Departamento de Lenguajes
y Ciencias de la Computación
UNIVERSIDAD DE MÁLAGA

LABORATORIO DE PROGRAMACIÓN

E.T.S.I.Informática- 1ºA GESTIÓN. 10-JUNIO-2004

Duración del Examen: 4 horas

APELLIDOS_____NOMBRE_____

ORDENADOR_____

Se nos ha solicitado la realización de una aplicación para el estudio y graficación de funciones matemáticas discretas enteras. Una función discreta puede implementarse consiste en una lista ordenada de pares (x,y). Por motivos de eficiencia se considera que lo más oportuno es el uso de una lista doblemente enlazada y ordenada por los valores de absisa (x) donde el puntero que referencia la lista se mueve a lo largo de la misma (**no está fijo en la cabeza de la lista)** y siempre apunta al último nodo visitado (insertado, leido, buscado o anterior/siguiente al eliminado). Se pide que se implemente un módulo llamado `MPunto` (`MPunto.h` , `MPunto.cpp`), un módulo llamado `MListaD` (`MListaD.h` , `MListaD.cpp`) con las operaciones que se indican más abajo y un programa llamado `grafica.cpp` con las siguientes opciones de menú:

```
Nombre: (Apellidos, Nombre)          Curso: 1º
Especialidad: (Gestión/Sistemas)     Grupo: A
Puesto: número de ordenador          Fecha: 10/06/04

MENU PRINCIPAL
==============
A. Insertar Punto desde Teclado.
B. Leer Gráfica desde Fichero de Texto.
C. Buscar Valor de la función en un punto.
D. Borrar Punto en Gráfica.
E. Pintar Lista de Puntos Pantalla/Teclado.
F. Pinta Gráfica Pantalla/Teclado.

X. Salir del Programa

Introduzca Opción:
```

Opciones:

A. Insertar Punto desde Teclado. Se solicitará al usuario los valores x e y de un punto y se insertará en la estructura. Ejemplo:

```
Coordenada X: -3
Coordenada Y: 9
```

B. Leer Gráfica desde Fichero de Texto. Se solicitará al usuario el nombre de un fichero de texto que contiene 2 valores enteros por línea y separados por un espacio y se insertarán todos los puntos en la estructura. Ejemplo:

```
Nombre del Fichero: fun.txt
```

C. Buscar Valor de la función en un punto. Se solicitará al usuario el valor de la x, se buscará en la estructura y se mostrará el valor de la y correspondiente o un mensaje indicando que no existe valor para esa x. Ejemplo:

```
Valor de x: -5                    Valor de x: -5
```

```
El punto es: -5 –10              El Punto NO se ha encontrado
```

D. Borrar Punto en Gráfica. Se solicitará al usuario el valor de la x y se borrará de la estructura el punto cuya x coincida con la solicitada. Si no existe el punto no se hará nada ni se dará ningún mensaje de error. Ejemplo:

```
Valor de x: -5
```

