System.out.print(ele + " ");
        }
    }
}
```

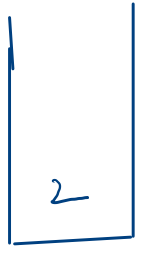
n = 6

2 4 1 7 5 6

0 1 2 3 4 5

-1 → 1	-1 → 1	-1	-1	-1	-1
0	1	2			

i



A[0] > A[2]

2 > 1

idx = 0

ans[0] = A[2]

= 1

nsor

ngor

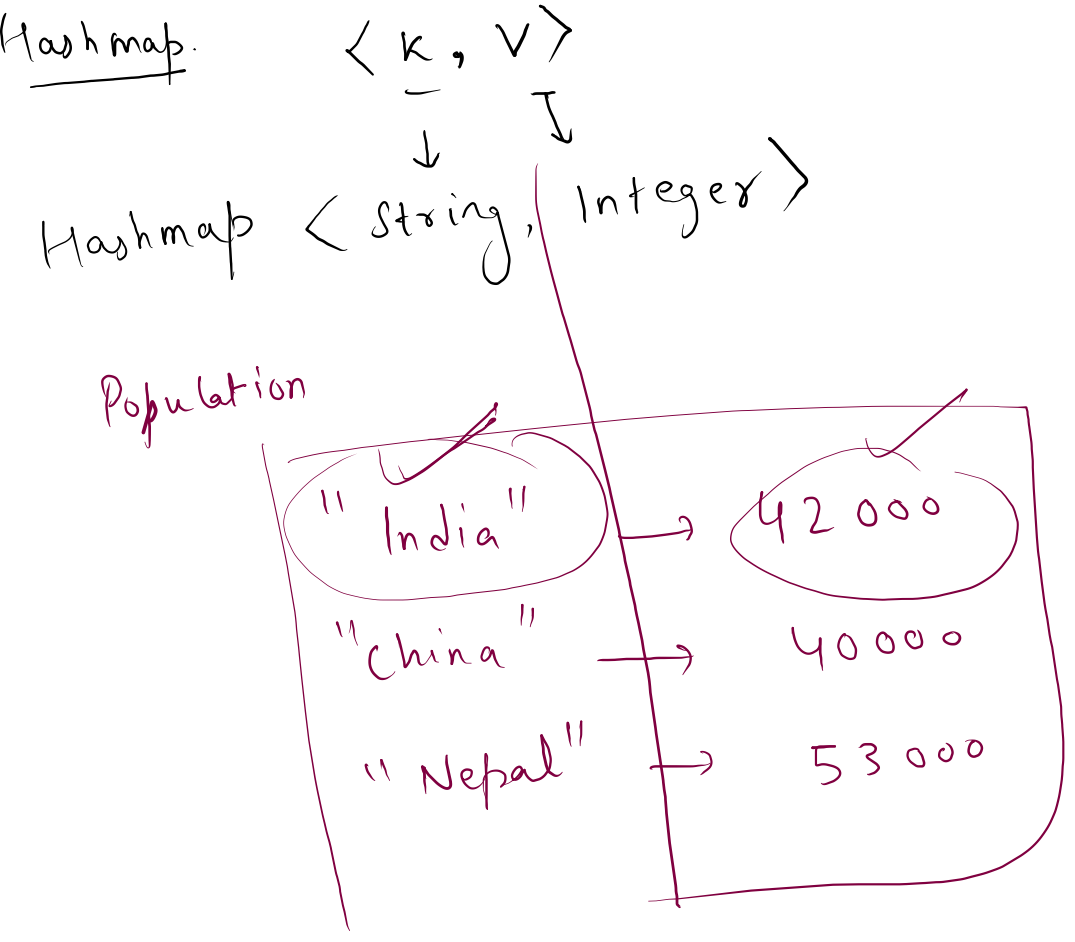
nsol

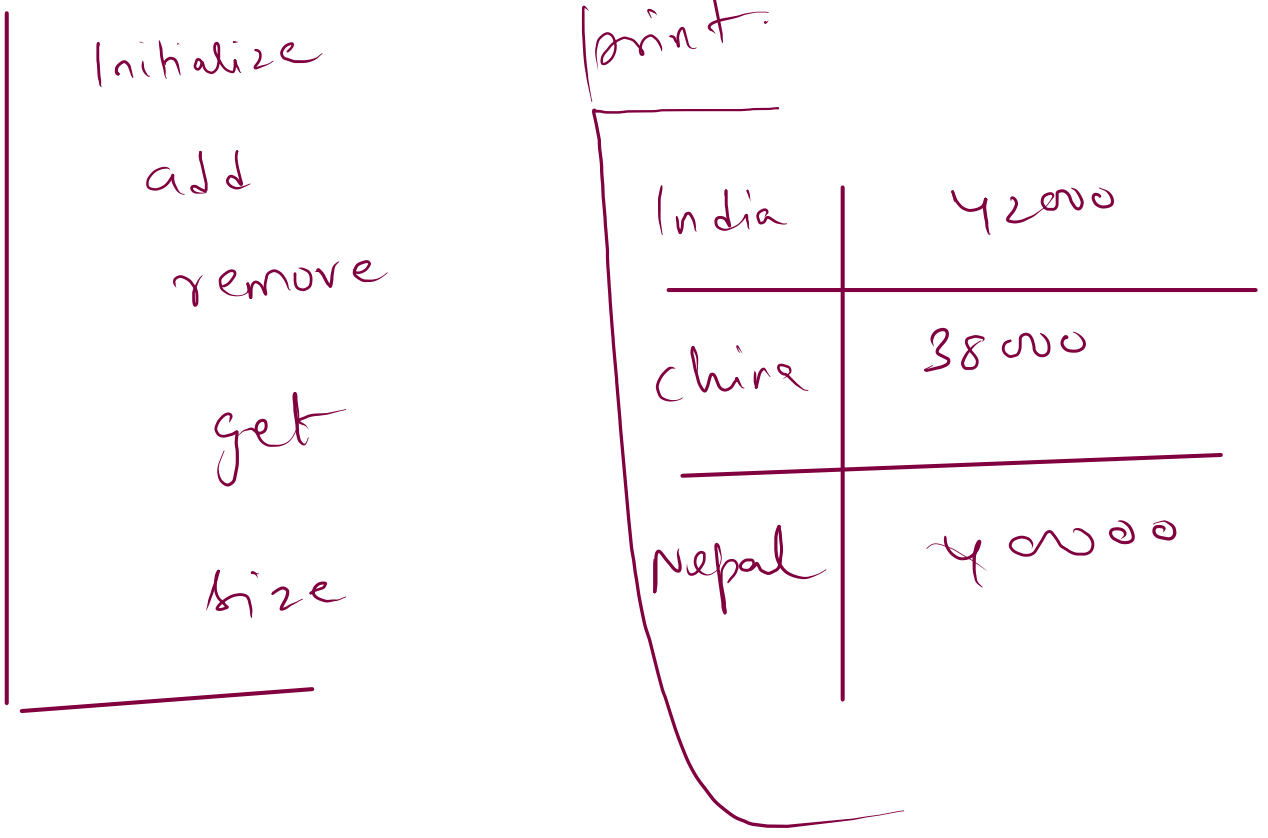
nsor

*

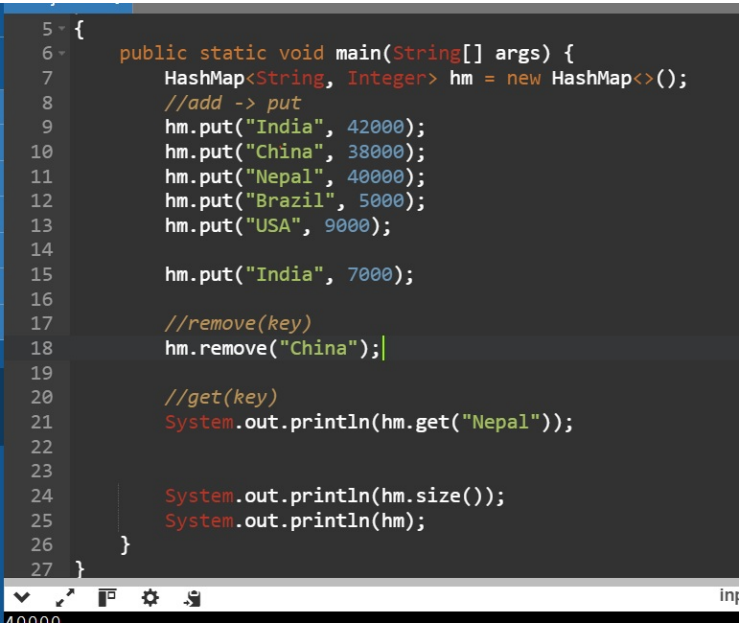
