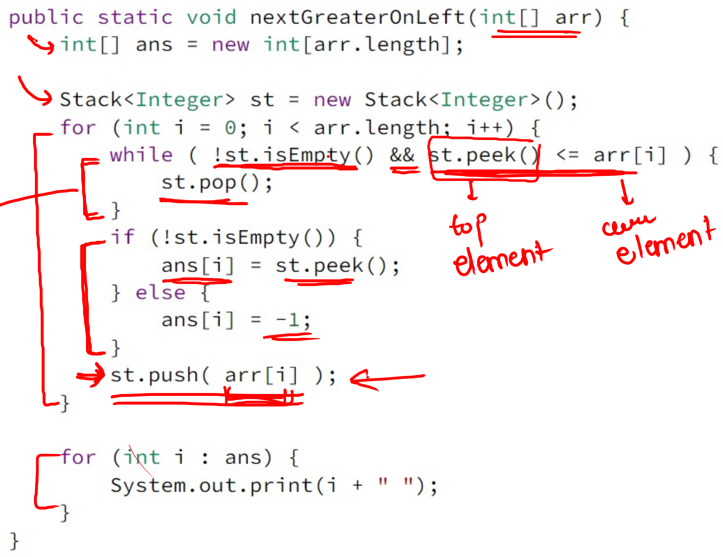
code

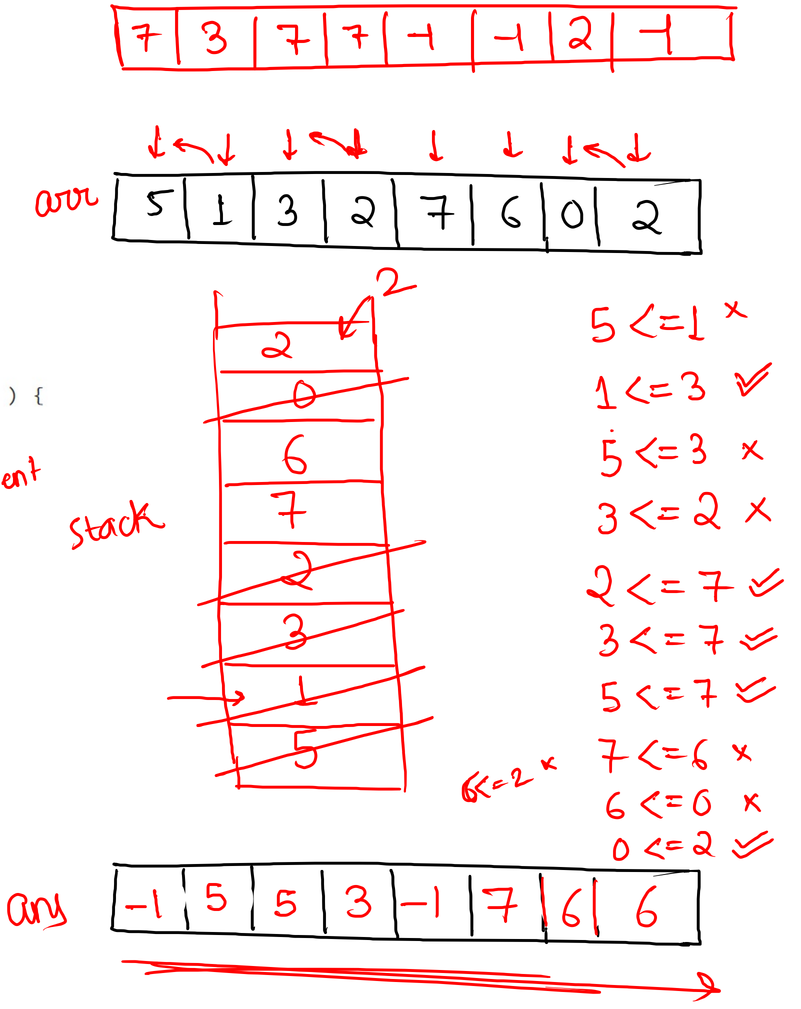


```
public static void nextGreaterOnLeft(int[] arr) {
    int[] ans = new int[arr.length];

    Stack<Integer> st = new Stack<Integer>();
    for (int i = 0; i < arr.length; 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 + " ");
    }
}
```

ensuring all unessary elements will be pop out



↳ Greater element on left (done)

