Ejercicios Resueltos

1) Programa para ingresar n valores reales en un Vector y luego invertir el vector.

Archivo Unit2.h

```
const int LIM=100;
void ingresoVector(float x[], int n, TListBox *lst);
void mostrarVector(float x[], int n, TListBox *lst);
void invertirVector(float x[], int n);
```

Archivo Unit2.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop

#include "Unit2.h"
#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
  {
     x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
     lst->Items->Add(x[i]);
  }
}

void mostrarVector(float x[], int n, TListBox *lst)
{
  int i;
```

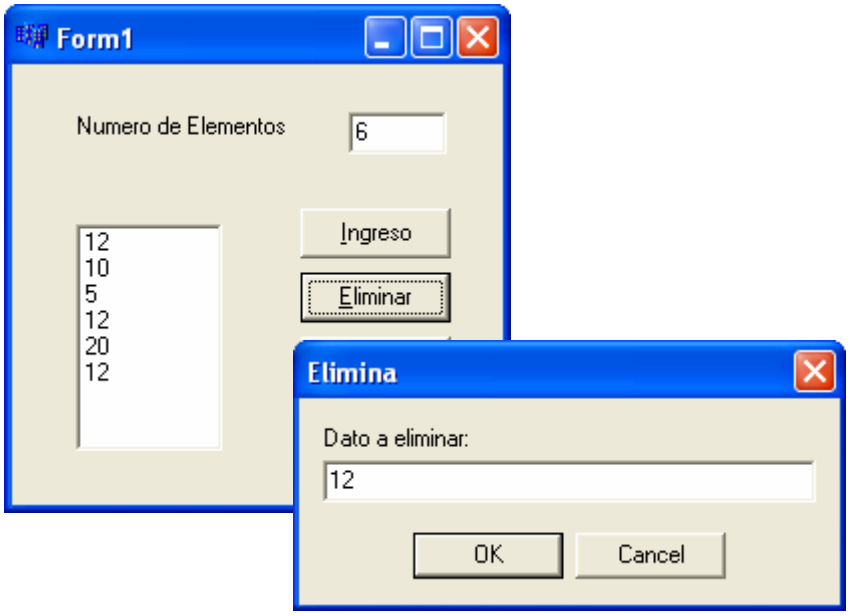
```
  lst->Items->Clear();
  for(i=0;i<n;i++)
     lst->Items->Add(x[i]);
}

void invertirVector(float x[], int n)
{
  int i,j;
  float temp;
  for(i=0,j=n-1;i<n/2;i++,j--)
  {
    temp=x[i];
    x[i]=x[j];
    x[j]=temp;
  }
}
```

Archivo Unit1.cpp

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
float x[LIM];
int n;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::btnIngresoClick(TObject *Sender)
{
   n=edN->Text.ToInt();
   if(n<=LIM)
   {
      ingresoVector(x,n,lstNum);
      btnInvertir->SetFocus();
   }
   else
      {
        ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
        edN->Clear();
        edN->SetFocus();
      }
}
```

```
//---------------------------------------------------------------------------
void __fastcall TForm1::btnInvertirClick(TObject *Sender)
{
  invertirVector(x,n);
  mostrarVector(x,n,lstNum);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
 edN->Clear();
 lstNum->Items->Clear();
 edN->SetFocus();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
 Close();
}
//---------------------------------------------------------------------------
```

4) Programa para ingresar n elementos en un vector y luego ingresar un elemento, si este se encuentra en el vector eliminarlo todas las veces que se encuentra



Archivo unit2.h

```
//---------------------------------------------------------------------------
#include<vcl.h>
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif

const int LIM=100;
void ingresoVector(float x[], int n, TListBox *lst);
void mostrarVector(float x[], int n, TListBox *lst);
void elimina(float x[], int &n, int p);
void eliminaElementoTodasLasVeces(float x[], int &n,float dato);
```

Archivo unit2.cpp

```
//---------------------------------------------------------------------------


#pragma hdrstop

#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
  {
     x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
     lst->Items->Add(x[i]);
  }
}

void mostrarVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
     lst->Items->Add(x[i]);
}

void elimina(float x[], int &n, int p)
{
  int i;
  for(i=p;i<n-1;i++)
    x[i]=x[i+1];
  n=n-1;
}

void eliminaElementoTodasLasVeces(float x[], int &n,float dato)
{
  int i;
  for(i=0;i<n;i++)
  {
    if(x[i]==dato)
    {
       elimina(x,n,i);
       i--;
    }
  }
}
```

