DECSAI
Departamento de Ciencias de la Computación e I.A.

Metodología de la Programación

Convocatoria de Junio. Curso 2011/2012

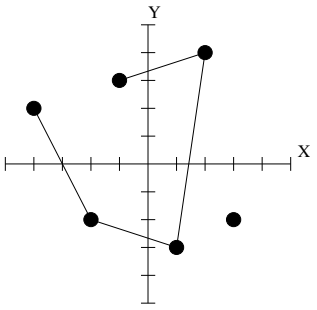
15 de Junio de 2012

Para poder gestionar figuras planas en una aplicación de gráficos 2D deseamos crear una estructura de datos capaz de representar adecuadamente esas figuras. La estructura escogida será la de una línea poligonal construida en base a una serie de puntos. Dispondremos de las clases `Punto` y `PoliLinea`

A continuación mostramos la declaración de ambas clases (sólo la representación interna) y un ejemplo de un punto en las coordenadas $(3, -2)$ y de una polilínea definida por los puntos $(-4, 2)$, $(-2, -2)$, $(1, -3)$, $(2, 4)$ y $(-1, 3)$

```
class Punto {
   int x, y;      // Coordenadas de un punto 2D
   ....
};

class PoliLinea {
   Punto *p;      // Vector de puntos
   int num;       // Número de puntos
   ....
};
```



1. Implemente los siguientes métodos para la clase `PoliLinea`:

   a) (0.5 punto) El constructor sin parámetros (crea una línea poligonal vacía) y el destructor.

   b) (1.5 puntos) El constructor de copia y el operador de asignación.

   c) (1 puntos) El operador de acceso `[]`, que permite acceder (tanto para lectura como para escritura) a un dato de tipo `Punto` en una `PoliLinea`

   d) (2 puntos) Los operadores lógicos de igualdad `==` y desigualdad `!=`. Dos datos `PoliLinea` son iguales si tienen el mismo número de puntos, éstos son iguales y están dispuestos en el mismo orden o en orden inverso (en definitiva, son iguales cuando al representarse gráficamente se obtiene la misma figura).

   e) (2 puntos) Sobrecargar el operador `+` para poder añadir un punto a una línea poligonal de manera que podamos ejecutar operaciones de la forma:

      1) polilínea + punto. Se crea una **nueva** polilínea añadiendo un punto al final de la misma.
      2) punto + polilínea. Se crea una **nueva** polilínea añadiendo un punto al inicio de la misma.

      En ambos casos, la `PoliLinea` inicial **no** se modifica.

   Si fuera preciso implementar algún método para la clase `Punto`, deberá hacerlo.

   Escribir el código correctamente modularizado en ficheros `.cpp` y `.h`.

2. Considere el siguiente formato que permite almacenar una `PoliLinea` en un fichero de texto:

   ```
   POLILINEA
   # Comentario opcional
   N
   X1 Y1
   X2 Y2
   X3 Y3
   ...
   Xn Yn
   ```

   El fichero siempre comienza por la cadena POLILINEA
   El comentario es opcional y, en caso de estar:
   - Comienza por el carácter #
   - Sólo ocupa una línea
   - No hay límite de caracteres

   N es el número de puntos que tiene la polilínea
   Despues de N tenemos la lista de coordenadas de cada punto
   - Cada punto se escribe en una nueva línea y las coordenadas están separadas por un espacio

   a) (2 puntos) Implementar un método para cargar en memoria una `PoliLinea` desde un fichero:
   `void LeerPolilinea (const char *nombre);`

   b) (1 puntos) Implementar un método para escribir en un fichero una `PoliLinea`:
   `void EscribirPolilinea (const char *nombre, const char *comentario=0);`

Duración del examen: 2 horas y media.