- next smaller on right
- next smaller on left
- next greater on right
- next greater on left

Hashmap.


<K, V>
Hashmap <String, Integer>
Population
"India"
42000
"China"
40000
"Nepal"
53000

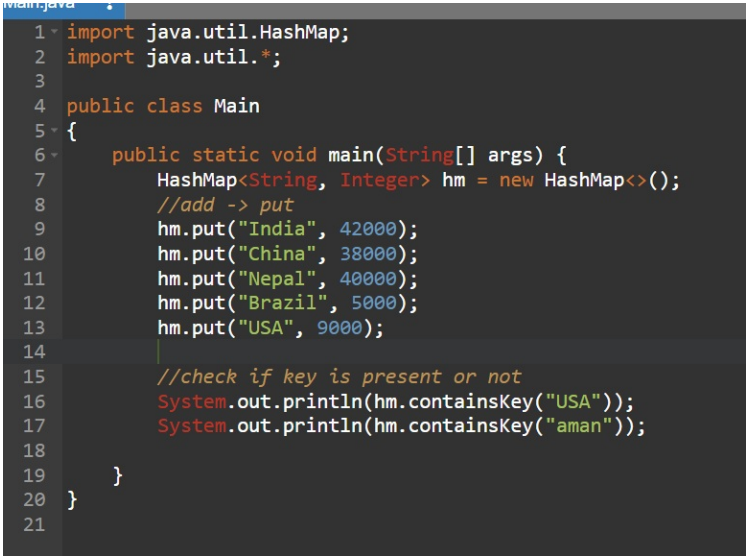

Initialize
add
remove
get
size
print
India
42000
China
38000
Nepal
40000

key of hm will be unique.