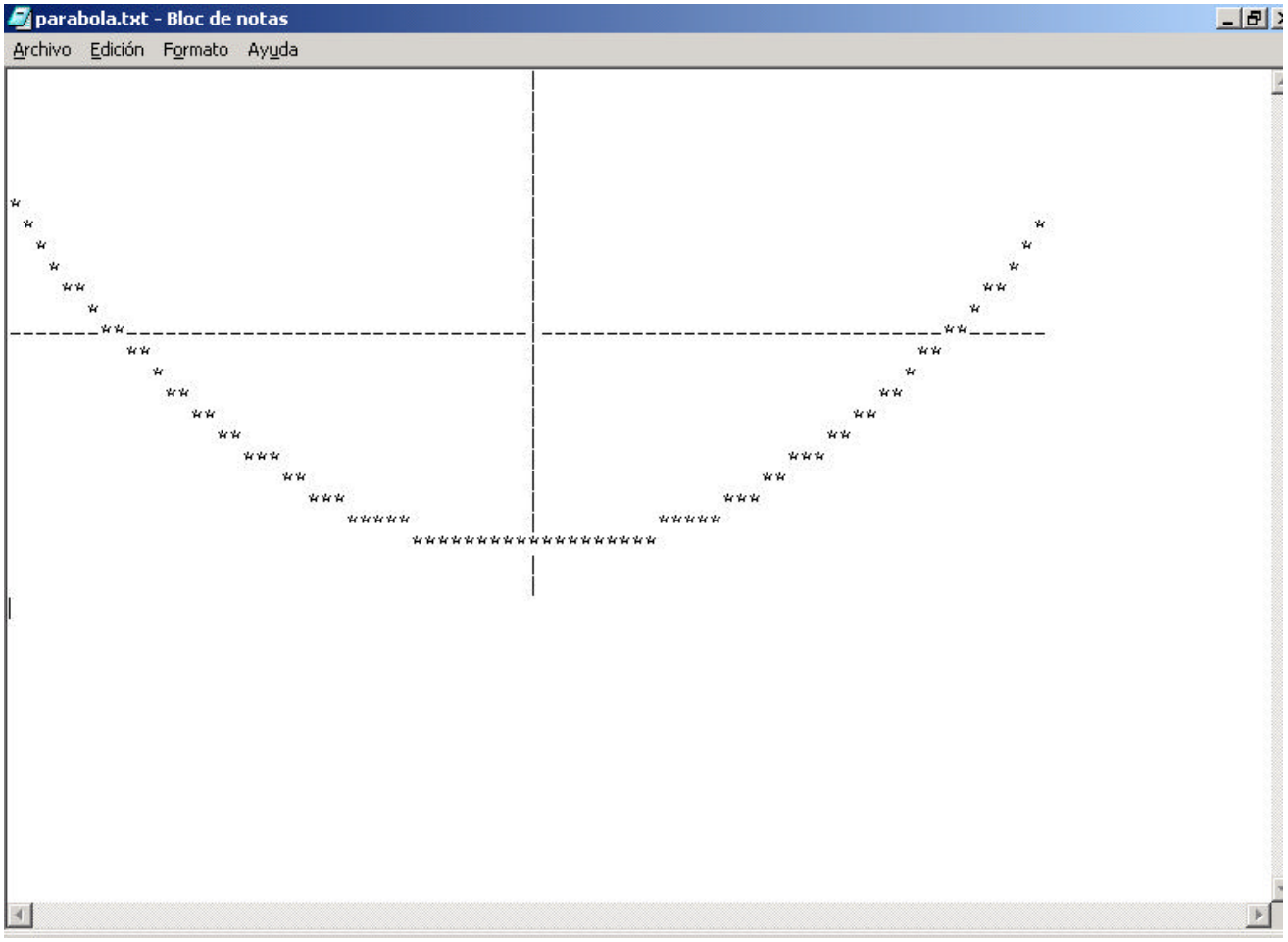
E. Pintar Lista de Puntos Pantalla/ Fichero. Se solicitará al usuario si desea la salida por Pantalla o Fichero (**deberá leerse como un enumerado**) y en caso de que sea fichero se solicitará el nombre del mismo. En Ambos casos se dará como salida la lista de puntos (un punto por líneay sólo los valores separados por un espacio). Ejemplo:

```
Salida a Pantalla o Fichero: pantalla
-4 16
-3 9
-2 4
-1 1
0 0
1 1
2 4
3 9
4 16
```

