בס"ד

Max - Max / Min - Min

<u>הבעיה</u>: למצוא את 2 האיברים המקסימלי / המינימלי במערך בסיבוכיות O(n + log n) ?

A1	A2	A3	A4	A5	A6	A7	A8

Arr[] =

? Max / Min

? Max / Min

<u>פתרון</u>:

(הפתרון הוא לפי max-max אבל גם אותו פרק אצל min-min).

1. תחילה נעביר את המערך לוקטור של node (בוקטור ניתן למחוק איברים).
2. נעבור על הוקטור ונשווה 2 איברים כל פעם, נמחק את האיבר המינימלי ונכניס אותו למחסנית באיבר המקסימלי יותר.
נעדכן כשנגיע לאיבר אחד בוקטור (שגם האיבר המקסימלי) .
3. נשארנו עם וקטור באורך 1, ניקח ממנו את הערך (num) ל- max1 .
ונחפש בתוך המחסנית את הערך המקסימלי ביותר (בפרק הרלוונטי למצוא מקסימלי) ונכניס ל- max2 .

ב.ס.ד

קוד:

```
import java.util.Stack; // מחלקת המחסנית

import java.util.Vector; // מחלקת הוקטור


public class Max_Max {

        public static void max_max ( int[] a ){

        // הפונקציה מפיסה 2 איברים מקסימלים ביותר במערך //

                Vector<Node> v= new Vector<Node>();

                for( int i =0 ; i< a.length ; i++){ // מעבירים את המערך לוקטור

                        Node n = new Node ( a[i] );

                        v.add( n );

                }

                int i= 0;

                while( v.size() > 1 ){

                        if ( v.elementAt(i).num > v.elementAt( i+1 ).num) {

                                v.elementAt(i).st.push(v.elementAt( i+1 ).num); // הכנסה

                                v.remove( i+1 ) ; // מחיקה

                        }

                        else {

                                v.elementAt( i+1 ).st.push(v.elementAt(i).num); // הכנסה

                                v.remove(i) ; // מחיקה

                        }

                        i++;

                        if ( ( v.size() == i ) || ( ( v.size() - 1 ) == i ) )

                                i=0; // איפוס אינדקס
```

```
        }


        int max1 = v.elementAt(0).num;
        int max2 = v.elementAt(0).st.pop();
        int temp;
        while( v.elementAt(0).st.size() > 0 ){
            temp = v.elementAt(0).st.pop();
            if( temp > max2 )
                max2 = temp;
        }
        System.out.println("O( n+log(n) )- max_1: "+max1+" , max_2: "+max2);
    }



    public static void main(String[] args) {
        int[] a = new int[100];
        for(int i=0; i < a.length ; i++){
            a[i] = ( (int) (Math.random()*100) ) + 10 ;
            System.out.println(" a["+i+"]: "+a[i]);
        }
        max_max(a);
    }
}
```

כמ"ש

```
class Node {
        int num; // ערך ראשי
        Stack<Integer> st; // מחסנית
        public Node ( int value ){    -- בנאי
                num = value;
                st = new Stack<Integer>();
        }
}
/*
Print:

 a[0]: 15
 a[1]: 62
 a[2]: 22
 a[3]: 76
 a[4]: 45
 a[5]: 57
 a[6]: 72
 a[7]: 98
 a[8]: 93
 a[9]: 21

O( n+log(n) )- max_1: 98 , max_2: 93

*/
```

<u>סיבוכיות:</u>

O(n) - עוברים על כל איברי המערך .

+

O(log n) - חיפוש max שני בתוך המחסנית

=

$\boxed{O(n + \log n)}$