SOLUCIONES AL EXAMEN DE FEBRERO DE 2000

Ejercicio 1.

```
#include <iostream.h>

class reloj
{
   unsigned int horas, minutos, segundos;
   void normalizar ( void );
   public:
   void asignar ( unsigned int h, unsigned int m, unsigned int s);//No se pide
   reloj ( int val ); // Apartado B
   reloj operator+ ( reloj ); // a+b
   reloj operator+= ( reloj ); // a+=b
   reloj operator++ ( void );  // ++a
   reloj operator++ ( int );   //a++
   reloj operator== ( reloj ); // a==b
   friend ostream& operator<< (ostream &, reloj );//Apartado B, cout << a
   friend istream& operator>> ( istream &, reloj & );//Apartado B, cin  >> a
};

void reloj::asignar ( unsigned int h, unsigned int m, unsigned int s )
{
      horas = h;
   minutos = m;
   segundos = s;
   normalizar ();
}

void reloj::normalizar ( void )
{
      unsigned int val = segundos+ 60*minutos + 60*60*horas;

   segundos = val % 60;
   minutos =  ( val / 60) % 60;
   horas = val / (60 * 60);
}

reloj::reloj ( int val =0 )
{
      segundos = val;
   horas = minutos = 0;
   normalizar ();
}

reloj reloj::operator+ ( reloj r )
{
      r.horas += horas;
   r.minutos += minutos;
   r.segundos += segundos;
   r.normalizar();
   return r;
}

reloj reloj::operator+= ( reloj r )
{
      horas += r.horas;
   minutos += r.minutos;
   horas += r.horas;
   normalizar();
```

```
   return *this;
}

reloj reloj::operator++ ( void )
{
       segundos++;
   if ( segundos>= 60) normalizar();
   return *this;
}

reloj reloj::operator++ ( int )
{
       reloj aux=*this;
   segundos++;
   if ( segundos>=60) normalizar();
   return aux;
}

ostream& operator<< ( ostream & os, reloj r )
{
   os << r.horas<<":"<<r.minutos<<":"<<r.segundos;
   return os;
}

istream & operator>> ( istream & is, reloj & r)
{
   char c1, c2;
   // Dado que el control de errores de E/S no va para el examen, se da por
   // supuesto que se introduce con el formato adecuado
   is>>r.horas>>c1>>r.minutos>>c2>>r.segundos;
   return is;
}


void main ( void )
{
   reloj r1, r2, r3;

   r1.asignar( 1, 12, 22 );
   cin >> r2;
   r3=r1+r2;
   cout << r3 << endl;
   r3++;
   cout << r3;
}
```