F. Pinta Gráfica Pantalla/Fichero. Se solicitará al usuario si desea la salida por Pantalla o Fichero (**deberá leerse como un enumerado**) y en caso de que sea fichero se solicitará el nombre del mismo. En Ambos supondremos que nuestra pantalla (o fichero) tiene una dimensión de 80x25, estando el eje de coordenadas (0,0) en el centro de la misma y se escribirá un * (asterisco) en las coordenadas que haya puntos, un – (guión) para pintar el eje X y | (barra vertical o pipe) para el eje Y. **ESTA PROHIBIDO EL USO DE LAS FUNCIONES GotoXY, WhereX, WhereY.**
Ejemplo: (suponiendo que hemos cargado el fichero ejemplo fun.txt):

```
Salida a Pantalla o Fichero: Fichero
Nombre del Fichero: Parabola.txt
```

Contenido de `Parabola.txt` visto con el block de notas:



X. Salir del Programa. En esta opción se pedirá confirmación de salida. Si es afirmativa, se terminará el programa, liberando todos los recursos que hayan sido reservados y si es negativa, se volverá al menú principal.

Después de cada opción se hará una pausa y se limpiará la pantalla antes de volver al menú principal.

Módulos a Implementar, además del programa principal (`grafica.cpp`):

- `MPunto` (`MPunto, MPunto`):: Módulo de manejo de Puntos.
  - Define el tipo `TCadena` como un array de caracteres.
  - Define el Tipo `TPunto` como un registro de 2 enteros.
  - Define el tipo `TSalida` como un enumerado de 2 valores: `Pantalla` y `Fichero`.
  - Define la Función `Cadena_a_TSalida` que transforma una cadena de caracteres en un enumerado del tipo `TSalida`.
  - Define la Función `TSalida_a_Cadena` que transforma un enumerado `TSalida` en la a cadena de caracteres que lo representa.
  - Define las funciones `LeerPunto, LeerPuntoFichero, EscribirPunto,` y `EscribirPuntoFichero` que leen/escriben un punto desde teclado/pantalla o dado el manejador del fichero correspondiente.
- `MListaD(MListaD.h, MListaD.cpp)`: Módulo de manejo de una LISTA DOBLEMENTE ENLAZADA Y ORDENADA por valores de x.
  - Define el tipo `TListaD`.
  - Define la Función `CrearListaD`: Crea una TListaD vacía
  - Define la Función `ListaDVacia`: Nos dice si una TListaD está vacía
  - Define la Función `ListaDLlena`: Nos dice si una TListaD está llena
  - Define la Función `InsertarListaD`: Inserta un Punto en la TListaD

- Define la Función `EliminarListaD`: ELimina (si existe) el punto cuya x se indica de la TListaD
- Define la Función `BuscarListaD`: Retorna (si existe) el punto cuya x se indica de la TListaD
- Define la Función `DestruirListaD`: Destruye una TListaD
- Define la Función MostarListaPuntos: Recibe como parámetros la lista, el tipo de salida (pantalla o fichero) y el nombre del fichero de salida (sólo se utilizará si la salida es a fichero) y muestra en la salida elegida la lista de puntos.
- Define la Función PintarGrafica: Recibe como parámetros la lista, el tipo de salida (pantalla o fichero) y el nombre del fichero de salida (sólo se utilizará si la salida es a fichero) y dibuja en la salida elegida la gráfica de la función. No habrá que pintar los puntos que estén fuera de la zona de dibujo.

EL AÑADIR OPERACIONES EN LA PARTE DE DEFINICIÓN DE UN MÓDULO, ASÍ COMO EL USO DE DETALLES DE IMPLEMENTACIÓN DE LOS MODULOS DENTRO DEL PROGRAMA PRINCIPAL SERÁ CAUSA DE SUSPENSO

Notas:

1. Todas las cadenas de caracteres tendrán un máximo de 20 caracteres.
2. Es obligatorio trabajar en el directorio **`C:\LP1AG`**. Si no existe se creará
3. Para **Aprobar** deberá ser correcta la definición de tipos, la modularización y funcionar **CORRECTAMENTE** las Opciones A,B,C, D, E y X del menú.
4. Véase el Ejemplo parabola.txt como salida generada por `fun.txt`