PRÁCTICA Nº 5: 2 sesiones

(del 11 de Abril al 17 de Abril de 2003)

Listas con punto de interés para la composición de melodías musicales

0. OBJETIVOS

El objetivo de esta práctica es la implementación del TAD lista con punto de interés y su uso para la composición y reproducción de melodías musicales.

1. INTRODUCCIÓN

Se puede considerar una partitura como una serie de pasajes sobre los cuales se puede insertar y borrar notas individuales. Estos pasajes se incorporan en cualquier posición de la partitura una vez que se han compuesto. Las notas individuales se añaden o borran sobre un pasaje que posteriormente se inserta en la partitura en la posición deseada.

Un pasaje está compuesto por una secuencia de notas musicales (Do, Do#, Re, Re#, Mi, Fa, Fa#, Sol, Sol#, La, La#, Si). A los fines de ésta práctica las notas se repetirán en dos octavas consecutivas. Cada nota musical está unívocamente definida por una frecuencia. Además de la altura de la frecuencia de la nota se debe tener en cuenta su duración (redonda, blanca, negra, corchea, etc.).

Así, se trata de realizar un programa que nos ayude a la composición musical: crearemos diferentes pasajes que posteriormente añadiremos a la partitura.

Tanto la partitura como el pasaje se puede representar por la clase siguiente:

```
class Pasaje
{
 public:
    void IntroNota(Valor n);
    bool EliminarNota(int idno);
    void Visualizar(void);
    void LimpiarPasaje(void);
    void Tocar(void);
    bool IntroPasaje(Pasaje pas, int idno); //Añade en una cierta posición
    bool EliminarPasaje(int idno, int n);

private:
    Lista lst;
    bool BuscarIdNota(int idno);
} ;
```

Nuestro programa utilizará dos objetos de la clase pasaje: una será la partitura y otra el pasaje que se está creando en ese momento.

Sobre el pasaje se podrá:

1. Añadir una nueva nota musical.
2. Borrar una nota musical.
3. Visualizar el estado actual del pasaje
4. Borrar por completo el pasaje.

Sobre la partitura se podrá:

1. Insertar un pasaje en un punto cualquiera de la partitura
2. Borrar un pasaje indicando la posición y el número de notas a borrar
3. Visualizar la partitura
4. Reproducir la partitura.

Destacar aquí que la clase **Pasaje** se apoya en la clase **Lista** vista en clases de teoría:

```
enum Tiponota  {do_, do_s, re, re_s ,mi, fa, fa_s, sol, sol_s, la, la_s, si};
enum Tipooctava {primera, segunda};
enum Tipoduracion {negra, blanca, redonda, corchea, semicorchea};
struct Valor
{
   int iden;
   Tiponota nota;
   Tipoctava octava;
   Tipoduracion duracion;
};
struct Nodo;
typedef Nodo * Puntero;
struct Nodo
{
   Valor Info;
   Puntero Sig;
};
class Lista
{
   public:
      Lista (void);
      Lista (const Lista &);
      ~Lista (void);
      bool Insertar (Valor x);
      bool Eliminar (void);
      bool Consulta (Valor & x);
      bool ListaVacia (void);
```

```
      void IrAInicio (void);
      bool Avanzar(void);
      bool FinalLista(void);
   private:
       Puntero ini;
       Puntero fin;
       Puntero pto;
};
```

Cada nota musical está definida por un elemento del tipo de dato **`Valor`**. El tipo de dato Valor contiene los campos : **`nota`** (Do, Do#, Re, Re#, etc), **`octava`** (primera, segunda) y **`duracion`** (negra, blanca, corchea, etc.). Además, para identificar cada nota unívocamente en la melodía el tipo de dato **`Valor`** posee un campo numérico (**`iden`**).

Frecuencia y duración de las notas musicales:

La frecuencia de las notas musicales de dos octavas consecutivas son las que aparecen en el siguiente matriz constante dónde la primera fila corresponde a las frecuencias de la primera octava y la segunda fila a la segunda octava.

```
const float frec[2][12]={
    {262, 277, 294, 311, 330, 349, 370, 392, 415, 440, 466, 494}
     523, 554, 587, 622, 659, 698, 740, 784, 831, 880, 932, 988} };
```

Los valores almacenados en los campos **`nota`** y **`octava`** del tipo de dato **`Valor`** nos servirán como índices para acceder a la frecuencia de la nota musical deseada que se almacena en la matriz constante **frec**. Por ejemplo, *un do de la 1º octava* (**`frec[primera][do]`**) tendría una frecuencia de **266 Hz.**

