Programación I

Estructura de datos VECTOR

TEMAS de la CLASE

1. Operaciones frecuentes en el tipo Vector
   - Cargar elementos
   - Agregar un elemento al final
   - Insertar un elemento en una posición determinada
   - Borrar un elemento de una posición determinada

2. Ejercitación

Tipo Vector: Dimensiones Física y lógica

Cuando trabajamos con vectores se deben considerar:

Dimensión Física del vector

Se especifica en el momento de la declaración y determina su ocupación de memoria. La cantidad de memoria no variará durante la ejecución del programa.

Dimensión Lógica del vector

Se determina cuando se cargan elementos al vector. Indica la cantidad de posiciones de memoria ocupadas con contenidos cargados desde la posición inicial en forma consecutiva.

Tipo Vector: Carga de datos

La operación de Cargar datos en un vector consiste en incorporar un elemento a continuación del otro desde la posición inicial en forma consecutiva.

Esta operación debe controlar que la cantidad de elementos que se cargan no supere la dimensión física.

CONSIDERACIONES

- **dimL** : cantidad de elementos en el vector (dimensión lógica).
- **dimF** : dimensión física del vector (tamaño especificado en la declaración del vector)
- **Espacio suficiente**: Debe verificarse que dimL < dimF (dimensión lógica < dimensión física)
- **Condición de corte:** El ingreso de datos debe tener un fin

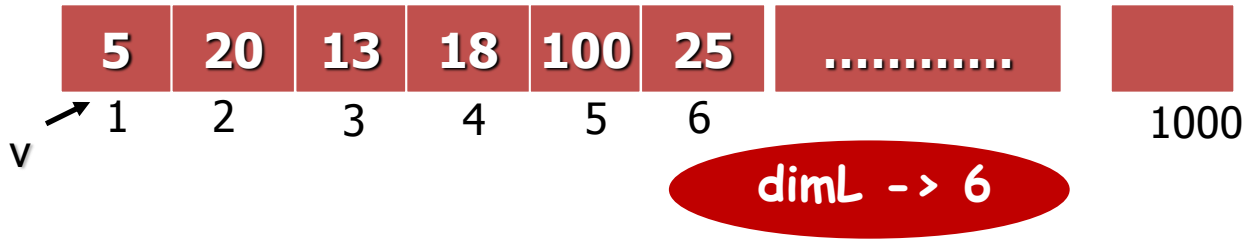
Tipo Vector: Carga de datos

Implementar un módulo que lea números enteros y los cargue en un vector de a lo sumo 1000 elementos. La lectura finaliza con el valor 99:

```
Const dimF = 1000;
Type vector =  Array [ 1..dimF] of integer;
Procedure CARGAR ( var v: vector; var dimL: integer );
 var dato: integer;
 begin
   dimL := 0;
   read (dato);
   while (dato <> 99)  and ( dimL < dimF ) do begin
          dimL := dimL + 1;
          v [dimL] := dato;
          read (dato);
   end;
End;
```

¿Qué retorna el procedimiento?

Si se leen los valores: 5, 20, 13, 18, 100, 25 y 99, obtendremos:



Tipo Vector: Agregar un elemento al final

La operación de Agregar un elemento en un vector consiste en incorporar el elemento a continuación del último ingresado, es decir, en la posición siguiente a la indicada en la dimensión lógica.

Esta operación debe verificar que haya lugar en la estructura, es decir que la dimensión lógica sea menor que la dimensión física.

CONSIDERACIONES

- **dimL** : cantidad de elementos en el vector (dimensión lógica).
- **dimF** : dimensión física del vector (tamaño especificado en la declaración del vector)
- **Espacio suficiente**: Debe verificarse que dimL < dimF (dimensión lógica < dimensión física)
- El elemento a **agregar** ocupa la **posición dimL+1**.
- Luego de la operación la **cantidad** de elementos es **dimL+1**.

