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

METODOLOGÍA DE LA PROGRAMACIÓN
EXAMEN PRÁCTICO 2
(Junio 2014)

Normas generales

¡Importante! No se corregirá ninguna solución que no cumpla escrupulosamente las normas que aparecen a continuación

- La práctica consiste en la implementación completa en ISO C++ de la solución a un problema.
- En la evaluación se tendrá en cuenta, además de la corrección de la solución propuesta, el estilo de programación, el uso correcto de espacios y tabuladores, así como la claridad del código fuente.
- Para iniciar sesión en las aulas de prácticas, tendrá que introducir su identificador de usuario, contraseña y el código que se le indique.
- No podrá acceder a su cuenta como alumno de la ETSIIT. No tendrá acceso al exterior, salvo a `decsai.ugr.es` de donde podrá recuperar las entregas realizadas, así como el material de teoría y las soluciones publicadas de los ejercicios.
- Crear una carpeta llamada `EXAMEN2` y crear la estructura de carpetas típica (`bin`, `include`,...) sobre `EXAMEN2`.
- Deberá escribir un fichero llamado `makefile` para generar el ejecutable, y todos los ficheros necesarios para conseguirlo. Cada fichero deberá estar almacenado en la carpeta que le corresponde.
- La entrega de la práctica se hará durante el periodo de tiempo en el que se realiza el examen, y desde los ordenadores instalados en el aula. Para efectuar la entrega se usará la plataforma **decsai** y se procederá de manera similar a como realiza las entregas de prácticas semanalmente.
- La práctica se puede entregar tantas veces como se quiera durante el examen. El sistema guarda la última entrega. De hecho, **se recomienda que se entregue varias veces a lo largo del examen**, ya que si el ordenador se quedara "colgado", habría que reiniciarlo y se perdería toda la información.
- Tiempo de examen: **75 minutos**

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

METODOLOGÍA DE LA PROGRAMACIÓN
EXAMEN PRÁCTICO 2
(Junio 2014)

EJERCICIO

Copiar en las carpetas adecuadas los ficheros necesarios para la gestión de matrices de datos `int` (`Matriz2D_1.h` y `Matriz2D_1.cpp`).

Escribir los siguientes métodos de la clase `Matriz2D_1`:

- `Matriz2D_1 (const char * nombre_fichero);`

  Constructor. Recibe el nombre de un fichero y rellena la matriz con los datos guardados en el fichero. Si el fichero no existe, o no se puede abrir, construye una matriz vacía.

- `void LeeDeFichero (const char * nombre_fichero);`

  Recibe el nombre de un fichero y rellena la matriz con los datos guardados en el fichero. La matriz que recibe los datos *ya existe*. Si ésta tuviera datos, se pierden. Si el fichero no existe, o no se puede abrir, se obtiene una matriz vacía.

- `void GuardaEnFichero (const char * nombre_fichero);`

  Recibe el nombre de un fichero y copia en él los datos guardados en la matriz. Si el fichero tuviera datos, éstos se pierden.

El formato del fichero es el siguiente:

> En primer lugar aparecen dos números enteros en formato **binario**, que indican el número de columnas (c) y filas (f), respectivamente. A continuación encontrará f líneas de **texto** formadas por c números enteros separados por espacios en blanco o tabuladores (que pueden estar en cualquier orden y número).

Estos métodos se probarán en la siguiente función `main()`:

```
......
Matriz2D_1 m1 (argv[1]); // Constructor
cout << m1;  // m1.Pinta() ó PintaMatriz (m1), a su elección

m1.GuardaEnFichero (argv[2]); // Método de escritura

Matriz2D_1 m2;
m2.LeeDeFichero(argv[2]); // Método de lectura
cout << m2;  // m2.Pinta() ó PintaMatriz (m2), a su elección

if (m1==m2)
    cout << "Lectura y escritura correcta." << endl;
else
    cout << "Error en la lectura y/o escritura." << endl;
......
```

El programa se ejecutará proporcionándole dos argumentos, que serán dos nombres de fichero.

- El primero se supone que existe, y contendrá una matriz (en el formato mixto descrito anteriormente).
- El segundo lo creará el programa y contendrá una copia (en el formato mixto descrito anteriormente) de la matriz. Si existiera un fichero con ese nombre, se perdería su contenido y se sustituiría por el nuevo.

  MEJORA: Comprobad si el segundo fichero existe, dar el aviso y preguntar si realmente se quiere borrar. Actuar en consecuencia.

Tareas a realizar:

1. Actualizar adecuadamente los ficheros `Matriz2D_1.h` y `Matriz2D_1.cpp`.
2. Escriba en el fichero `examen.cpp` la función `main()` que permita probar la solución.
3. Escribir el fichero `makefile`.