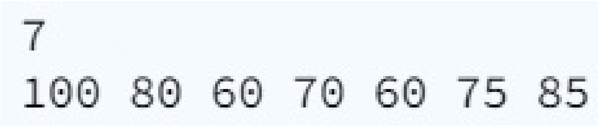
↳ Greater element on right (reverse the for loop)

↳ Smaller element on left (condition is while loop will get reverse)

↳ Smaller element on right

↳ reverse the loop & and the cond^n is while as well)

Online Stock Spanner

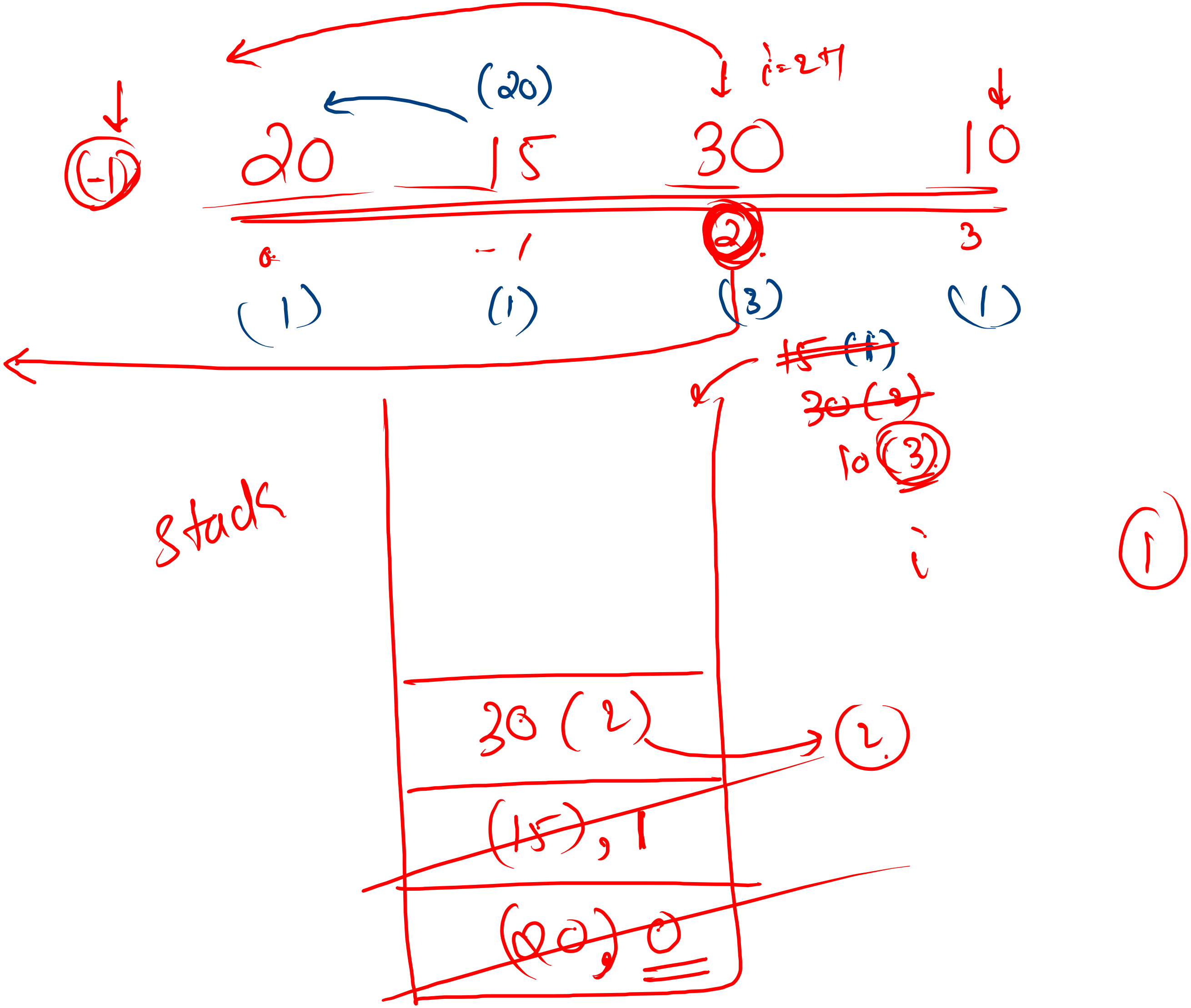


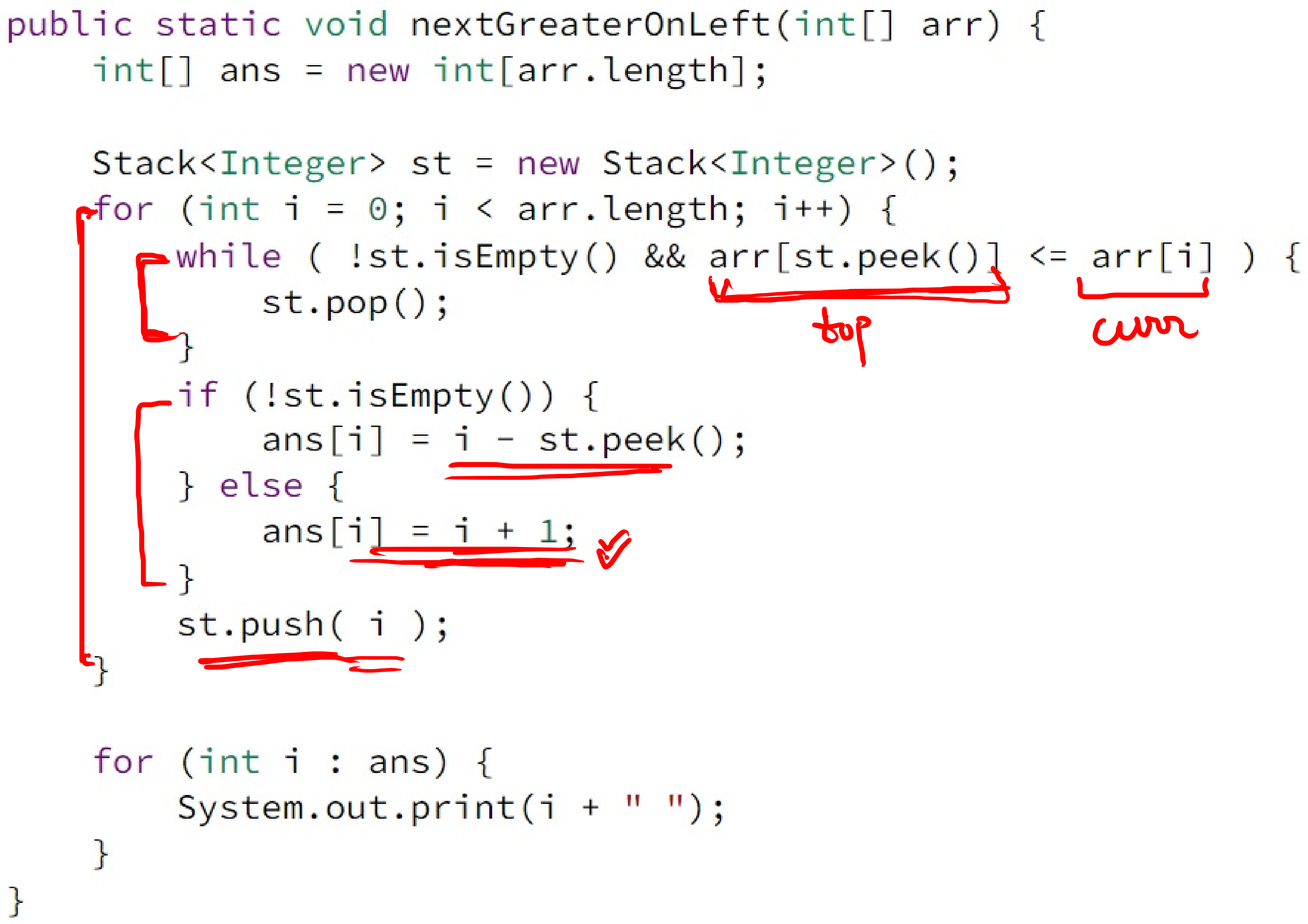
```
7
100 80 60 70 60 75 85
```

0	1	2	3	4	5	6
1	1	1	2	1	4	6
(-1)	(100)	(80)	(80)	(70)	(80)	(100)

Smart work

↳ Instead of storing the element, I will store the index of each element.

