Código Asignatura:
ISC-314

Nombre:
Ronald Mariotti

Matricula:
2014-0698

Trabajo:
Hallar la mediana en línea

```
package logical;

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

class Maxheap {

	public int heap[] = new int[10000];
	public int n = 0, hijo, padre, hijo1, hijo2;

	public void insert(int val){
		heap[n] = val;
		padre = (n+1)/2 - 1;
		if(padre < 0) {n++; return;}
		hijo = n;


		while(true){


			if(heap[hijo] > heap[padre]){


				int temp = heap[hijo];
				heap[hijo] = heap[padre];
				heap[padre] = temp;


				if(padre == 0) break;


				hijo = padre;
				padre = (padre + 1)/2 - 1;
			}
			else break;
		}
		n++;
	}


	public int del(){


		int ele = heap[0];
		heap[0] = heap[n-1];
```

```
                padre = 0;

                while(true){
                        hijo1 = padre*2 + 1;
                        hijo2 = padre*2 + 2;


                        if(hijo1 > n-1 || hijo2 > n-1) break;

                        if(heap[padre] < heap[hijo1] || heap[padre] <
heap[hijo2]){
                                if(heap[hijo1] > heap[hijo2]){
                                        int temp = heap[hijo1];
                                        heap[hijo1] = heap[padre];
                                        heap[padre] = temp;
                                        padre = hijo1;
                                }
                                else{
                                        int temp = heap[hijo2];
                                        heap[hijo2] = heap[padre];
                                        heap[padre] = temp;
                                        padre = hijo2;
                                }
                        }
                        else break;
                }
                n--;
                return ele;     }


        public int getMax(){
                return heap[0];
        }
}

class Minheap {

        public int heap[] = new int[10000];
        public int n, hijo, padre, hijo1, hijo2;

        public void insert(int val){
                heap[n] = val;
                padre = (n+1)/2 - 1;
                if(padre < 0) {n++; return;}
                hijo = n;
```

```
                while(true){
                        if(heap[hijo] < heap[padre]){
                                int temp = heap[hijo];
                                heap[hijo] = heap[padre];
                                heap[padre] = temp;
                                if(padre == 0) break;
                                hijo = padre;
                                padre = (padre + 1)/2 - 1;
                        }
                        else break;
                }
                n++;
        }

        public int del(){


                int ele = heap[0];
                heap[0] = heap[n-1];

                padre = 0;


                while(true){
                        hijo1 = padre*2 + 1;
                        hijo2 = padre*2 + 2;


                        if(hijo1 > n-1 || hijo2 > n-1) break;


                        if(heap[padre] > heap[hijo1] || heap[padre] >
heap[hijo2]){
                                if(heap[hijo1] < heap[hijo2]){
                                        int temp = heap[hijo1];
                                        heap[hijo1] = heap[padre];
                                        heap[padre] = temp;
                                        padre = hijo1;
                                }
                                else{
                                        int temp = heap[hijo2];
                                        heap[hijo2] = heap[padre];
                                        heap[padre] = temp;
                                        padre = hijo2;
                                }
```

```
                        }
                        else break;
                    }
                    n--;
                    return ele;       }


            public int getMin(){
                    return heap[0];
            }
        }

        class MedianNew {


            public static void main(String[] args) throws FileNotFoundException,
IOException {
                    Maxheap maxh = new Maxheap();
                    Minheap minh = new Minheap();
                    int sum = 0, mediana = 0;

                    BufferedReader br = new BufferedReader(new
FileReader("Media//DataMedian.txt"));
                    String linea;
                    int count = 0;
                    while((linea = br.readLine()) != null){
                            int next = Integer.parseInt(linea);


                            if(count == 0){
                                    maxh.insert(next);
                                    sum += maxh.getMax();
                                    count++;
                                    continue;
                            }


                            int maxno = maxh.n;
                            int medianano = (count % 2 ==
0)?(count/2):((count+1)/2);
                            if(medianano == maxno){
                                    mediana = maxh.getMax();
                            }
                            else if(medianano > maxno){
                                    mediana = minh.getMin();
                            }
```

```
                    if(next < mediana){
                        maxh.insert(next);
                        count++;
                    }
                    else if(next >= mediana){
                        minh.insert(next);
                        count++;
                    }


                    if(maxh.n - minh.n > 1){
                        int temp = maxh.del();
                        minh.insert(temp);
                    }
                    else if(minh.n - maxh.n > 1){
                        int temp = minh.del();
                        maxh.insert(temp);
                    }



                    maxno = maxh.n;
                    medianano = (count % 2 == 0)?(count/2):((count+1)/2);
                    if(medianano == maxno){
                        mediana = maxh.getMax();
                    }
                    else if(medianano > maxno){
                        mediana = minh.getMin();
                    }


                    sum += mediana;

                }


                System.out.println(sum%10000);


        }
}
```