Tipo Vector: Agregar un elemento al final

```
Const
    dimF = 1000;
Type
   vector =  Array [ 1..dimF] of integer;
```

```
Procedure AGREGAR (var v: vector; var dimL: integer;
                   elemento: integer; var exito: boolean);
Begin
 exito:= false;
 {verificar espacio suficiente}
 If (dimL < dimF) then begin
                    exito:= true;
                    dimL:= dimL+1; {actualizar cantidad de elementos}
                    v [dimL]:= elemento;
                   end;
end;
```

Tipo Vector: Insertar un elemento

La operación de Insertar un elemento en un vector consiste en incorporar el elemento en una posición determinada o de acuerdo a un orden impuesto a sus datos.

Esta operación también tiene que verificar que haya lugar en la estructura, es decir que la dimensión lógica sea menor que la dimensión física.

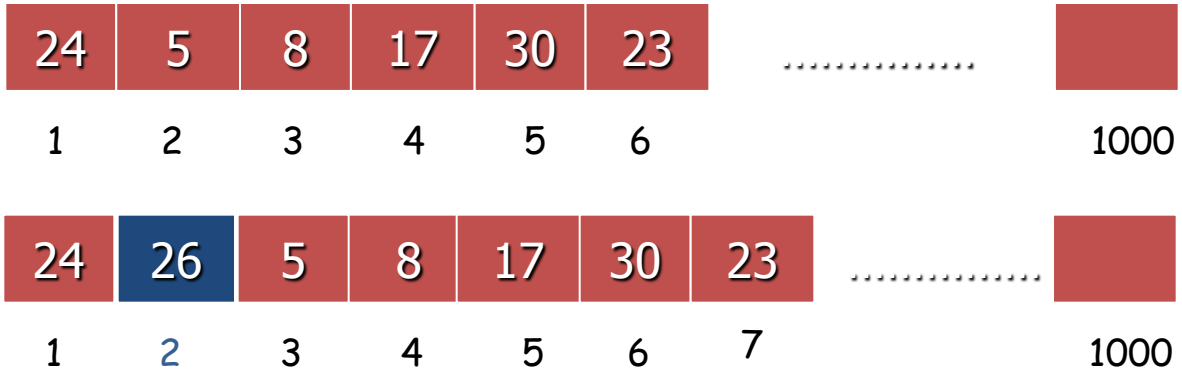
CONSIDERACIONES

- **dimL** : cantidad de elementos en el vector (dimensión lógica).
- **dimF** : dimensión física del vector (tamaño especificado en la declaración del vector)
- **Espacio suficiente**: Debe verificarse que dimL < dimF (dimensión lógica < dimensión física)
- Luego de la operación la **cantidad** de elementos es **dimL+1**.

Tipo Vector: Insertar un elemento

Cuando se requiere insertar un elemento en un vector se presentan dos posibilidades distintas:

