```
#ifndef LISTAPOS
#define LISTAPOS
#include <iostream.h>
#define OK  0
#define ERR -1

class ListaPos
{
   struct NodoPos
   {
      int clave;
      int offset;
      NodoPos *sig;
   };
   NodoPos *Cap;
   public:
   ListaPos(void) {Cap=NULL;}
   ~ListaPos();
   ListaPos (const ListaPos &);
   int Insertar (int, int);
   int Leer (int, int &);
   int Borrar (int);
   void Listar ();
};

ListaPos::~ListaPos ()
{
   NodoPos *aux;

   while (Cap!=NULL)
   {
      aux=Cap;
      Cap=Cap->sig;
      delete aux;
   }
}

ListaPos::ListaPos (const ListaPos &L)
{
   NodoPos *aux;

   Cap=NULL;
   aux=L.Cap;
   while (aux!=NULL)
   {
      Insertar(aux->clave, aux->offset);
      aux=aux->sig;
   }
}

int ListaPos::Insertar (int clave, int off)
{
   NodoPos *aux;
   aux=new NodoPos;
   aux->clave=clave;
   aux->offset=off;
   aux->sig=NULL;
   if (Cap==NULL)
   {
      Cap=aux;
      return OK;
   }
   NodoPos *ant, *act;
   int seguir=(Cap->clave<clave);
   ant=NULL;
   act=Cap;
   while( act!=NULL && seguir)
      if(seguir=act->clave<clave)
      {
         ant=act;
         act=act->sig;
      }
   if(act==NULL)
   {
```

```
      ant->sig=aux;
      return OK;
   }
   if(act->clave==clave)
      return ERR;
   if(ant==NULL)
   {
      aux->sig=Cap;
      Cap=aux;
   }
   else
   {
      ant->sig=aux;
      aux->sig=act;
   }
   return OK;
}

int ListaPos::Leer (int clave, int &off)
{
   NodoPos *aux;
   int encontrado=0;

   for(aux=Cap; aux!=NULL && !encontrado; )
      if(!(encontrado=aux->clave==clave))
         aux=aux->sig;
   if(aux)
   {
      off=aux->offset;
      return OK;
   }
   else
      return ERR;
}

int ListaPos::Borrar (int clave)
{
   NodoPos *act, *ant;

   if(Cap==NULL) return ERR;
   if(Cap->clave==clave)
   {
      act=Cap;
      Cap=Cap->sig;
      delete act;
      return OK;
   }
   ant=Cap;
   act=Cap->sig;
   int seguir=1;
   while (act!=NULL && seguir)
      if(seguir=act->clave!=clave)
      {
         ant=act;
         act=act->sig;
      }
   if (act==NULL)
      return ERR;
   ant->sig=act->sig;
   delete act;
   return OK;
}

void ListaPos::Listar ()
{
   for (NodoPos *aux=Cap; aux!=NULL; aux=aux->sig)
      cout << "clave:"<<aux->clave<<" offset:"<<aux->offset<<endl;
}

//#define LISTAPOS_TEST
#ifdef LISTAPOS_TEST
void main ()
{
   ListaPos L;
```

```
    int clave, off;
    int seguir=1;

    while(seguir)
    {
       cout<<"0. Salir"<<endl;
       cout<<"1. Insertar"<<endl;
       cout<<"2. Leer"<<endl;
       cout<<"3. Borrar"<<endl;
       cout<<"4. Listar"<<endl;
       cin>>seguir;
       switch (seguir)
       {
          case 1:  cout<<"Deme clave y offset: ";
                   cin>>clave>>off;
                   if(L.Insertar(clave, off)!=OK)
                      cout<<"Elemento repetido. Inserción no realizada."<<endl;
                   break;
          case 2:  cout<<"Deme la clave:";
                   cin>>clave;
                   if(L.Leer(clave, off)==OK)
                      cout<<"El offset es:"<<off<<endl;
                   else
                      cout<<"Elemento no encontrado"<<endl;
                   break;
          case 3:  cout<<"Deme la clave:";
                   cin>>clave;
                   if(L.Borrar(clave)!=OK)
                      cout<<"Elemento no encontrado"<<endl;
                   break;
          case 4:  L.Listar();
                   break;
       }
    }
}
#endif
#endif
```