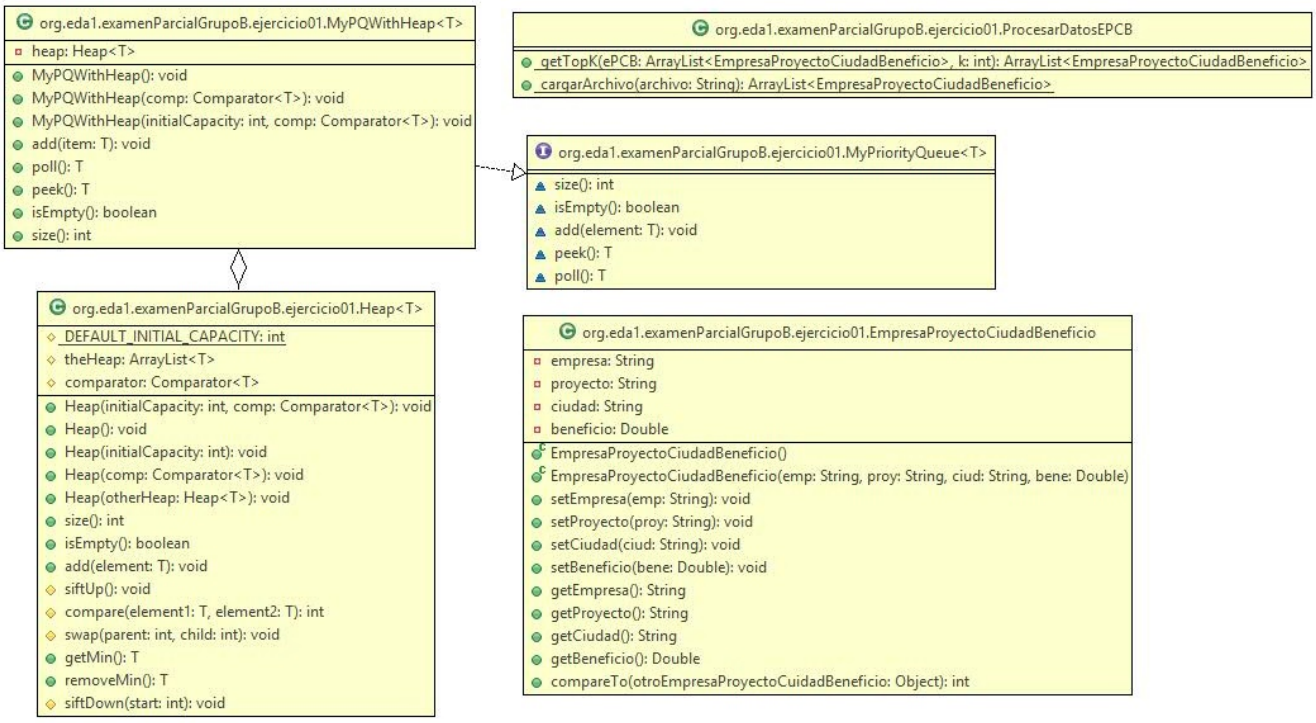
ESTRUCTURAS DE DATOS Y ALGORITMOS I
Grado en Ingeniería Informática (Segundo Curso, Primer Cuatrimestre)
Primer Examen Parcial Grupo B. 18-11-2013

Ejercicio Heap. En teoría hemos estudiado los *heap* (montículos) máximos y mínimos, y en prácticas hemos hecho uso de dicha implementación en la clase `class Heap<T>`. En dicha implementación hemos utilizado un `ArrayList<T> theHeap`, que tiene la raíz en la posición 0. Para obtener un heap mínimo de enteros (`Integer`) hemos declarado un *Comparator* de la siguiente forma `Less<Integer> less = new Less<Integer>()`, y dicho *heap mínimo* lo hemos declarado de la siguiente manera `Heap<Integer> heap = new Heap<Integer>(less)`. En este ejercicio se pide implementar en Java una cola de prioridad utilizando el *Heap* estudiado en clase y que implemente la interface `MyPriorityQueue.java` que se proporciona como material.

Para probar su correcto funcionamiento, se pide implementar el *topK* de un conjunto de objetos. Estos objetos se proporciona en el archivo *EPCB.txt* y en cada entrada tenemos datos de la siguiente forma `Empresa Proyecto Ciudad Beneficio`, y se trata de encontrar las *5* empresas que con sus proyectos, en sus respectivas sedes, obtienen los 5 mayores beneficios.



Ejercicio ABB. Dada la estructura de datos `BSTree<T>` estudiada en clase e implementada en prácticas, que se corresponde con un ABB (árbol binario de búsqueda) que no contiene duplicados. Se pide, implementar en Java una función que elimine todos los nodos hoja del ABB. **`public void removeLeaves()`**. Para este caso será necesario implementar tanto la función pública como la privada.

```
private BSTNode<T> removeLeaves(BSTNode<T> curr)

public void removeLeaves()
```

Para cada ejercicio se proporciona el correspondiente test que se deberá de pasar correctamente.