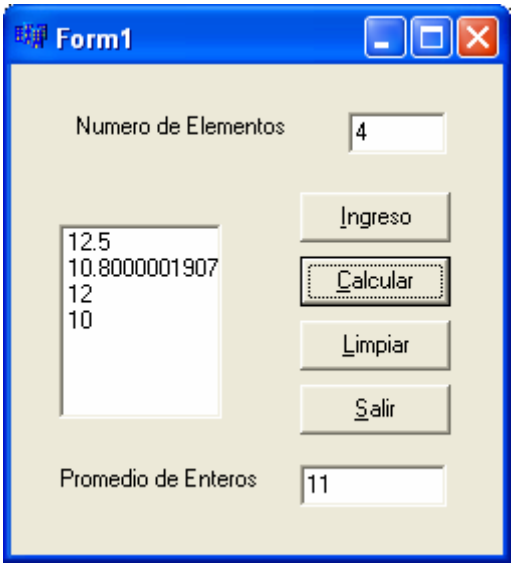
Archivo unit1.cpp

```
//---------------------------------------------------------------------------
```

```
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"

TForm1 *Form1;
float x[LIM];
int n;

//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::btnIngresoClick(TObject *Sender)
{
 n=edN->Text.ToInt();
  if(n<=LIM)
  {
     ingresoVector(x,n,lstNum);
     btnEliminar->SetFocus();
  }
  else
     {
       ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
       edN->Clear();
       edN->SetFocus();
     }

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnEliminarClick(TObject *Sender)
{
  float dato;
  dato=InputBox("Elimina","Dato a eliminar:","").ToDouble();
  eliminaElementoTodasLasVeces(x,n,dato);
  mostrarVector(x,n,lstNum);
}

//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
   edN->Clear();
   lstNum->Items->Clear();
   edN->SetFocus();
}

//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------------
```

7) Sea un Vector de tipo float de n elementos. Encuentre el promedio de los numeros enteros ingresados en el vector



Archivo unit2.h

```
//---------------------------------------------------------------------------
#include<vcl.h>
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif
const int LIM=100;
void ingresoVector(float x[], int n, TListBox *lst);
float promedioEnteros(float x[], int n);
```

Archivo unit2.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
  {
    x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
    lst->Items->Add(x[i]);
  }
}
```

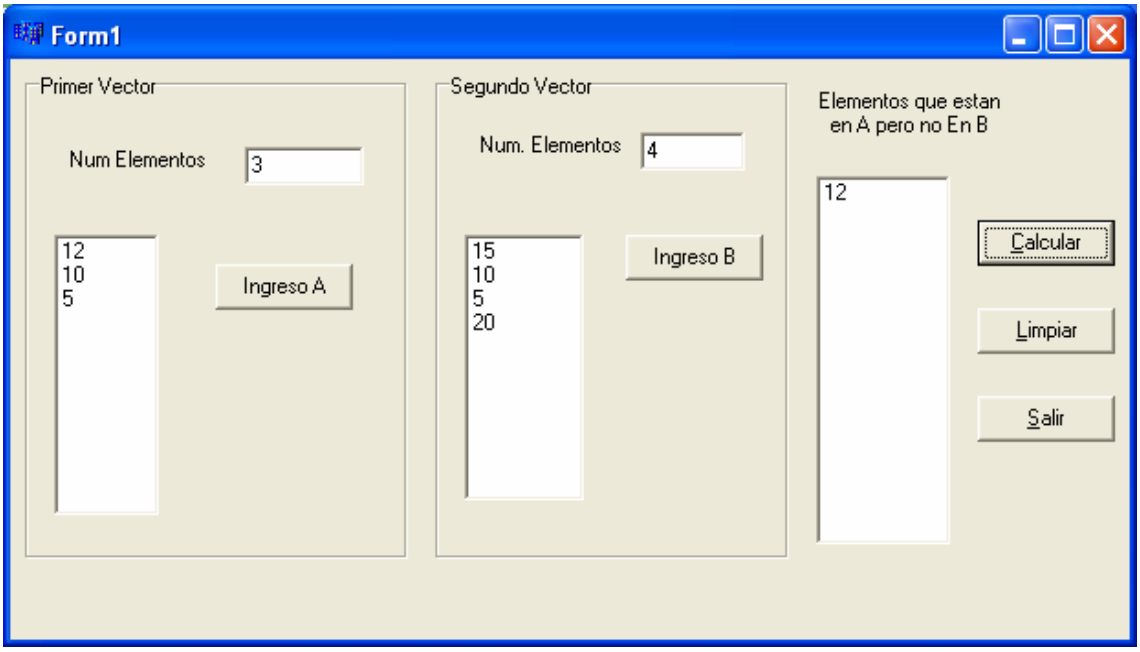
```
float promedioEnteros(float x[], int n)
{
  int i,c=0;
  float s=0;
  for(i=0;i<n;i++)
  {
     if(x[i]==int(x[i]))
     {
       s=s+x[i];
       c++;
     }
  }
  if(c>0)
       return s/c;
  else
       return 0;
}
```

Archivo unit1.cpp

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
float x[LIM];
int n;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::btnIngresarClick(TObject *Sender)
{
   n=edN->Text.ToInt();
   if(n<=LIM)
   {
      ingresoVector(x,n,lstNum);
      btnCalcular->SetFocus();
   }
   else
      {
        ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
        edN->Clear();
        edN->SetFocus();
      }
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::btnCalcularClick(TObject *Sender)
{
  edProm->Text=promedioEnteros(x,n);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
  edN->Clear();
  lstNum->Items->Clear();
  edN->SetFocus();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
   Close();
}
//---------------------------------------------------------------------------
```

