VNIVERSITAT DE VALÈNCIA

Laboratorio de PROGRAMACIÓN
Grado en Ingeniería Informática (1º)
Grado en Ingeniería Multimedia (1º)
Curso 2013-14

Alumno: Julio Alejandro Ayala Roman

Alumno: Marcos Vives del Sol

Código grupo: _

_

Previo Práctica Nº 8: Listas

Fecha límite de entrega: **Domingo 18/05/2014 a las 23:55**

Material a entregar

- Pr08_Previo.doc: Este documento, incluyendo la solución a los ejercicios.

Ejercicio 1

Implementar los métodos *BuscarPorDorsal* y *BuscarPorNombre* del TAD Lista con implementación dinámica que permitan buscar un atleta mediante su dorsal o su nombre. Los métodos deben devolver *true* o *false* si han encontrado o no al atleta, estableciendo el punto de interés sobre el elemento encontrado.

```
bool Lista::BuscarPorDorsal (int dorsal)
{
      bool encontrado = false;
      int dorsal_busc;

      IrAInicio ( );
      while ( !FinalLista ( ) && !encontrado )
      {
            Consultar (dorsal_busc);
            if (dorsal_busc == dorsal)
                  encontrado = true;
            else
                  list.Avanzar ( );
      }

      return encontrado;
}

bool Lista::BuscarPorNombre (string nombre)
{
      bool encontrado = false;
      string nombre_busc;

      IrAInicio ( );
      while ( !FinalLista ( ) && !encontrado )
      {
            list.Consultar (nombre_busc);
            if (nombre_busc == nombre)
                  encontrado = true;
            else
                  list.Avanzar ( );
      }

      return encontrado;
}
```

Ejercicio 2

Diseña una función para convertir una marca de tiempo (en segundos) de un corredor en una cadena con el formato “HH:MM:SS”, rellenando con ceros las números menores que 10.

Nota: busca en Internet alguna función (estándar) de conversión de entero a cadena, y utilízala.

```
std::string convertirTiempo (int segundos)
{
      int segundo_a_s, minutos, hora, minutos_a_s;
      char buffer[10];

      segundo_a_s = segundos % 60;
      minutos_a_s = (segundos - segundo_a_s) % (60*60);
      minutos = ((segundos - segundo_a_s) % (60 * 60)) / 60;
      hora = (segundos - segundo_a_s - minutos_a_s) / (60 * 60);

      s <<hora<<":" <<minutos <<":" <<segundo_a_s;
      sprintf(buffer, “%02i:%02i%02i”,  hora,  minutos,  segundo_a_s)
      return string(buffer);
}
```

Ejercicio 3

Dada una lista con implementación dinámica, sobrecarga el operador << mediante una función amiga que permita mostrar el contenido de la lista, desde el primer hasta al último elemento, sin modificar el punto de interés.

```
ostream & operator << (ostream &out, const Lista &Lista)
{

      Puntero pto_aux = pto;
      pto_aux = cola.inicio;
    while (pto_aux != NULL)
    {
        out << pto_aux->info << endl;
        pto_aux = pto_aux->sig;
    }

}
```