Técnicas algorítmicas en ingeniería del software

Grado en Ingeniería del Software (UCM)

PRÁCTICA DE MONTÍCULOS — Curso 2014/2015

FECHA LÍMITE DE ENTREGA: 7 DE NOVIEMBRE

Se valorará tanto la *corrección* como la *eficiencia* y la *simplicidad* de las soluciones propuestas.

Ejercicio 1 La *mediana* de un conjunto de N elementos ordenables es el elemento que ocuparía la posición $\lfloor (N+1)/2 \rfloor$ si los elementos se ordenaran. Diseña e implementa una estructura de datos para almacenar números enteros, con las siguientes operaciones: *insertar* un elemento con coste logarítmico, *consultar la mediana* con coste constante y *eliminar la mediana* con coste logarítmico.

Pista: Puede convenir utilizar un montículo de máximos y otro de mínimos.

Ejercicio 2 Supongamos que tenemos cierto número de ficheros cada uno conteniendo una lista ordenada de números enteros. Queremos producir (por pantalla o en otro fichero) la mezcla ordenada del contenido de todos los ficheros, con las siguientes restricciones: los ficheros de entrada pueden leerse solamente una vez y no puede almacenarse en memoria su contenido completo. Implementa una función

```
void mezcla_multiple(ifstream streams[], int N)  // ocupadas las posiciones [1..N]
```

que recibe un array de ficheros abiertos dispuestos para ser leídos y produce la mezcla ordenada de sus contenidos.

El programa principal (que recibe el nombre de los ficheros de entrada como argumentos en la línea de comandos) podría ser así:

```
int main(int argc, char* argv[]) {
    if (argc > 1) {
        ifstream *streams = new ifstream[argc];
        for(int i=1; i < argc; i++)
            streams[i].open(argv[i]);

        mezcla_multiple(streams, argc-1);

        for(int i=1; i < argc; i++)
            streams[i].close();

    }
}
```

Por ejemplo, si tenemos ficheros

```
m1.txt:  1 2 3 6 7 9 9 26
m2.txt:  2 4 8 16 17 17
m3.txt:  1 2 5 6 10 14
```

la salida sería

```
1 1 2 2 2 3 4 5 6 6 7 8 9 9 10 14 16 17 17 26
```