9) Lea dos Vectores A y B y luego diga que elementos del vector A no se encuentran en B.



Archivo unit2.h

```
//---------------------------------------------------------------------------
#include<vcl.h>
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif
const int LIM=100;

void ingresoVector(float x[], int n, TListBox *lst);
void mostrarVector(float x[], int n, TListBox *lst);
int buscar(float x[], int n, float dato);
void diferencia(float a[], int n, float b[], int m, float c[], int &p);
```

Archivo unit2.cpp

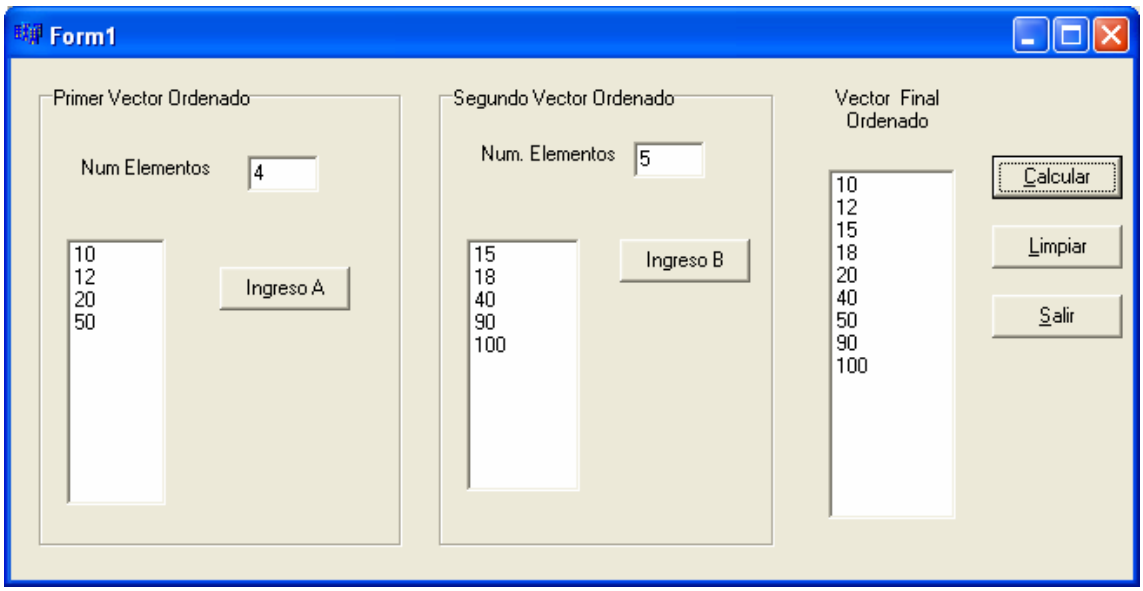
```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
  {
     x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
     lst->Items->Add(x[i]);

  }
}

void mostrarVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
     lst->Items->Add(x[i]);
}

int buscar(float x[], int n, float dato)
{
  int i;
  for(i=0;i<n;i++)
  {
    if(x[i]==dato)
      return i;
  }
  return -1;
}

void diferencia(float a[], int n, float b[], int m, float c[], int &p)
{
   int i;

   p=0;

   for(i=0;i<n;i++)
   {
     if(buscar(c,p,a[i])==-1 && buscar(b,m,a[i])==-1)
     {
        c[p]=a[i];
        p++;
     }
   }
}
```

