Programación 2

Examen de teoría (junio 2012)

31 de mayo de 2012

Instrucciones

- **Duración: 3 horas**
- El fichero del primer ejercicio debe llamarse `ej1.cc`. Para el segundo problema es necesario entregar cuatro ficheros, llamados `Coordenada.cc, Coordenada.h, Poligono.cc, Poligono.h`. Pon tu DNI y tu nombre en un comentario al principio de los ficheros fuente.
- La entrega se realizará del mismo modo que las prácticas, a través del servidor del DLSI (`http://pracdlsi.dlsi.ua.es`), en el enlace `Programación 2`. Puedes realizar varias entregas, aunque sólo se corregirá la última.

Problemas

1. **(6 puntos)**

   Tenemos un fichero de texto con datos de jugadores de fútbol en formato `dorsal:nombre:equipo`. Un ejemplo del fichero sería:

   ```
   1:Casillas, Iker:Real Madrid
   8:Iniesta, Andres:Barcelona
   9:Falcao,Radamel:Atletico de Madrid
   10:Messi, Lionel:Barcelona
   14:Alonso, Xavi:Real Madrid
   5:Pujol,Carles:Barcelona
   ```

   Se trata de hacer un programa que reciba como parámetro el nombre de un fichero de jugadores, lo lea e imprima el nombre de cada equipo seguido de los datos de sus jugadores. Con el ejemplo anterior, ejecutando el programa como `./ej1 jugadores.txt`, se mostraría por pantalla lo siguiente:

   ```
   -----Real Madrid-----
   1 Casillas, Iker
   14 Alonso, Xavi
   -----Barcelona-----
   8 Iniesta, Andres
   10 Messi, Lionel
   5 Pujol,Carles
   -----Atletico de Madrid-----
   9 Falcao,Radamel
   ```

   Para implementar el problema, es necesario crear un array de equipos. Cada equipo tendrá un nombre y un array de jugadores[1]. Puede haber como máximo 20 equipos, y cada uno de ellos puede tener hasta 22 jugadores.

   Se supone que el formato del fichero será siempre correcto y que acaba con un salto de línea. Hay que controlar los errores de argumentos del programa y de apertura de fichero.

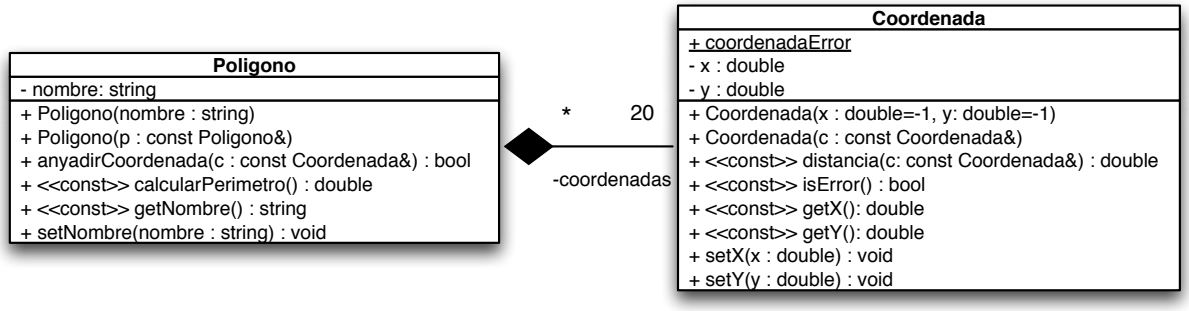
   Nota: Este problema debe implementarse usando programación procedural (no orientada a objetos).

2. **(4 puntos)**

   Queremos hacer un programa que nos permita crear polígonos y calcular su perímetro. Para ello, partimos del siguiente diagrama de clases[2]:

[1] Se pueden crear los registros con tantos campos como se considere necesario.

[2] A pesar de que la relación de composición debería implementarse con un vector de punteros, se hará mediante un vector de objetos, al igual que en la práctica 3. En los métodos, podéis elegir los nombres de los parámetros (no tienen por qué coincidir con los sugeridos en el diagrama). Se pueden añadir métodos o atributos privados, aunque no es necesario.



Un polígono viene definido por una secuencia de coordenadas[3]. La clase `Coordenada` tiene un método que nos indica la distancia con otra coordenada, que puede calcularse mediante la siguiente fórmula:

$$distancia(a,b) = \sqrt{(a_x - b_x)^2 + (a_y - b_y)^2} \quad (1)$$

Para calcular la raíz cuadrada de un número `x` puede usarse la función `sqrt(x)`, y para elevarlo al cuadrado se usa `pow(x,2.0)`. Para poder utilizar estas funciones es necesario incluir la librería `cmath`.

El método `Poligono::anyadirCoordenada` debe devolver `false` si no se puede añadir la coordenada porque el vector está completo.

El método `Poligono::calcularPerimetro` calcula la suma de las distancias entre cada coordenada y su coordenada anterior, mas la distancia entre la última coordenada y la primera, para cerrar el polígono.

Dado el siguiente fichero `main.cc`:

```
#include <iostream>
#include "Poligono.h"
using namespace std;

int main()
{
   Poligono p("mi triangulo");

   p.anyadirCoordenada(Coordenada(0,0));
   p.anyadirCoordenada(Coordenada(3,3));
   p.anyadirCoordenada(Coordenada(3,0));

   cout << "Perimetro de " << p.getNombre() << " = " << p.calcularPerimetro() << endl;
}
```

El programa debería imprimir:

```
Perimetro de mi triangulo = 10.2426
```

Ayuda: Puedes usar el siguiente `makefile` para compilar el programa:

```
main : main.o Poligono.o Coordenada.o
        g++ -o main main.o Poligono.o Coordenada.o
main.o: main.cc Poligono.h Coordenada.h
        g++ -c -g -Wall main.cc
Poligono.o: Poligono.cc Poligono.h Coordenada.h
        g++ -c -g -Wall Poligono.cc
Coordenada.o: Coordenada.cc Coordenada.h
        g++ -c -g -Wall Coordenada.cc
```

[3]Para este ejercicio se supondrá que no habrán coordenadas negativas.