Parcial 2 - PRÁCTICAS - PRG - GIIA. Curso 2017-18

28 de mayo de 2018. Duración: 1 hora

Nota: El examen se evalúa sobre 10 puntos, pero su peso específico en la nota final de la asignatura es de 1,2 puntos.

NOMBRE:	GRUPO DE PRÁCTICAS:

1. [5 puntos] Implementar un método en la clase `SortedRegister` de la práctica 4 con perfil:

```
public void testNegData(Scanner s, PrintWriter err)
```

donde los parámetros son ficheros de texto creados y abiertos fuera del método. El primero (`s`) está abierto para lectura de los datos y en el segundo (`err`) se escribe el resultado de la ejecución. En el fichero `s` se espera que todas las líneas estén formadas por tres números enteros (positivos o negativos) separados por espacios en blanco.
El método debe generar un fichero de texto con las líneas de `s` cuyos tres primeros elementos sean números enteros y alguno sea negativo. Si al menos uno de los tres números es negativo, los tres se escriben en el fichero `err` precedidos del número de línea que ocupan en el fichero origen `s`. Considera que si en alguna línea aparece algún valor que no es un entero, el tratamiento de las excepciones de tipo `InputMismatchException` sólo consiste en pasar a la siguiente línea. También se supone que el cierre de los dos ficheros se realiza fuera del método.

Ejemplo:

Fichero de entrada (`s`)

```
1 12 34
5 -4 3
-30 abril 2
28 mayo 3
34 2 3
-1 3 -2
```

fichero de salida (`err`)

```
2 5 -4 3
6 -1 3 -2
```

Solución:

```
public void testNegData(Scanner s, PrintWriter err) {
         int count = 0;
         while (s.hasNext()) {
             count++;
             try {
                 int day = s.nextInt();
                 int month = s.nextInt();
                 int amount = s.nextInt();
                 if (day < 0 || month < 0 || amount < 0  ) {
                    err.println( count + " " + day + " " + month+ " " + year);
                 }
             } catch (InputMismatchException e) {
              s.nextLine();}
         }
     }
```

2. [5 puntos] Considera la clase `SetString` de la práctica 5 que define objetos que son conjuntos de `String`. La estructura de datos usada en su implementación consiste en una secuencia enlazada que contiene los elementos del conjunto ordenados ascendentemente por el orden de `String` y sin elementos repetidos. El primer nodo de esta secuencia está referenciado por el atributo `first`.

```
public class SetString {
    private NodeString first;
    private int size;
    /** Crea un conjunto vacío */
    public SetString() {
        this.first = null;
        this.size = 0;     }
    ....
}
```

Se pide implementar un nuevo método dentro de la clase, con perfil

```
public SetString subsetMaxLength (int t)
```

que devuelva un `SetString` con todos los elementos del conjunto que tengan una longitud menor o igual que `t`. Para cada elemento que cumpla la condición debe crearse un nuevo elemento en el conjunto resultante. El método ha de tener un coste lineal $O(n)$, siendo n la talla de `this`, y por lo tanto, se ha de evitar el uso del método `add()`.

Ejemplo:

Conjunto `this`	Conjunto resultante para t = 6
`"abcd"`	`"abcd"`
`"cc"`	`"cc"`
`"ejercicio"`	`"examen"`
`"examen"`	

Solución:

```
public SetString subsetMaxLength(int t) {
    SetString cs = new SetString();
    NodeString aux = first,
               lastNode = null;
    while (aux != null ) {
        if (aux.data.length() <= t) {
            NodeString nuevo = new NodeString(aux.data);
            if (cs.first == null) { cs.first = nuevo; }
            else { lastNode.next = nuevo;  }
            lastNode = nuevo;
            cs.size++;
        }
        aux = aux.next;
    }
    return cs;
}
```