```
unit StackArray;

interface

Uses
  Tipos, stdctrls, SysUtils, Variants;

Const
  MIN = 1;  // Base de la Pila
  MAX = 2000; // Maximo de la Pila
  Nulo= 0;  // Posicion NO valida del TOPE de la Pila

Type
  PosicionPila = Integer;

  Pila = Object
    Private
      Elementos: Array Of TipoElemento;
      Inicio: PosicionPila;   // manejo del tope
      Q_Items: Integer;       // cantidad de elementos. Solo de uso interno.
      TDatoDeLaClave: TipoDatosClave;
      Size : LongInt;
      Function CantidadElementos(): LongInt;
    Public
      Function Crear(avTipoClave: TipoDatosClave; alSize: LongInt): Resultado;
      Function EsVacia(): Boolean;
      Function EsLlena(): Boolean;
      Function Apilar(X:TipoElemento): Resultado;
      Function DesApilar(): Resultado;
      Function Recuperar(): TipoElemento;
      Function RetornarClaves(): String;
      Function InterCambiar(Var PAux: Pila; bCrearVacia: Boolean): LongInt;
      Function LlenarClavesRandom(alSize: LongInt; RangoDesde, RangoHasta: LongInt): Resultado;
      Function Tope():PosicionPila;
      Function DatoDeLaClave: TipoDatosClave;
      Function SizeStack(): LongInt;
      Function MaxSizeStack(): LongInt;
  End;

implementation

// Crea la pila vacia
Function Pila.Crear(avTipoClave: TipoDatosClave; alSize: LongInt): Resultado;
Begin
  if alSize < Min then Crear:= CError;
  if alSize > Max then Crear:= CError;
  if (alSize >= Min) And (alSize <= Max) then Begin
    SetLength(Elementos, (alSize+1));
    Inicio := Nulo;
    Q_Items := 0;
    TDatoDeLaClave := avTipoClave;
    Size := alSize;
    Crear := OK;
  End;
End;

// Control de pila vacia
Function Pila.EsVacia(): Boolean;
Begin
  EsVacia := (Inicio = Nulo);
End;

// Control de pila llena
Function Pila.EsLlena(): Boolean;
Begin
  EsLLena := (Inicio = Size);
End;

// Agrega un Items a la Pila
Function Pila.Apilar(X:TipoElemento): Resultado;
Begin
  Apilar := CError;
```

```
  // Controla que el Tipo de Dato de la Clave sea Homogeneo a la Lista
  if X.TipoDatoClave(X.Clave) <> TDatoDeLaClave then Begin
    Apilar := ClaveIncompatible;
    Exit;
  End;
  // ahora lo apilo
  If EsLlena() Then Apilar := LLena
  Else Begin
    Inicio := Inicio + 1;
    Elementos[Inicio] := X;
    Inc(Q_Items);
    Apilar := OK;
  End;
End;

// Elimina un item de la Pila. Siempre del Tope
Function Pila.DesApilar(): Resultado;
Begin
  DesApilar := CError;
  If EsVacia() Then DesApilar := Vacia
  Else Begin
    Dec(Inicio);
    Dec(Q_Items);
    DesApilar := OK;
  End;
End;

// retorna por referencia en X el item de la Pila. Siempre el Tope
Function Pila.Recuperar(): TipoElemento;
Var X: TipoElemento;
Begin
  Recuperar := X.TipoElementoVacio;
  If Not EsVacia() Then Begin
    Recuperar := Elementos[Inicio];
  End;
End;

// Pasa los Items de una Pila Auxiliar a la Pila "Pila"
Function Pila.InterCambiar(Var PAux: Pila; bCrearVacia: Boolean): LongInt;
Var X: TipoElemento;
    I: LongInt;
Begin
  I := 0;
  If bCrearVacia = true Then Crear(TDatoDeLaClave, Size);
  While Not PAux.EsVacia() Do Begin
    X := PAux.Recuperar();
    If Apilar(X) = OK Then Inc(I);
    PAux.DesApilar;
  End;
  InterCambiar := I;
End;

// Retorna un string con todos los elementos de Pila
// Cada Item separado por Retorno de Carro + Final de Linea
Function Pila.RetornarClaves():String;
Var X: TipoElemento;
    S, SS: String;
    PAux: Pila;
Begin
  SS:= '';
  PAux.Crear(TDatoDeLaClave, Size);
  While Not EsVacia() Do Begin
    X := Recuperar();
    PAux.Apilar(X);
    S := X.ArmarString;
    SS := SS + S + cCRLF;
    DesApilar();
  End;
  InterCambiar(Paux, True);
  RetornarClaves := SS;
End;
```

```
// Llena la pila con valores random en el atributo DI
// desde 0 (cero) hasta <RangoHasta>
Function Pila.LlenarClavesRandom(alSize: LongInt; RangoDesde, RangoHasta: LongInt): Resultado;
Var X: TipoElemento;
Begin
  Randomize;
  TDatoDeLaClave := Numero;
  If Crear(TDatoDeLaClave, alSize) <> OK Then Begin
    LlenarClavesRandom := CError;
    Exit;
  End;
  // Ahora la LLeno random
  X.Inicializar(TDatoDeLaClave,'');
  While Not EsLLena() Do Begin
    X.Clave := RangoDesde + Random(RangoHasta);
    Apilar(X);
  End;
  LlenarClavesRandom := OK;
End;

// retorno la Posicion del Tope
Function Pila.Tope():PosicionPila;
Begin
  Tope := Inicio;
End;

Function Pila.CantidadElementos(): LongInt;
Begin
  CantidadElementos := Q_Items;
End;

Function Pila.DatoDeLaClave: TipoDatosClave;
Begin
  DatoDeLaClave := TDatoDeLaClave;
End;

Function Pila.SizeStack(): LongInt;
Begin
  SizeStack := Size;
End;

Function Pila.MaxSizeStack(): LongInt;
Begin
  MaxSizeStack := MAX;
End;

end.
```