Programación 2

Examen de teoría (julio 2013)

12 de julio de 2013

Instrucciones

- **Duración: 3 horas**
- El fichero del primer ejercicio debe llamarse `ej1.cc`. Para el segundo problema es necesario entregar cuatro ficheros, llamados `Playlist.cc, Playlist.h, Cancion.cc, Cancion.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. **(5 puntos)**

   Queremos hacer un programa que obtenga la frecuencia de aparición de cada palabra contenida en un documento. Para ello, el programa recibirá como entrada el nombre de un fichero de texto con el documento a analizar. Un ejemplo del fichero:

   ```
   calor, sofocante calor!...sudo
   CALOR, horrible Calor, calor... la playa,la playa, ¿y la playa?
   ```

   El programa extraerá todas las palabras del texto, imprimiendo por pantalla el número de veces que aparece cada una. Para la entrada anterior:

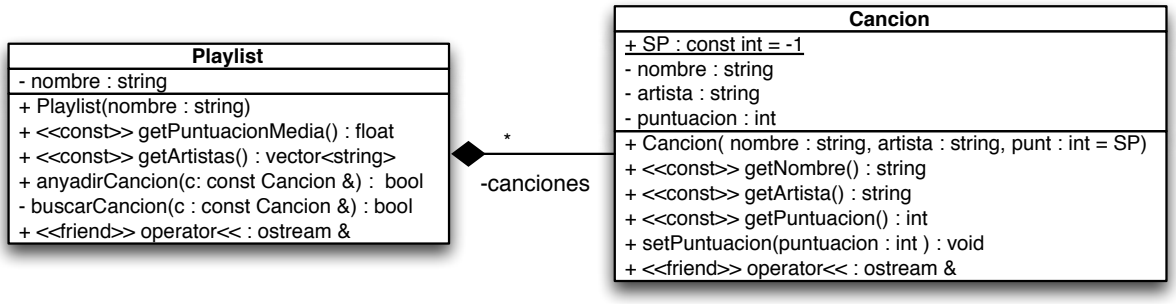
   ```
   calor: 5
   sofocante: 1
   sudo: 1
   horrible: 1
   la: 3
   playa: 3
   y: 1
   ```

   Consideraremos palabra a una secuencia de letras. Las palabras pueden tener tanto mayúsculas como minúsculas, pero es conveniente pasarlas a minúsculas cuando se lean para poder hacer la búsqueda de palabras únicas e imprimir el resultado. No hay que ordenar la lista de palabras.

   Nota: Este programa debe implementarse usando programación procedimental (no orientada a objetos). Para resolverlo te hará falta crear un registro `Palabra` y hacer un vector STL para almacenar las palabras (si no te sale con vectores, puedes usar arrays). Debe comprobarse que el número de parámetros de entrada es correcto. También deben comprobarse los errores de apertura de fichero, pero no es necesario controlar los de lectura y escritura (fail).

2. **(5 puntos)**

   Queremos hacer un programa que nos permita crear playlist (listas de reproducción) de canciones. Para ello partimos del siguiente diagrama:



Cada canción tiene un nombre, su artista y opcionalmente una puntuación. Por defecto, la puntuación será `SP` (sin puntuar). La puntuación de una canción deben estar entre 0 y 5. El método `setPuntuacion` debe comprobar que el parámetro que se le pasa esté en este rango, y si no es así imprimir un mensaje de error y dejar la puntuación como estaba. El operador salida imprimirá los datos de la canción separados por un guión. Si la canción no está puntuada, no se imprimirá su puntuación, tal como se puede ver en el ejemplo de abajo.

Un playlist tiene un nombre y una serie de canciones. El método `getPuntuacionMedia` debe obtener la puntuación media de sus canciones sin tener en cuenta las que no están puntuadas. El método `getArtistas` devuelve un listado de artistas sin duplicados.

El método `anyadirCancion` debe añadir una canción si no está en el playlist, y en caso contrario debe mostrar un mensaje de error y devolver `false`. Para buscar una canción puedes usar el método `buscarCancion`, que devuelve `true` si el nombre y el artista (los dos) coinciden con los de la canción a buscar.

A continuación se muestra un ejemplo con un fichero `main.cc`:

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

int main()
{
  Cancion c("Cadenza","Dutch Uncles",5);
  Cancion c2("The ink", "Dutch Uncles",3);
  Cancion c3("Amor imposible", "Camela");
  Cancion c4("Trainwrecks","Weezer");
  c4.setPuntuacion(5);

  Playlist p("Grandes exitos");
  p.anyadirCancion(c);
  p.anyadirCancion(c2);
  p.anyadirCancion(c3);
  p.anyadirCancion(c4);

  cout << p << endl;
}
```

Para este ejemplo, la salida del programa debería ser la siguiente:

```
Titulo: Grandes exitos
Artistas: Dutch Uncles,Camela,Weezer
-----------
Cadenza - Dutch Uncles - 5
The ink - Dutch Uncles - 3
Amor imposible - Camela
Trainwrecks - Weezer - 5
-----------
Puntuacion media= 4.33333
```

Ayuda: Puedes descargar un `makefile` de ayuda desde `http://www.dlsi.ua.es/~pertusa/exam/makefile`