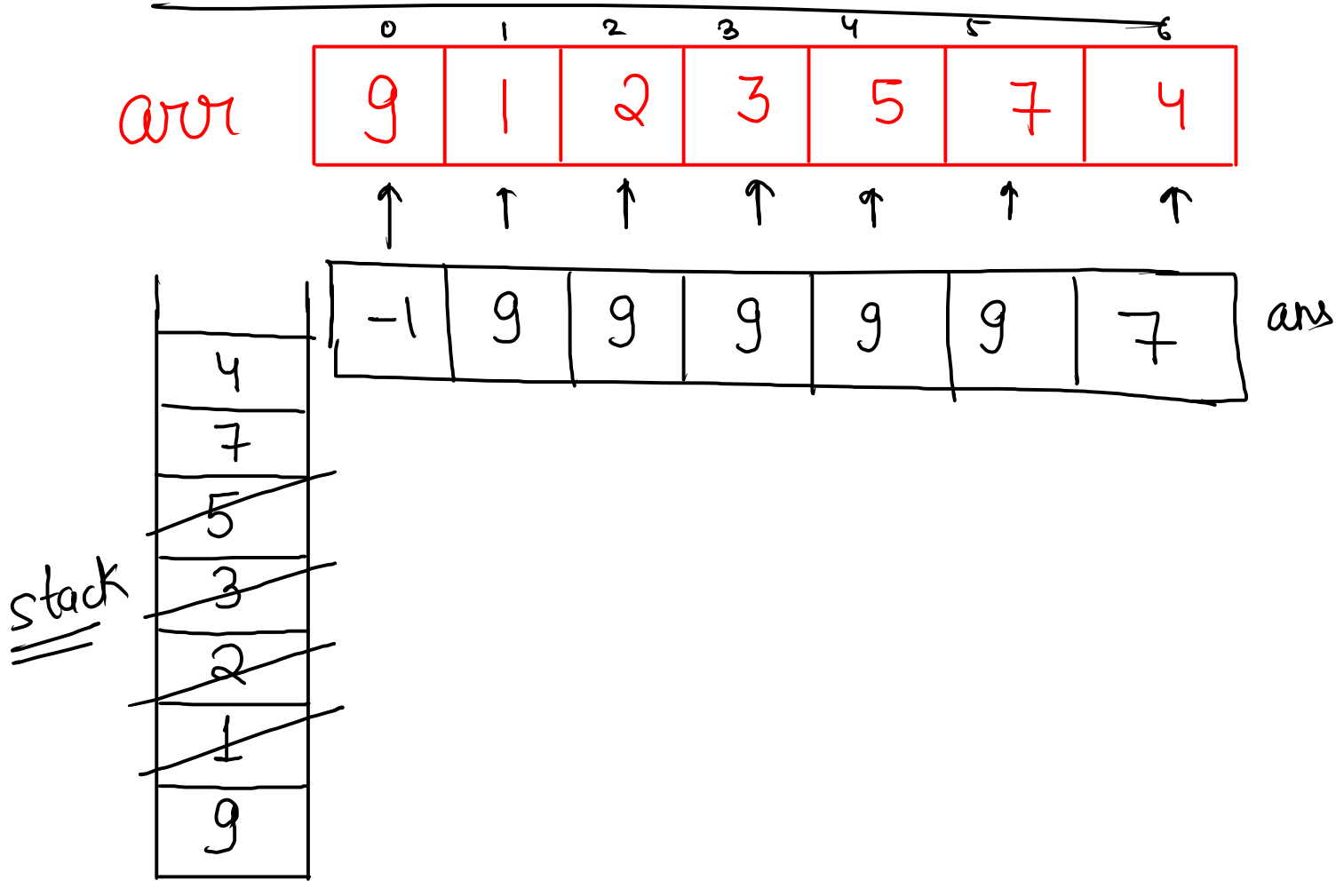
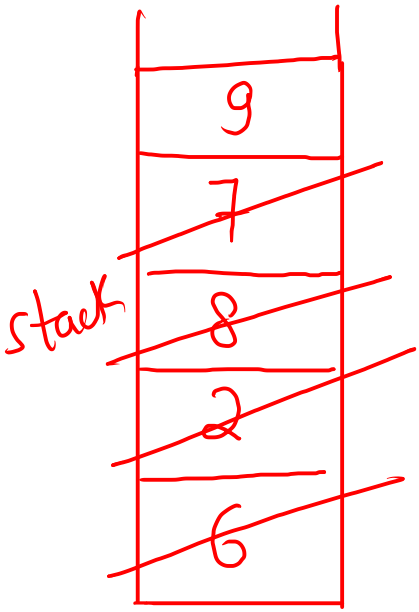
⇒ Next greater element on left



arr	6	2	8	7	9

ans	-1	6	-1	8	-1



1) traverse from left to right

1.1) while (curr > st.top)
st.pop()

1.2) if(st.isEmpty()) ans[i]=-1
ans[i] = st.top()

1.3.) st.push(curr);