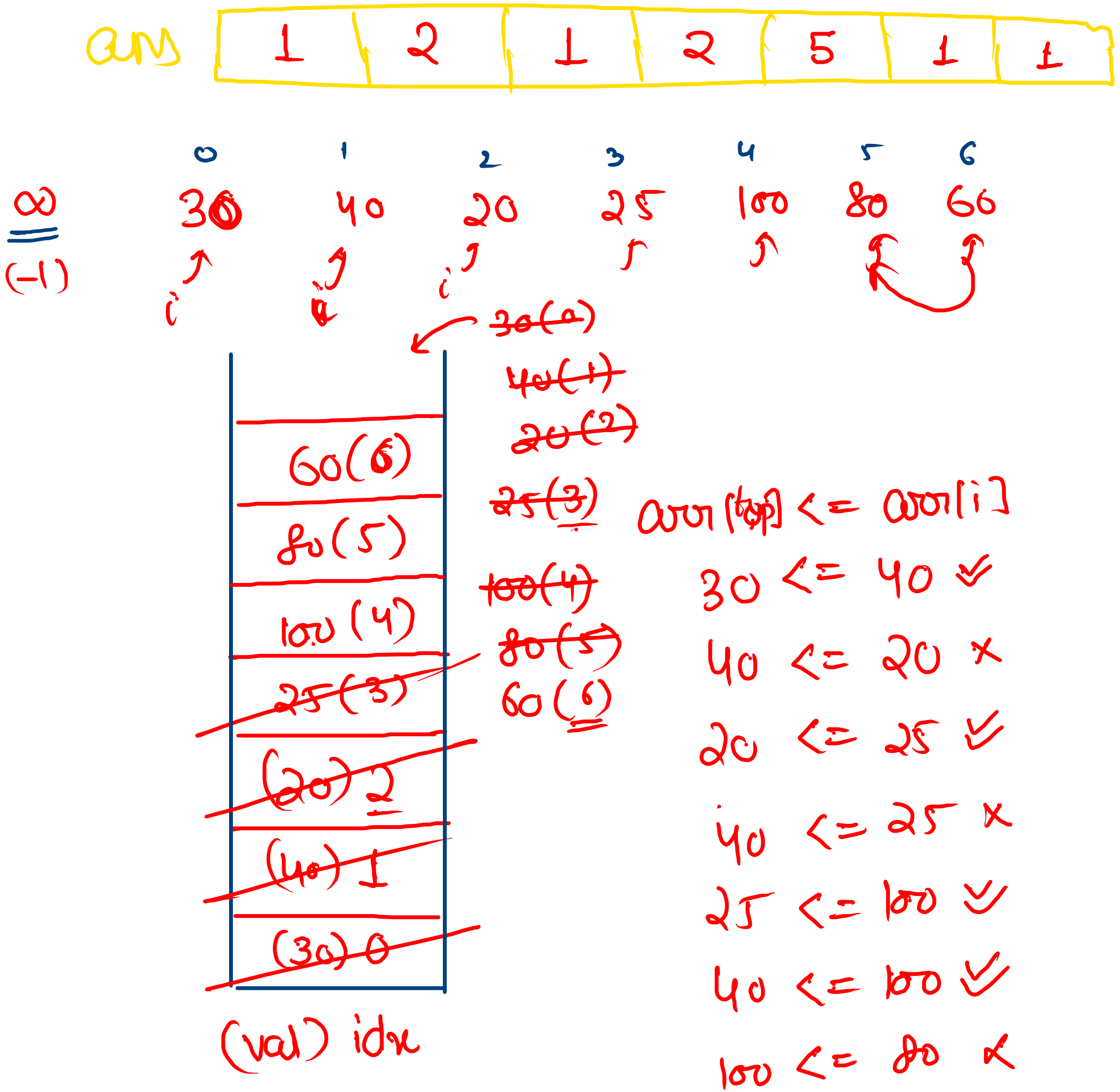

(20)
i=2+1
(-1)
20
15
30
10
0
-1
2
3
(1)
(1)
(3)
(1)
~~15 (1)~~
~~30 (2)~~
10 (3)
stack
i
1
30 (2)
2
(15), 1
(20), 0



```java
public static void nextGreaterOnLeft(int[] arr) {
    int[] ans = new int[arr.length];

    Stack<Integer> st = new Stack<Integer>();
    for (int i = 0; i < arr.length; i++) {
        while ( !st.isEmpty() && arr[st.peek()] <= arr[i] ) {
            st.pop();
        }
        if (!st.isEmpty()) {
            ans[i] = i - st.peek();
        } else {
            ans[i] = i + 1;
        }
        st.push( i );
    }

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



arr[top] <= arr[i]

30 <= 40 ✓

40 <= 20 ✗

20 <= 25 ✓

40 <= 25 ✗

25 <= 100 ✓

40 <= 100 ✓

100 <= 80 ✗

80 <= 60 ✗

⇒ Hashmap (very useful)

↳ it store a pair of key and value