Ejercicio 2

```
#include <iostream.h>
#include <fstream.h>
#include <stdio.h>
#include <stdlib.h>

class Fcola
{
   fstream fp;
   char nombre[50];
   unsigned int cola;
   static int orden;
   public:
   Fcola ( void );
   Fcola ( Fcola & );
```

```
   ~Fcola() { unlink (nombre);}
   Fcola operator= ( Fcola );
   void Insertar ( int );
   int Extraer ( void );
   int Vacia ( void );
};

int Fcola::orden = 0;

Fcola::Fcola ( void )
{
   cola=0;
   sprintf ( nombre, "Fichero%d", orden++ );
   fp.open ( nombre, ios::out|ios::in|ios::binary|ios::trunc );
   if ( !fp )
   {
      cerr << "Imposible abrir el fichero " << nombre<< endl;
      exit ( 0 );
   }
}

Fcola::Fcola ( Fcola & fc )
{
   int i;

   sprintf ( nombre, "Fichero%d", orden++ );
   fp.open ( nombre, ios::out|ios::in|ios::binary|ios::trunc);
   if ( !fp )
   {
      cerr << "Imposible abrir el fichero " << nombre<< endl;
      exit ( 0 );
   }
   fc.fp.seekg ( fc.cola*sizeof(int), ios::beg);
   cola=0;
   while ( fc.fp.read( (char *)&i, sizeof(int) )  )
      fp.write ((char *) &i, sizeof(int) );
}

Fcola Fcola::operator= ( Fcola fc )
{
   char s [30];
   int i;

   fp.close();
   fp.open ( nombre, ios::out|ios::in|ios::binary );
   if ( !fp )
   {
      cerr << "Imposible abrir el fichero " << s<< endl;
      exit ( 0 );
   }
   fc.fp.seekg ( fc.cola*sizeof(int), ios::beg);
   cola=0;
   while ( fc.fp.read( (char *)&i, sizeof(int) )  )
      fp.write ((char *) &i, sizeof(int) );
   return *this;
}

void Fcola::Insertar ( int val )
{
   fp.seekp( 0, ios::end );
   fp.write ( (char *) &val, sizeof( int ));
}
```

```
int Fcola::Extraer ( void )
{
   int val;

   fp.seekg ( cola * sizeof(int), ios::beg);
   fp.read ( (char *) &val, sizeof( int ));
   cola++;
   return val;
}

int Fcola::Vacia ( void )
{
   long len;
   fp.seekg(0, ios::end);
   len=fp.tellg();
   return (len/sizeof(int)<=cola);
}


class FcolaPos: public Fcola     // APARTADO B
{
   public:
   void Insertar ( int val );
};

void FcolaPos::Insertar ( int val )
{
   if ( val < 0 ) { cerr << "Insercion positiva"; exit (0); }
   Fcola::Insertar ( val);
}

void main (void )
{
   FcolaPos Fc1;
   int i, j;

   for (int i=0; i<10; i++)
       Fc1.Insertar ( i );
   FcolaPos Fc2 = Fc1;
   while ( !Fc2.Vacia())
   {
       i=Fc2.Extraer ();
      cout << i << endl;
   }
   cin >> i;
}
```

Ejercicio 3.

```
#include <iostream.h>

class prenda
{
   int codigo, talla;
   prenda *sig;
   public:
   prenda ( int cod, int tall) {codigo=cod; tall=talla; sig=NULL;}
   friend class ListaProductos;
   virtual void mostrar ( void );
};
```

```
void prenda::mostrar ( void )
{
   cout << "codigo:"<<codigo<<endl;
   cout << "talla: "<<talla<<endl;
}

class pantalon: public prenda
{
   int longitud;
   public:
   pantalon( int cod, int tall, int lon ): prenda(cod,tall) { longitud=lon;}
   void mostrar ( void ) { prenda::mostrar();
                                                    cout <<"longitud:"<<longitud;
                                               }
};

class camisa: public prenda
{
   int cl;
   public:
   camisa(int cod, int tall, int c): prenda(cod,tall) { cl=c;}
   void mostrar ( void ) { prenda::mostrar();
                                                    cout <<"cl:"<<cl;
                                               }
};


class ListaProductos
{
   prenda *cap;
   public:
   ListaProductos ( void ) { cap=NULL; }
   void Insertar ( void );
   void Listar ( void );
};

void ListaProductos::Insertar ( void )
{
   prenda *p;
   int ptalla, pcodigo;
   char c;

   cout << "Camisa o pantalon:";
   cin>>c;
   cout <<"Talla:"; cin>>ptalla;
   cout <<"codigo:";cin>>pcodigo;
   if (c=='c')
   {
       int pcl;
      cout<<"Manga corta o larga (0/1)";
      cin>>pcl;
      p=new camisa (pcodigo,ptalla,pcl);
   }
   else
   {
       int len;
      cout<<"longitud:";
      cin>>len;
      p=new pantalon(pcodigo,ptalla,len);
   }
   p->sig=cap;
   cap=p;
}
```

```
void ListaProductos::Listar ( void )
{
   prenda *aux;

   for (aux=cap; aux!=NULL; aux=aux->sig )
       aux->mostrar();
}
```