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    nextGreaterOnLeft(arr, n);
}
public static void nextGreaterOnLeft(int[] arr, int n) {
    Stack<Integer> st = new Stack<>();
    int[] ans = new int[n];
    for (int i = 0; i < n; i++) {
        while ( !st.isEmpty() && st.peek() <= arr[i] ) {
            st.pop();
        }
        if ( !st.isEmpty() ) {
            ans[i] = st.peek();
        } else {
            ans[i] = -1;
        }
        st.push( arr[i] );
    }
    for (int i : ans) {
        System.out.print(i + " ");
    }
}
```

Next Smaller Element To The Right

Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    nextGreaterOnLeft(arr, n);
}
public static void nextGreaterOnLeft(int[] arr, int n) {
    Stack<Integer> st = new Stack<>();
    int[] ans = new int[n];
    for (int i = n - 1; i >= 0; i--) {
        while ( !st.isEmpty() && st.peek() >= arr[i] ) {
            st.pop();
        }
        if ( !st.isEmpty() ) {
            ans[i] = st.peek();
        } else {
            ans[i] = -1;
        }
        st.push( arr[i] );
    }
    for (int i : ans) {
        System.out.print(i + " ");
    }
}
```

inp

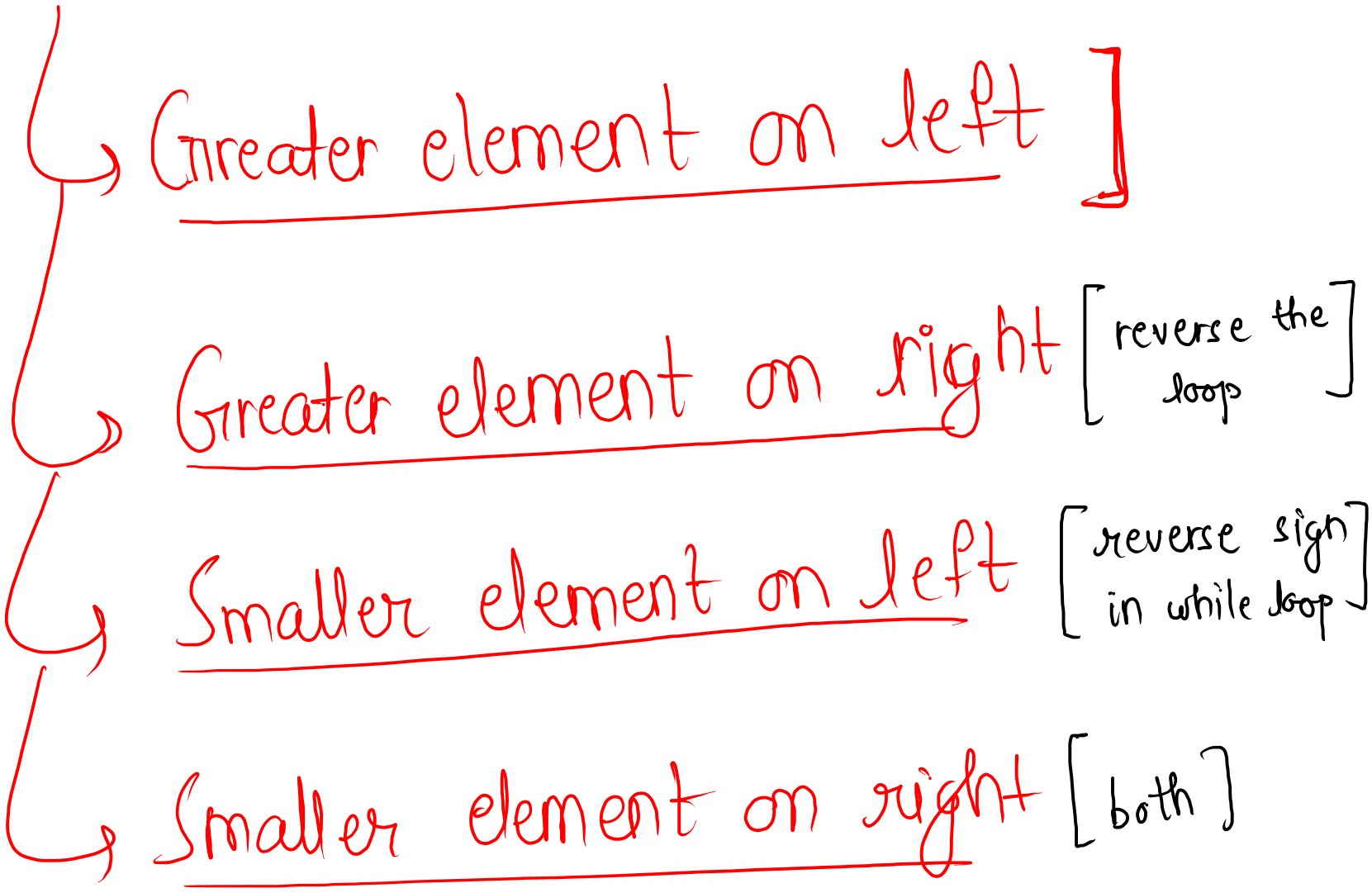
T.C

$2 \times N$

$\cong O(N)$

S.C

$\cong O(N)$


Greater element on left
Greater element on right
[reverse the loop]
Smaller element on left
[reverse sign in while loop]
Smaller element on right
[both]

⇒ Hashmap [all operations in HM are constant]

↳ key, value

↳ String, Integer, Boolean, ...