Lo mismo ocurre con el campo **`duracion`** y el vector **`dura`** que se define a continuación:

```
const float dura[5]={1.0, 2.0, 4.0, 0.5,0.25};
```

Señalar aquí que la duración representa el tiempo que sonará la nota musical con respecto a un tiempo base que se establece en el programa :

```
const int tiempobase = 1000 ; //Equivale a 1s como tiempo base
```

Para hacer que el ordenador reproduzca una determinada nota musical se utiliza la siguiente función de C++ que tiene como argumentos la frecuencia de la nota y su duración:

```
bool Beep(int frec, int duracion)
```

incluyendo previamente el fichero:

```
#include <windows.h>
```

2. REALIZACIÓN DE LA PRÁCTICA

a) Implementación de la clase Lista con punto de interés y la clase Pasaje y comprobación de su correcto funcionamiento

En esta práctica se debe implementar la clase lista con punto de interés mediante punteros vista en clase de teoría y la clase Pasaje especificada anteriormente.

Se realizará un programa **##musica.cpp** que permita componer una partitura y reproducirla. Este programa trabajará con dos objetos de la clase Pasaje: una que representa la partitura y otra que representa el pasaje en creación. Así, tal como se indicó en la introducción, mediante un menú se podrá realizar las siguientes operaciones:

1. Añadir una nota musical al pasaje
2. Borrar una nota musical del pasaje.
3. Borrar pasaje completo.
4. Visualizar pasaje por pantalla.
5. Añadir pasaje a la partitura.
6. Borrar pasaje de la partitura.
7. Visualizar partitura por pantalla.
8. Reproducir partitura.

Opcional:

a) Reproducir la melodía al revés.

Realizar las modificaciones necesarias en la clase lista implementada para que sea posible la reproducción al revés (del final al principio de la lista).

b) Guardar partitura en fichero .

Añadir nuevas opciones al programa anterior para que permita guardar la partitura en fichero y posteriormente leerla para continuar en su composición.

4. ENTREGA DE PROGRAMAS

Al comienzo de la siguiente sesión de prácticas se entregarán al profesor tres ficheros:

1. Ficheros de la clase Lista (**`##Lista.cpp`** y **`##Lista.h`**).
2. Ficheros de la clase Pasaje (**`##Pasaje.cpp`** y **`##Pasaje.h`**).
3. Programa principal (**`##musica.cpp`**).

Nota Muy Importante

Antes de poder empezar a realizar cualquiera de las prácticas es necesario presentar las hojas de especificación de programas (documentación de programas) con las tareas que se van a realizar en la práctica, explicando brevemente como se van a solucionarse los problemas que se plantean.

ENTREGA DE PROGRAMAS: Al comenzar la sesión de prácticas del 12 al 19 de Mayo.

Canción de ejemplo:

0: Octava 0
1: Octava 1
re: redonda
bl: blanca
ne: negra
co: corchea
sc: semicorchea
S: silencio

```
0 Sol  co
0 Fa#  co
0 Sol  co
0 La   co
0 Sol  co
0 Fa   co
0 Mi   co
0 Re#  co
0 Mi   co
0 Fa   co
0 Mi   co
0 Re   co
0 Do   co
0 Mi   co
0 Sol  co
0 La   co
0 La   ne
S      co
0 La   co
0 Sol# co
0 La   co
0 Do   co
0 Si   co
0 La   co
0 Fa   co
0 Fa   co
S      sc
0 Fa   sc
S      sc
0 Fa   sc
0 La   co
0 Sol# co
0 La   co
0 Do   co
0 Si   co
0 La   co
0 Mi   co
0 Mi   co
S      sc
0 Mi   sc
S      sc
0 Mi   sc
0 Sol  co
0 Fa#  co
0 Sol  co
0 La   co
0 Sol  co
0 Fa   co
0 Mi   co
0 Re#  co
0 Mi   co
0 Fa   co
0 Mi   co
0 Re   co
0 Do   co
0 Mi   co
0 Sol  co
0 La   co
0 Si   ne
0 Si   co
0 La#  co
0 Si   co
1 Mi   co
1 Re#  co
1 Do   co
0 La   co
0 La   co
S      sc
0 La   sc
S      sc
0 La   sc
0 La   co
0 Sol# so
0 La   co
1 Do#  co
0 Si   co
0 Re#  co
0 Mi   ne
S      co
```