Archivo unit1.cpp

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
float A[LIM],B[LIM],C[LIM];
int n,m,p;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm1::btnIngresoPrimerVectorClick(TObject *Sender)
{
  n=edN->Text.ToInt();
  if(n<=LIM)
  {
    ingresoVector(A,n,lstA);
    edM->SetFocus();
  }
  else
     {
       ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
       edN->Clear();
       edN->SetFocus();
     }

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnIngresoSegundoVectorClick(TObject *Sender)
{
  m=edM->Text.ToInt();
  if(m<=LIM)
  {
    ingresoVector(B,m,lstB);
    btnCalcular->SetFocus();
  }
  else
     {
       ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
       edM->Clear();
       edM->SetFocus();
     }
}
```

```
//---------------------------------------------------------------------------
void __fastcall TForm1::edNKeyPress(TObject *Sender, char &Key)
{
  if(Key==13 && !edN->Text.IsEmpty())
     edM->SetFocus();
  else
     if((Key<48 || Key>57) && Key!=8)
       Key=0;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::edMKeyPress(TObject *Sender, char &Key)
{
  if(Key==13 && !edN->Text.IsEmpty())
     btnCalcular->SetFocus();
  else
     if((Key<48 || Key>57) && Key!=8)
       Key=0;


}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnCalcularClick(TObject *Sender)
{

  diferencia(A,n,B,m,C,p);
  if(p!=-1)
    mostrarVector(C,p,lstC);

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
  edN->Clear();
  edN->Clear();
  lstA->Items->Clear();
  lstB->Items->Clear();
  lstC->Items->Clear();
  edN->SetFocus();
  n=0;
  m=0;
  p=0;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
  Close();

}
//---------------------------------------------------------------------------
```

15) Se tienen 2 vectores ordenados y se desea unirlos en un tercero pero manteniendo los datos ordenados (Sin usar metodos de ordenamiento)



Unit2.h

```
//---------------------------------------------------------------------------
#include<vcl.h>

#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif

const int LIM=100;

void ingresoVector(float x[], int n, TListBox *lst);
void mostrarVector(float x[], int n, TListBox *lst);
void unirOrdenado(float A[], int n, float B[], int m, float C[], int &p);
bool estaOrdenadoAscendentemente(float A[], int n);
```

Unit2.cpp

```
//---------------------------------------------------------------------------


#pragma hdrstop

#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
```

```
  for(i=0;i<n;i++)
  {
     x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
     lst->Items->Add(x[i]);

  }
}
void mostrarVector(float x[], int n, TListBox *lst)
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
     lst->Items->Add(x[i]);
}

bool estaOrdenadoAscendentemente(float x[], int n)
{
   int i;

   for(i=0;i<n-1;i++)
   {
     if(x[i]>x[i+1]) return false;
   }
   return true;
}

void unirOrdenado(float A[], int n, float B[], int m, float C[], int &p)
{
  int i=0,j=0;
  p=0;
  while(i<n && j<m)
  {
     if(A[i]<=B[j])
     {
        C[p]=A[i];
        i++;
     }
     else
     {
       C[p]=B[j];
       j++;
     }
     p++;
  }
  while(i<n)
  {
     C[p]=A[i];
     i++;
     p++;
  }
  while(j<m)
  {
     C[p]=B[j];
     j++;
     p++;
```

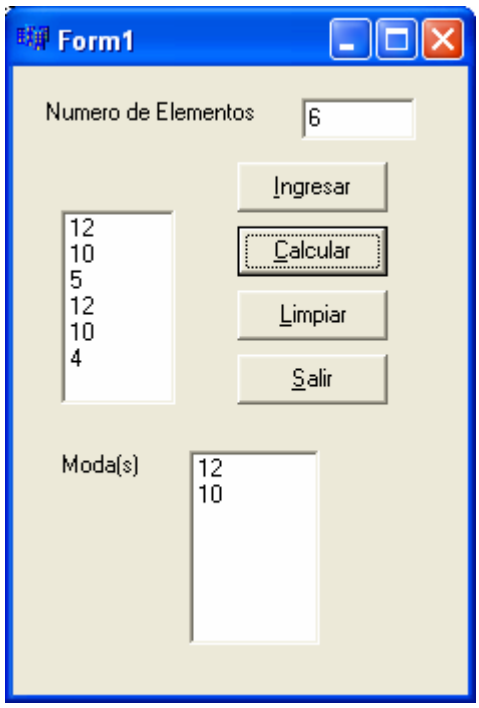
```
 }
}
```

Archivo Unit1.cpp