```
{
    public static void main(String[] args) {
        HashMap<String, Integer> hm = new HashMap<>();
        //add -> put
        hm.put("India", 42000);
        hm.put("China", 38000);
        hm.put("Nepal", 40000);
        hm.put("Brazil", 5000);
        hm.put("USA", 9000);

        hm.put("India", 7000);

        //remove(key)
        hm.remove("China");

        //get(key)
        System.out.println(hm.get("Nepal"));


        System.out.println(hm.size());
        System.out.println(hm);
    }
}
```

```
40000
```



```
import java.util.HashMap;
import java.util.*;

public class Main
{
    public static void main(String[] args) {
        HashMap<String, Integer> hm = new HashMap<>();
        //add -> put
        hm.put("India", 42000);
        hm.put("China", 38000);
        hm.put("Nepal", 40000);
        hm.put("Brazil", 5000);
        hm.put("USA", 9000);

        //check if key is present or not
        System.out.println(hm.containsKey("USA"));
        System.out.println(hm.containsKey("aman"));

    }
}
```

```java
import java.util.HashMap;
import java.util.*;

public class Main
{
    public static void main(String[] args) {
        HashMap<String, Integer> hm = new HashMap<>();
        //add -> put
        hm.put("India", 42000);
        hm.put("China", 38000);
        hm.put("China", 37000);
        hm.put("Nepal", 40000);
        hm.put("Brazil", 5000);
        hm.put("USA", 9000);

        //print all keys
        System.out.println(hm.keySet());

        for( String k : hm.keySet()){
            System.out.println(k + " -- "+ hm.get(k));
        }

    }
}
```

input

```
[USA, China, Brazil, Nepal, India]
USA -- 9000
China -- 37000
Brazil -- 5000
Nepal -- 40000
India -- 42000
```