Parte 1: H(eap)piness		
2	**Heap Medians**	Siendo **N pingüinos dentro de un sector, y S sectores en total dentro del tour**, la solución debe utilizar de forma mínima 2 heaps para cada sector que permitan determinar la mediana. De forma específica, un maxheap con los pinguinos de felicidades menores o iguales a la mediana actual y un minheap con los pinguinos con felicidades mayores o iguales a la mediana actual. Si no explica como genera el balance de los heaps para encontrar la mediana ni la diferenciación de caso impar y par se entrega puntaje parcial.
1	**Tiempo constante**	La solución indica metodología que permite acceder de forma constante $\mathcal{O}(1)$ a los pingüinos de mayor y menor felicidad dentro de un sector. De forma análoga realiza lo mismo con los sectores de menor y mayor felicidad media. Un ejemplo de esto puede ser utilizar max y min heaps.
1	**Referencia heaps**	Indica que utiliza referencias de las posiciones de los elementos dentro de los heaps, para permitir que las operaciones de : Actualización, y Eliminación no requieran una búsqueda exhaustiva para identificar el elemento. De esta forma se logra obtener una complejidad de $\mathcal{O}(\log(X))$ en vez de $\mathcal{O}(X\log(X))$, siendo X pingüinos (N) o sectores (S) dependiendo del heap en cuestión.
1	**Complejidad**	Explica como lo logrado en los items anteriores le permiten cumplir con las complejidades solicitadas en cada uno de los eventos solicitados. Entregar puntaje parcial en caso de no explicitar algún evento.

Parte 2: O(n)rdenamiento		
1	**Orden lineal**	Se debe implementar correctamente un algoritmo de ordenación lineal, como **Counting Sort, Bucket Sort o Radix Sort**, destacando su aplicabilidad cuando el rango de los valores está acotado.
2	**Sorts**	Para ordenar por un atributo ascendente o descendente, se puede ejecutar el algoritmo seleccionado una vez, invirtiendo el orden de los índices en el caso descendente. Para el problema **DOUBLE-SORT-BY**, se debe aplicar el algoritmo dos veces, una por cada atributo, respetando el orden requerido. 2 Es importante destacar el uso de un algoritmo de ordenación lineal debido a la presencia de un rango bien definido, como las edades (hasta 99 años) o valores booleanos (0 y 1).
1	**Rangos**	Implementar el algoritmo de ordenación lineal seleccionado especificando claramente los rangos asociados a las edades (hasta 99 años) y los valores booleanos (0 y 1).
1	**Complejidad**	Se debe mencionar explícitamente que la complejidad de la ordenación es $\mathcal{O}(n+k)$ o $\mathcal{O}(nk)$ dependiendo del Sort, donde k es el rango de los valores (edad o atributo booleano), y n es el número de elementos a ordenar.