↳ Arraylist, Stack, . . .

Cricket match

key → value
"Bharat" → 240
"SriLanka" → 100
"Australia" → 241
"Pak" → 0

Note:- key will always be unique

Inbuilt function

HashMap<DataType of key, DataType of Value> map = new HashMap<>();

```
map.put("Kunal", 50);
map.put("Ajay", 75);
map.put("Ajay", 80);
map.remove("Kunal");
map.get("Ajay"); ⟶ 80
```

String, Integer

Ajay → 80

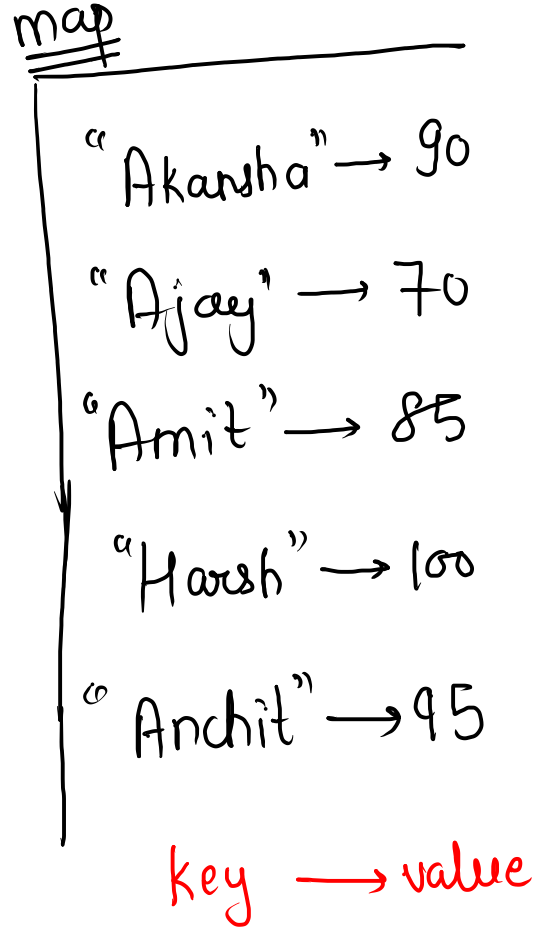
↳ map.containsKey("karan")

//False

↳ map.containsValue(70);

//True

↳ map.size(); //5



Word Meaning

Code

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    HashMap<String, String> map = new HashMap<>();
    while (true) {
        int n = scn.nextInt();
        if ( n == 1 ) {

            String word = scn.next();
            String meaning = scn.next();
            addWord(map, word, meaning);

        } else if ( n == 2 ) {

            String word = scn.next();
            printMeaning(map, word);

        } else if ( n == 3 ) {

            String word = scn.next();
            removeWord(map, word);

        } else if ( n == 4 ) {
            break;
        }
    }
}
public static void addWord(HashMap<String, String> map, String word, String meaning) {
    map.put(word, meaning);
}
public static void printMeaning(HashMap<String, String> map, String word) {
    if ( map.containsKey(word) )
        System.out.println(map.get(word));
    else
        System.out.println("-1");
}
public static void removeWord(HashMap<String, String> map, String word) {
    map.remove(word);
}
```

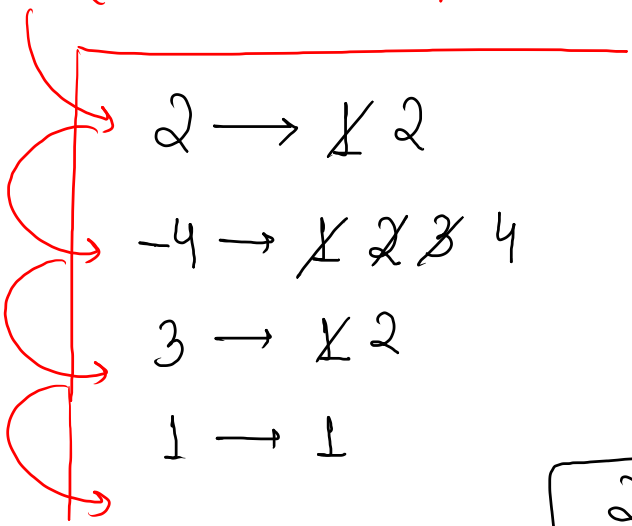
Same Number Same Frequency

n = 9

2 -4 3 -4 1 3 -4 -4 2

map

Integer, Integer
(number) (frequency)



1) traverse hashmap

```
if( |key| == value)
    print (key)
```

|2| == 2

|-4| == 4

|3| == 2

|1| == 1

ans

2
-4
1

for each loop in hashmap

Imp 1)

```
for (Map.Entry<Integer, Integer> e : map.entrySet()) {

    print ( e.getKey() + "  " + e.getValue());

}
```