possibility:-

key	value
↳ Integer,	Integer
↳ Integer,	String
↳ String,	Integer
↳ Integer,	AL
↳ Integer,	array
↳ ⋮	

etc.

ex: String, Integer

{
"Aus", 200
"Z", 150
"India", 500
~~"Pak", 0~~
"Srilanka", 50
}

"Pak", -10

Important properties of HM

- Store key/value pair
- It doesn't maintain an order (unorganised)
- In case of duplicate key, then new pair will overwrite previous value

Syntex

```
HashMap<keyDataType, valueDataType> map = new HashMap<>();
```

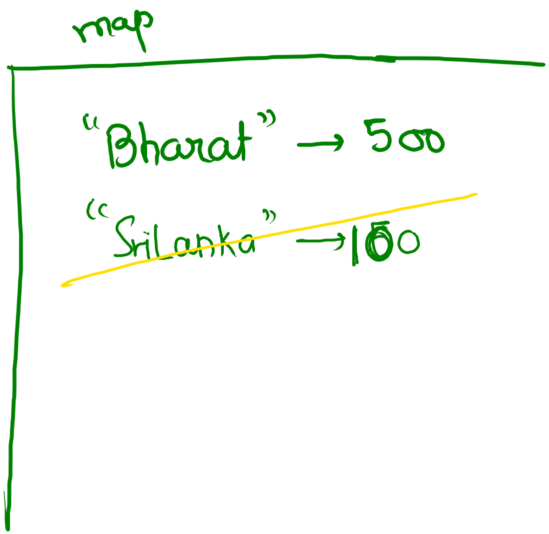
→
```
HashMap<String, Integer> map = new HashMap<>();
```

Inbuilt functions

key value

```
map.put("Bharat", 500);
map.put("SriLanka", 50);
map.put("SriLanka", 100);
map.remove("SriLanka")
```

key



decimal (10)		binary (2)
0	→	0
1	→	1
2	→	10
3	→	11
4	→	100
5	→	101
6	→	110
7	→	111
8	→	1000
9	→	1001

space