1 Insertar un elemento en una posición determinada



2 Insertar un elemento manteniendo un orden predeterminado

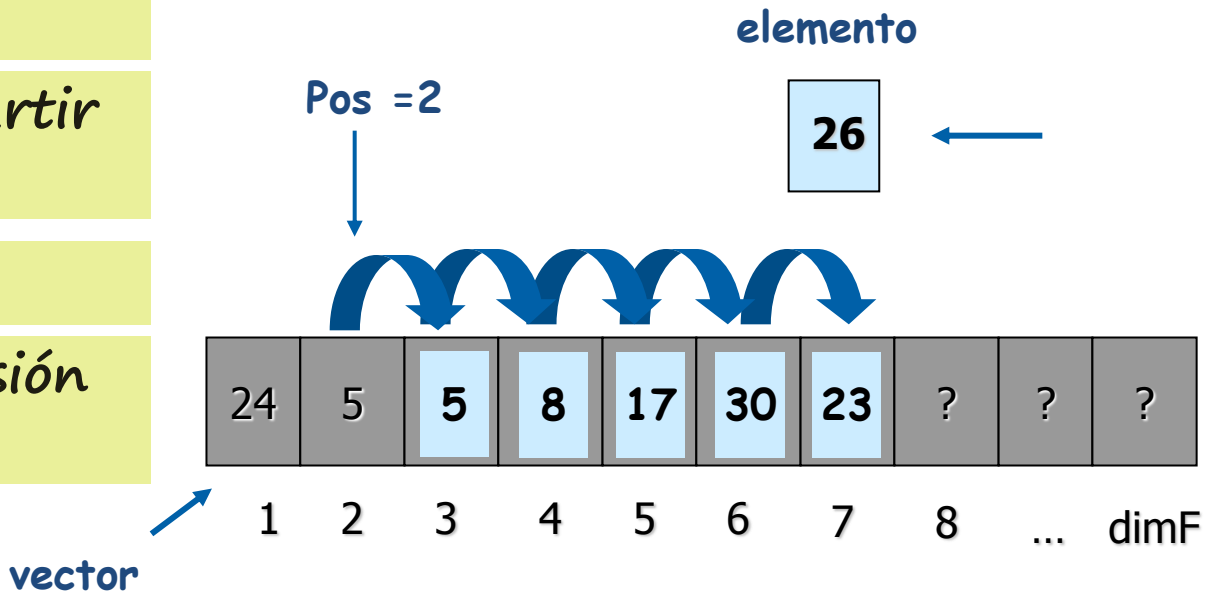
Lo veremos más adelante

Tipo Vector: Insertar un elemento en una posición determinada

➡ Esta operación también tiene que verificar que la posición sea válida.

1. Verificar la posición a insertar
2. Verificar espacio en el vector
3. Abrir el vector (a partir de la dimensión lógica)
4. Asignar el valor
5. Aumentar la dimensión lógica

Supongamos que se quiere insertar el valor 26 en la posición 2 de un vector de valores enteros…



Dimensión lógica = 6 → Dimensión lógica = 7

Tipo Vector: Insertar un elemento en una posición determinada

1. Verificar la posición a insertar
2. Verificar espacio en el vector
3. Abrir el vector (a partir de la dimensión lógica)
4. Asignar el valor
5. Aumentar la dimensión lógica

Parámetros de la operación:

- ✓v: vector a trabajar
- ✓dimL: cantidad de elementos
- ✓elemento: dato a insertar
- ✓pos: posición donde insertar
- ✓exito: resultado operación

```
Procedure INSERTARPOS (var v:vector; var dimL: integer;
  elemento: integer; pos: integer; var exito: boolean);

var i : integer;
Begin
 exito:= false;
 if (dimL < dimF) and ((pos>=1) and (pos<= dimL))
     then begin
            exito:= true;
            for  i:= dimL downto pos do
              v [ i + 1 ] := v [ i ] ;
            v [pos] := elemento;
            dimL := dimL + 1;
          end;
end;
```

Verificar espacio y posición válida

Abrir el arreglo

Asignar el valor

Aumentar la dimensión

Tipo Vector: Borrar un elemento

La operación de Borrar un elemento en un vector consiste en eliminar un elemento determinado o bien eliminar un elemento que ocupa una posición determinada.
En el caso de borrar un elemento en una posición determinada, se debe verificar que la posición sea válida. En el caso de eliminar un elemento determinado, hay que verificar que exista dicho elemento.

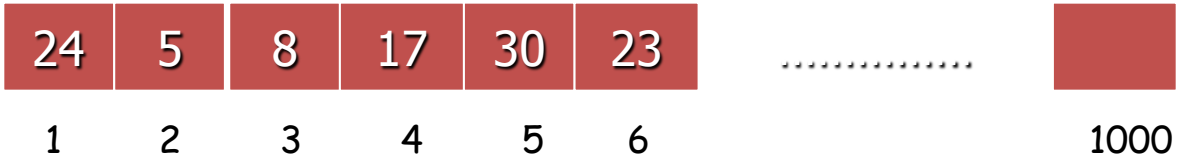
CONSIDERACIONES

- **dimL** : cantidad de elementos en el vector (dimensión lógica).
- Luego de la operación la **cantidad** de elementos es **dimL-1**.

