ESTRUCTURA DE DADES: PRÁCTICA 4

Sergio Barril Pizarro — 16653350
Rodrigo Cabeza Quirós — 20100426
Grupo: D00
Adán Beltran Gómez

MinHeap:

- Tiempo generación estructura:

Fichero	Tiempo	Estructura
movie_rating_small.txt	0.0005182	MinHeap
movie_rating.txt	0.01589	MinHeap

El tiempo de generación de la estructura para el fichero grande, es algo mayor respecto al fichero pequeño. Aun así, es razonable, pues el fichero grande equivale 90 veces el pequeño.

- Tiempo acceso estructura (cercaFitxers):

Fichero	Tiempo	Estructura
movie_rating_small.txt	0.0341641	MinHeap
movie_rating.txt	2.70034	MinHeap

Para el fichero grande, el tiempo de acceso es bastante mayor, aunque razonable. Teniendo en cuenta que el fichero pequeño tiene 100 entradas mientras que el grande tiene 9066 y que el coste computacional teórico de cada búsqueda es O(n), podemos afirmar que los tiempos obtenidos son razonables e inmejorables. Concluimos así, que el Heap no es una buena estructura para hacer búsquedas de elementos.

- Coste computacional teórico:

MinHeap

Función	Coste	Justificación
Constructor	O(1)	Inicializa atributos.
Constructor copia	O(n)	Inicializa los atributos.
Destructor	O(1)	
size	O(1)	Retorna el tamaño del heap.
empty	O(1)	Comprueba si el heap está vacío.
insert	O(log n)	Inserta una nueva Entry en el heap.
min	O(1)	Retorna la mínima clave.
minValues	O(1) / O(n)	Retorna valores de la clave mínima. O(1) si las claves son únicas, O(n) en el peor caso, si se pueden repetir.
removeMin	O(log n)	Elimina Entry con la mínima clave.
printHeap	O(n)	Muestra el heap por pantalla.
search	O(n)	Retorna un vector con los valores correspondientes a una clave existente en el heap.
search recursivo	O(n)	Recorre el heap llenando el vector con los valores que tengan la key que se busca.
left	O(1)	Retorna la posición en el heap del hijo izquierdo.
right	O(1)	Retorna la posición en el heap del hijo derecho.
parent	O(1)	Retorna la posición en el heap del padre.
hasLeft	O(1)	Comprueba si existe hijo izquierdo.

hasRight	O(1)	Comprueba si existe hijo derecho.
hasParent	O(1)	Comprueba si tiene padre.
upHeap	O(log n)	Operación de upHeap.
downHeap	O(log n)	Operación de downHeap.
swap	O(1)	Intercambia los valores de dos claves.

- Implementación del TAD MinHeap:

  Para representar los nodos del heap hemos utilizado una clase Entry, que contiene una clave int y el valor a guardar, hecho con templates.

  Hemos reutilizado el código principalmente de las funciones del main y la función appendMovies de HeapMovieFinder.

  La implementación con listas enlazadas no habría sido más eficiente. Usando vectores, la implementación es igual de eficiente en términos de complejidad teórica de tiempo, y es más eficiente en espacio.