```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"

//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
float A[LIM],B[LIM],C[LIM];
int n,m,p;

//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnIngresoPrimerVectorClick(TObject *Sender)
{
  n=edN->Text.ToInt();
  if(n<=LIM)
  {
    ingresoVector(A,n,lstA);
    if(estaOrdenadoAscendentemente(A,n))
      edM->SetFocus();
    else
       {
         ShowMessage("El vector debe estar ordenado, Ingrese de Nuevo");
         edN->Clear();
         lstA->Items->Clear();
         n=0;
         edN->SetFocus();
       }
  }
  else
     {
       ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
       edN->Clear();
       edN->SetFocus();
     }
}

//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::btnIngresoSegundoVectorClick(TObject *Sender)
{
  m=edM->Text.ToInt();
  if(m<=LIM)
  {
    ingresoVector(B,m,lstB);
    if(estaOrdenadoAscendentemente(B,m))
      btnCalcular->SetFocus();
    else
       {
         ShowMessage("El vector debe estar ordenado, Ingrese de Nuevo");
         edM->Clear();
         lstB->Items->Clear();
         m=0;
         edM->SetFocus();
       }
  }
  else
     {
      ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
      edM->Clear();
      edM->SetFocus();
     }
}
//---------------------------------------------------------------------------

void __fastcall TForm1::btnCalcularClick(TObject *Sender)
{
 unirOrdenado(A,n,B,m,C,p);
 mostrarVector(C,p,lstC);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
  edN->Clear();
  edM->Clear();
  lstA->Items->Clear();
  lstB->Items->Clear();
  lstC->Items->Clear();
  edN->SetFocus();
  n=0;
  m=0;
  p=0;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------------
```

16) La Moda de un conjunto de datos es el elemento que mas se repite. Encuentre la(s) moda(s) de elementos almacenados en un vector.



Archivo Unit2.h

```
//---------------------------------------------------------------------------

#include<vcl.h>
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif

const int LIM=100;

void ingresoVector(float x[], int n, TListBox *lst);
int buscar(float x[], int n, float dato);
void calculoModa(float x[], int n,TListBox *lst);
```

Archivo Unit2.cpp

```
//---------------------------------------------------------------------------


#pragma hdrstop

#include "Unit2.h"

//---------------------------------------------------------------------------

#pragma package(smart_init)

void ingresoVector(float x[], int n, TListBox *lst)
```

```
{
  int i;
  lst->Items->Clear();
  for(i=0;i<n;i++)
  {
     x[i]=InputBox("Ingreso","elemento["+IntToStr(i)+"]:","").ToDouble();
     lst->Items->Add(x[i]);

  }

}
int buscar(float x[], int n, float dato)
{
  int i;
  for(i=0;i<n;i++)
  {
    if(x[i]==dato)
      return i;
  }
  return -1;
}

void calculoModa(float x[], int n,TListBox *lst)
{
   float dato[LIM];
   int cont[LIM],c,may,p=0,i,j;

   for(i=0;i<n;i++)
   {
     if(buscar(dato,p,x[i])==-1)
     {
        c=0;
        for(j=i+1;j<n;j++)
          if(x[i]==x[j]) c++;
        dato[p]=x[i];
        cont[p]=c;
        p++;
     }
   }
   may=cont[0];
   for(i=1;i<p;i++)
   {
     if(cont[i]>may)
        may=cont[i];
   }
   for(i=0;i<p;i++)
   {
     if(cont[i]==may)
       lst->Items->Add(dato[i]);
   }
}

//---------------------------------------------------------------------------
```

Archivo unit1.cpp

```
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
float x[LIM];
int n;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnIngresarClick(TObject *Sender)
{
  n=edN->Text.ToInt();
  if(n<=LIM)
  {
    ingresoVector(x,n,lstNum);
    btnCalcular->SetFocus();
  }
  else
     {
       ShowMessage("Cantidad Invalida, La cantidad Maxima es :"+IntToStr(LIM));
       edN->Clear();
       edN->SetFocus();
     }

