**Fundamentos de Programación. Convocatoria de Septiembre. Curso 2010/2011**

## Recuperación de la parte práctica de Febrero (40 % de la nota final)

**Tiempo: una hora y media.**

**IMPORTANTE: Los algoritmos han de ir correctamente explicados.**

Se pide construir un programa que lea una matriz y que calcule la posición de aquel elemento que sea el mayor de entre los mínimos de cada fila. El programa mostrará la posición de dicho elemento (fila y columna). Por ejemplo, dada la matriz $M$ $(3 \times 4)$,

$$\begin{array}{cccc} 9 & 7 & 4 & 5 \\ 2 & 18 & 2 & 12 \\ 7 & 9 & 1 & 5 \end{array}$$

el máximo entre 4, 2 y 1 (los mínimos de cada fila) es 4 que se encuentra en la posición $(0, 2)$. Los datos de entrada al programa se deben dar en el siguiente orden:

1. Número de filas de la matriz.
2. Número de columnas de la matriz.
3. Los elementos de la matriz.

**Restricciones del problema:**

1. Debe implementar una clase "Matriz". Dicha clase debe incorporar la funcionalidad necesaria para resolver el problema.
2. No se admitirá que la entrada o la salida de datos se realice dentro de la clase que incorpora la funcionalidad para resolver este problema. Esas operaciones se resolverán fuera de la clase.

**Ejemplo de fichero de validación:**

```
3
4
9  7   4   5
2  18  2   12
7  9   1   5
```

**Salida del programa:** Dos enteros que representan la fila y columna del elemento buscado:

```
0  2
```

Departamento de Ciencias de la Computación e I.A.
Universidad de Granada
DECSAI

**Fundamentos de Programación. Convocatoria de Septiembre. Curso 2010/2011**

## Recuperación de la parte escrita de Febrero (60 % de la nota final)

**Tiempo: dos horas y media.**
**IMPORTANTE: Los algoritmos han de ir correctamente explicados.**

1. (2 puntos) Definir una función **recursiva** a la que se le pasen dos valores de tipo `int` $n$ y $k$ y devuelva como resultado la suma de todos los $k$ dígitos menos significativos de $n$. Por ejemplo para $n = 61427$ y $k = 3$ el resultado es $7 + 2 + 4 = 13$, y para $n = 23$ y $k = 4$ el resultado es $2 + 3 = 5$ .

2. Implemente la clase `Texto` para representar una colección de cadenas de caracteres (`string`). Esta clase debe implementar la siguiente funcionalidad pública:

   **Bloque 1** (2 puntos). **Tareas generales**. Escribir métodos para: 1) consultar cuántas cadenas contiene, 2) obtener la cadena $i$-ésima, 3) añadir una nueva cadena, 4) insertar una cadena en la posición $i$-ésima, y 5) eliminar la cadena $i$-ésima. Para todas estas funciones, $0 \leq i < n$ donde $n$ es el número actual de cadenas.

   **Bloque 2** (2 puntos). **Tareas de "limpieza"**. Escribir métodos para: 1) eliminar los "blancos" *iniciales* de todas las cadenas, 2) eliminar los "blancos" *finales* de todas las cadenas, y 3) eliminar los "blancos" *iniciales* y *finales* de todas las cadenas.

   **Bloque 3** (2 puntos). **Tareas de reorganización**. Puede ser interesante reordenar las cadenas que componen la colección en un momento dado. En lugar de los criterios clásicos (orden lexicográfico, longitud, ...) proponemos emplear otro criterio, basado en lo que llamaremos *índice de ocupación de una cadena*. Un índice de ocupación es un número real entre 0.0 y 1.0, que indica cuántas casillas de la cadena contienen espacios en blanco. El valor 1.0 indica que no hay ninguna casilla en blanco y el valor 0.0 que todas son blancos. El alumno debe escribir un método que modifique el objeto `Texto` de manera que reorganice las cadenas de mayor a menor índice de ocupación.

   **Notas:**

   *a*) En todos los casos se deberán comprobar y tratar las situaciones posibles de error.

   *b*) Se pueden implementar tantos métodos/funciones como considere oportuno para conseguir la solución óptima.

3. (2 puntos)

   Desarrolle un programa que lea una secuencia de números enteros en el rango de 0 a 100 terminada en un número mayor que 100 o menor que 0 y encuentre la subsecuencia de números creciente (ordenada de menor a mayor), de mayor longitud, dentro de dicha secuencia. Supondremos que dos valores iguales consecutivos forman parte de la misma subsecuencia creciente. El programa nos debe decir la posición (empezando desde 1) donde comienza la subsecuencia y su longitud.

| Entrada: | 23 23 7 45 45 45 73 73 71 4 9 101 |
|---|---|
| Salida: | POSICION = 3<br>LONGITUD = 6 |