Programación 2009

Tarea 9

Las preguntas precedidas por un asterísco son para los alumnos de maestría. En licenciatura, dan puntos extras.

Ejercicio 1 *Crear una clase Federal, luego una clase Estatal anidada en la primera, luego una clase Municipal anidada en la segunda. Para cada de esas tres clases, definir un método imprimir(), y instanciar un objeto correspondiente en la función main, que llama a este método imprimir().*

Ejercicio 2 *Una **cola** es una estructura de datos abstracta que permite implementar la idea de FIFO (first in, firs out), como en una cola de banco. Soporta métodos push, pop, front, back que, respectivamente, ponen un nuevo elemento en la estructura (al final de la cola), quitan el elemento a la cabeza de la cola (el que llegó primero), regresan el elemento a la cabeza/al final de la cola. Usaremos una lista ligada como estructura interna, y, como queremos que la cola sea genérica (para enteros, flotantes...), usamos apuntadores (i.e. estructuras dinámicas) para almacenar los datos.*

```
class Queue {
  class Node {
    unsigned char *data;
    Node* next;
    ~Node();
    Node(void *data,unsigned int dsize);
  };
  unsigned int dataSize;
  Node *head;
  Queue(int dataSize);
  ~Queue();
  void push(void *data);
  void pop();
  void *front();
  void *back();
};
```

Partiendo de este esqueleto, implementar la cola (añadir las friendships, los controles de aceso necesarios) y probarla en la función main. Cuidado a bien liberar la memoria.

Ejercicio 3** *Uno de los inconvenientes de la estructura del ejercicio precedente es que mezcla dos cosas : la* ***especificación "abstracta" *de la cola (métodos push, pop, front, back) con la* ***implementación*** *elegida, en este caso una lista ligada. Usando el Cheshire cat, separar especificación e implementación creando una clase "visible" Queue que contendrá sólo los métodos de manipulación de la cola y un apuntador hacia otra clase QueueImpl que realmente implementa la cola con una lista ligada (como en el ejercicio precedente). Verificar que las modificaciones efectuadas en el código de QueueImpl no obligan recompilar QueueInt.*

Ejercicio 4 *Explicar la naturaleza del problema* #125 *que puede haber con el standard actual* `http://www.open-std.org/JTC1/SC22/WG21/docs/cwg_defects.html#125`

Ejercicio 5 *Re-escribir la estructura Complex de la tarea 8 como clase, con un constructor, un destructor, los métodos correspondiendo a las funciones de la tarea, y el control de aceso adecuado. Escribir un ejemplo de uso.*