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnCalcularClick(TObject *Sender)
{
 calculoModa(x,n,lstModa);

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
  edN->Clear();
  lstNum->Clear();
  lstModa->Clear();
  edN->SetFocus();
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------------
```

20) Ingresar los nombres y las notas de n alumnos y reportar una lista en orden Alfabético y una lista en orden de merito

Unit2.h

```
//---------------------------------------------------------------------------
#include<vcl.h>
#ifndef Unit2H
#define Unit2H
//---------------------------------------------------------------------------
#endif
const int LIM=100;
void ingreso(String nombres[], float notas[], int n, TListBox *lst1,  TListBox *lst2);
void mostrar(String nombres[], float notas[], int n, TListBox *lst1, TListBox *lst2);
void ordenAlfabetico(String nombres[], float notas[], int n);
void ordenDeMerito(String nombres[], float notas[], int n);
```

Unit2.cpp

```
//---------------------------------------------------------------------------
#pragma hdrstop
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
void ingreso(String nombres[], float notas[], int n, TListBox *lst1,
          TListBox *lst2)
{
  int i;
  lst1->Items->Clear();
  lst2->Items->Clear();
  for(i=0;i<n;i++)
  {
    nombres[i]=InputBox("Ingreso","Nombre["+IntToStr(i)+"]:","");
    notas[i]=InputBox("Ingreso","Nota["+IntToStr(i)+"]:","").ToDouble();
    lst1->Items->Add(nombres[i]);
    lst2->Items->Add(notas[i]);
  }
}


void mostrar(String nombres[], float notas[], int n, TListBox *lst1,
```

```
          TListBox *lst2)
{
  int i;
  lst1->Items->Clear();
  lst2->Items->Clear();
  for(i=0;i<n;i++)
  {
    lst1->Items->Add(nombres[i]);
    lst2->Items->Add(notas[i]);
  }
}

void ordenAlfabetico(String nombres[], float notas[], int n)
{
 int i,j;
 String temp1;
 float temp2;
 for(i=0;i<n-1;i++)
   for(j=i+1;j<n;j++)
     if(nombres[i]>nombres[j])
     {
        temp1=nombres[i];
        nombres[i]=nombres[j];
        nombres[j]=temp1;
        temp2=notas[i];
        notas[i]=notas[j];
        notas[j]=temp2;
     }
}
void ordenDeMerito(String nombres[], float notas[], int n)
{
 int i,j;
 String temp1;
 float temp2;
 for(i=0;i<n-1;i++)
   for(j=i+1;j<n;j++)
     if(notas[i]<notas[j])
     {
        temp1=nombres[i];
        nombres[i]=nombres[j];
        nombres[j]=temp1;
        temp2=notas[i];
        notas[i]=notas[j];
        notas[j]=temp2;
     }
}
//---------------------------------------------------------------------------
```

Unit1.cpp

```
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "Unit2.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
String nombres[LIM];
float notas[LIM];
int n;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm1::btnIngresoClick(TObject *Sender)
{
  if(n<=LIM)
  {
    n=edN->Text.ToInt();
    ingreso(nombres,notas,n,lstNombres,lstNotas);
  }
  else
     ShowMessage("Cantidad invalida, La cantidad maxima es : "+IntToStr(LIM));
}
//---------------------------------------------------------------------------

void __fastcall TForm1::btnOrdenAlfabeticoClick(TObject *Sender)
{
  ordenAlfabetico(nombres,notas,n);
  mostrar(nombres,notas,n,lstNombres,lstNotas);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnOrdenDeMeritoClick(TObject *Sender)
{
  ordenDeMerito(nombres,notas,n);
  mostrar(nombres,notas,n,lstNombres,lstNotas);
}

void __fastcall TForm1::edNKeyPress(TObject *Sender, char &Key)
{
    if(Key==13 && !edN->Text.IsEmpty())
      btnIngreso->SetFocus();
    else
        if((Key<48 || Key>57) && Key!=8)
            Key=0;

}
//---------------------------------------------------------------------------
```

```
//---------------------------------------------------------------------------
void __fastcall TForm1::btnLimpiarClick(TObject *Sender)
{
  edN->Clear();
  lstNombres->Items->Clear();
  lstNotas->Items->Clear();
  n=0;
  edN->SetFocus();

}
//---------------------------------------------------------------------------
void __fastcall TForm1::btnSalirClick(TObject *Sender)
{
  Close();
}
//---------------------------------------------------------------------------
```