Tecnologías de la Información

Universidad de Huelva

Práctica 2

Asignación Dinámica de Memoria

PRÁCTICA 2: *LETRIX*

1.- Introducción

Esta práctica trata sobre la implementación, usando memoria dinámica, de la clase *Letrix*, la cual sirve para almacenar una serie de letras dispuestas según una cierta estructura.

Un ejemplo de cómo sería una posible ejecución del código completo de la práctica es:

Las letras (de la 'A' a la 'H') se van añadiendo aleatoria e incrementalmente a una estructura incialmente vacía, con la restricción de que no pueden repetirse ni horizontal ni verticalmente ninguna de las letras.

Existen 8 columnas sobre las que se dispondrán un número variable de letras.

Al inicio del programa se pide la velocidad (de 1 a 3) según la cual se van a ir insertando automáticamente las letras.

2.- Clase *Letrix*

Para almacenar las letras en cada columna se usará una estructura dinámica de nodos enlazados. Los nodos que guardarán letras (*char*) serán del tipo *NodoLetra*:

```
struct NodoLetra {
   char letra;
   NodoLetra *sig;
};
```

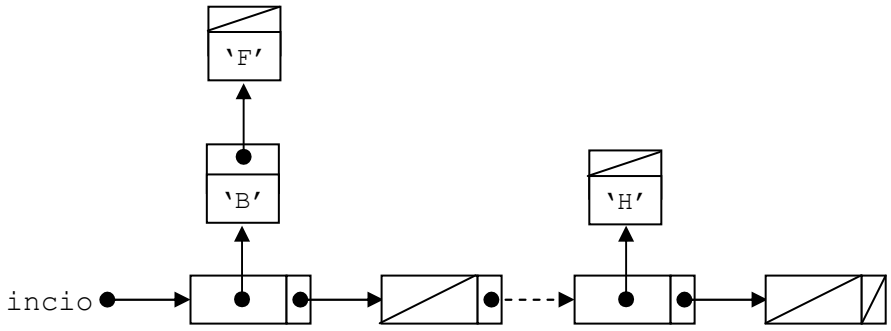
Y los nodos encargados de almacenar cada una de las columnas serán del tipo *NodoBase*:

```
struct NodoBase {
   NodoLetra *col;
   NodoBase *sig;
};
```

De esta forma, la clase *Letrix* consta simplemente de un puntero al *NodoBase* inicial:

```
class Letrix {
   private:
      NodoBase *inicio;
   public:
      Letrix();
      char letra(int f, int c);
      void insertar(int col, char let);
      void dibujar(int vel);
      ~Letrix();
};
```

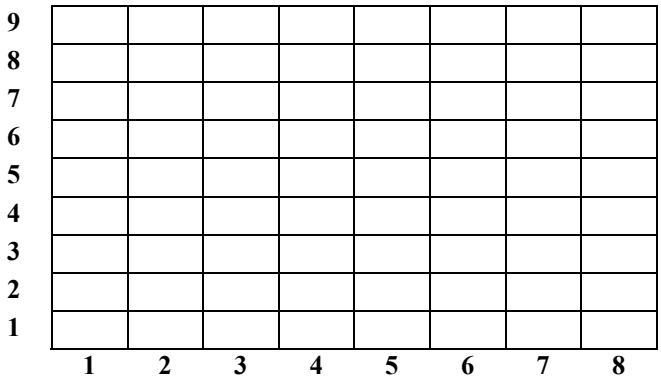
Un ejemplo gráfico de cómo podría quedar la estructura es el siguiente:



La descripción de cada método de la clase *Letrix* es:

- **Letrix**: es el constructor de la clase, que se encarga de enlazar 8 nodos de tipo *NodoBase*, cada uno de los cuales no apuntaría a ningún nodo *NodoLetra*. La estructura no guarda ninguna luetra.

- **letra**: devuelve la letra guardada en la fila y la columna dadas como argumentos. Devuelve el *char* ' ' si no se guarda ninguna letra en esa combinación de fila y columna. La posición en la estructura correspondiente a la fila inferior y columna izquierda es la (1, 1), y puesto que la estructura va a almacenar un máximo de 9 filas, las posiciones posibles para las letras se corresponden con las de la siguiente matriz:



- **insertar**: guarda una letra en la columna dada. Tras la inserción, se pasa a comprobar si la nueva letra puede mantenerse, chequeando que no se repite con ninguna letra de su misma fila ni de su misma columna. En caso de repetición, se dibuja la estructura (llamando a *dibujar*) y se procede a eliminar dicha letra insertada de la estructura.

- **dibujar**: muestra todas las letras de la estructura (desde la fila 9 a la 1, y para cada fila, de la columna 1 a la 8). Luego hace una pausa de 500**veloc* milisegundos (llamando a *espera*).

- **~Letrix**: es el destructor de la clase, que se encarga de liberar toda la memoria de todos los nodos de la estructura.

3.- Ficheros que se proporcionan

Se proporcionan los ficheros ***Letrix.h***, ***Letrix.cpp*** y ***Pract2.cpp***, con los que habría que crear un proyecto y completar el código de la clase *Letrix*.