Tipo Vector: Borrar un elemento

Cuando se requiere borrar un elemento en un vector se presentan dos posibilidades distintas:

1 Borrar un elemento de una posición determinada



Posición 2

2 Borrar un elemento determinado del vector

Lo veremos más adelante

Tipo Vector: Borrar un elemento de una posición determinada

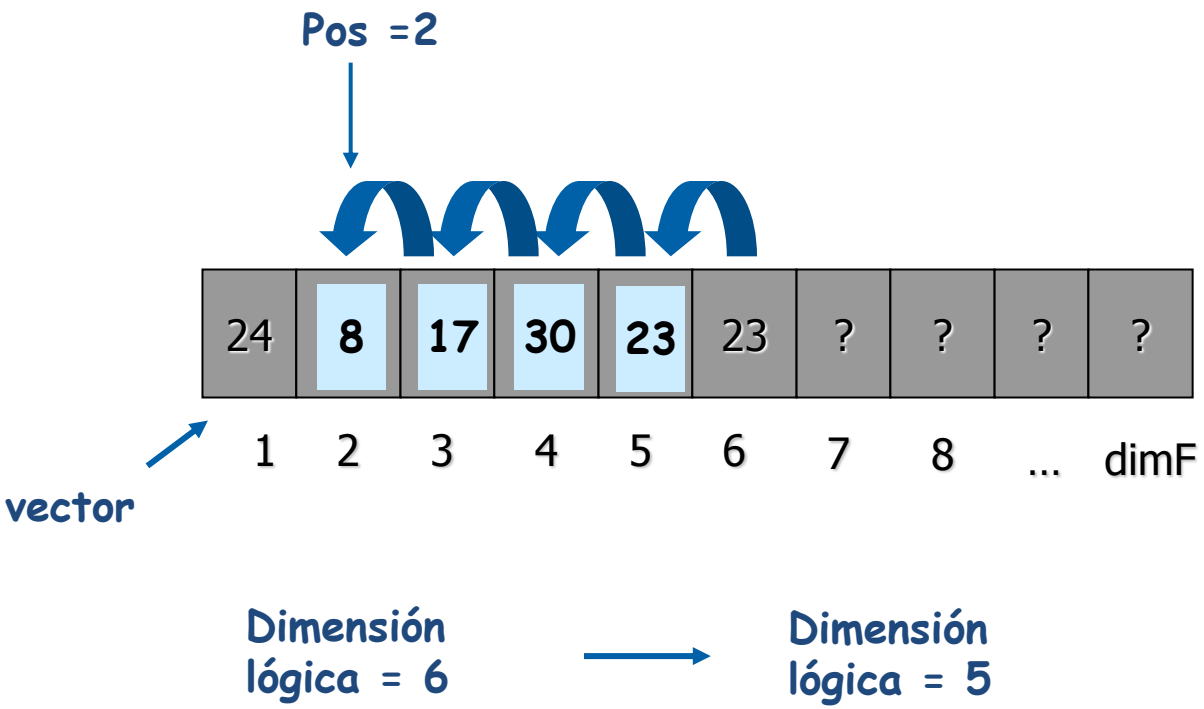
➡ Esta operación también tiene que verificar que la posición sea válida.

1. Validar la posición a borrar

2. Desplazar elementos (a partir de la posición siguiente)

3. Disminuir la dimensión lógica

Supongamos que se quiere borrar el elemento de la posición 2 de un vector de valores enteros…



Tipo Vector: Borrar un elemento de una posición determinada

1. Validar la posición a borrar

2. Desplazar elementos (a partir de la posición siguiente)

3. Disminuir la dimensión lógica

Parámetros de la operación:

- ✓v: vector a trabajar
- ✓dimL: cantidad de elementos
- ✓pos: posición a borrar
- ✓exito: resultado operación

```
Procedure BorrarPos (var v: vector;
                     var dimL: integer; pos: posicion;
                     var exito: boolean );
 var i: integer;
 begin
  exito := false;          Validar la posición a borrar
  if (pos >=1 and pos <= dimL)
     then begin
            exito := true;
            for i:= pos + 1  to   dimL  do
                v [ i - 1 ]  :=  v [ i ] ;
            dimL := dimL - 1 ;  Disminuir la dimensión lógica
          end;
 end;
```

Desplazar elementos

Ejercitación

1. Se lee una sucesión nombres y notas de alumnos que finaliza con nombre “Fulano”. Informar los nombres y notas de los alumnos que superan el promedio del grupo.

Nota: Como máximo se pueden ingresar 100 alumnos.

