Ricardo Pineda
20160164

Laboratorio Alogoritmia 2

Ejercicio 1

Ordenar la lista de mayor a menor.

antes de ordenar:
[3, 39, 61, 91, 57, 22, 75, 89, 9, 90, 63, 78, 28, 73, 20]
luego de ordenar
[91, 90, 89, 78, 75, 73, 63, 61, 57, 39, 28, 22, 20, 9, 3]
El contador nos indica que el algoritmo fue iterado 45 veces.

Para resolver este problema, se utilizo el algoritmo de *Merge Sort* debido a que incluso en su Worst-Case, tarda O($nlog(n)$) en ordenar la lista.

Ejercicio 2

Es un codigo que recorre la lista dada, luego ordena la lista de tal manera que crea un heap, en este caso un Max-Heap. Si logra terminar de ordenar la lista retorna *True* y podemos saber que la Lista si es un heap. Caso contrario, retornara *False* y sabremos que la Lista no es un Heap.

Running Time del algoritmo es de O(n)

Ejercicio 3

.

Algorithm 1 MAX-HEAPIFY sin recursion

```
1: While( i <= heapsize)
2: izquiero <- LEFT(i)
3: derecho <- RIGHT(i)
4: if izquiero <= heapsize and A[izquiero] > A[i] then
5:     largest <- izquiero
6: else
7:     largest <- i
8: if derecho <= heapsize and A[derecho] > A[largest] then
9:     largest <- derecho
10: if largest != i then
11:     exchange A[i] <-> A[largest]
12:     i <- largest
```