- **Leer, Guardar y Sumar todas las notas**
- **Calcular promedio**
- **Recorrer y Comparar cada nota con el promedio**

Leer, Guardar y Sumar todas las notas

```
Inicializar suma de notas
Leer datos de alumno
Mientras (haya lugar) y (no ingrese “Fulano” )
    guardar en el vector
    actualizar suma de notas
    leer datos de alumno
```

Recorrer y Comparar cada nota con el promedio

```
Repetir cantidad alumnos guardada
   acceder a los datos del alumno
   si nota > promedio entonces
                informar nombre
```

3. Se lee una sucesión nombres y notas de alumnos que finaliza con nombre “Fulano”. Informar los nombres y notas de los alumnos que superan el promedio del grupo.

Nota: como máximo se pueden ingresar 100 alumnos.

- *Leer, Guardar y Sumar todas las notas*
- *Calcular promedio*
- *Recorrer y Comparar cada nota con el promedio*

```
Program ejemplo2;

const dimF= 100; {dimensión física}
type
   str20 := string [20];
   alumno = record
              nombre: str20;
              nota: real;
            end;
   rango = 0.. dimF;
   vectorDatos = array [1..dimF] of alumno;

{implementación LeerGuardarSumar}
{implementación RecorreryComparar}

var
   datos: vectorDatos; dimL: rango;
   suma, promedio : Real;

Begin
  LeerGuardarSumar(datos, dimL, suma);
  if dim <>0
     then begin
            promedio := suma/dimL;
            RecorreryComparar(datos, dimL, promedio)
          end;
End.
```

Leer, Guardar y Sumar todas las notas

Inicializar suma de notas

Leer datos de alumno

Mientras (haya lugar) y (no ingrese "Fulano")

- *guardar en el vector*
- *actualizar suma de notas*
- *leer datos de alumno*

```
const dimF= 100; {dimensión física}
type
 str20 := string [20];
 alumno = record
             nombre: str20;
             nota: real;
          end;
 rango = 0.. dimF;
 vectorDatos=array[1.. dimF]of alumno;
```

```
procedure LeerGuardarSumar
      (var dat:vectorDatos; var dimL: rango;
        var sum: Real);

 Procedure leerAlumno (var alu:alumno);
 begin
  read (alu.nombre);
  if (alu.nombre <> ´Fulano´)
      then read (alu.nota);
 end;

var
   a : alumno;

begin
 sum := 0;
 dimL := 0; {dimensión  lógica}
 leerAlumno (a);
 while (dimL < dimF) and (a.nombre <> ´Fulano´)
   do begin
     dimL := dimL + 1;
     dat[dimL]:= a;
     sum := sum + dat[dimL].nota;
     leerAlumno (a);
   end
end;
```

Recorrer y Comparar cada nota con el promedio

Repetir cantidad alumnos guardada
  acceder a los datos del alumno
  si nota > promedio entonces
    informar nombre

```
const dimF = 100; {dimensión física}
type
 str20 := string [20];
 alumno = record
            nombre: str20;
            nota: real;
          end;
 rango = 0.. dimF;
 vectorDatos=array[1.. dimF]of alumno;
```

```
Procedure RecorreryComparar  (dat: vectorDatos; dimL:rango; prom: real);
 var i: integer;
begin
  for i := 1 to dimL do
       if (dat[i].nota > prom) then
          Writeln (dat[i].nombre, ´ ´, dat[i].nota );
End.
```

ESTRUCTURA DE DATOS VECTOR

Un centro de deportes nos ha encargado un sistema para el manejo de sus clientes. En el centro se ofrecen 5 actividades distintas. De cada cliente se conoce código de cliente, DNI, apellido, nombre, edad, el número de actividad que realiza, mes y año del último pago. La lectura finaliza cuando llega el DNI 0.

Se pide:

a) Almacenar el precio mensual de las actividades a través de la lectura de os valores.
b) Generar la estructura de datos que almacene los clientes del centro de deportes (como máximo 500), leyendo los datos de los clientes hasta ingresar un DNI -1.
c) Informar la cantidad de clientes que no tienen la cuota al día.
d) Eliminar de la estructura el cliente de una posición que se lee.