

# Programación Lógica II

<b>INSTITUCIONALES .....</b>	1
<b>¿Como esta organizado este texto? .....</b>	4
<b>Introducción .....</b>	5
<b>Esquema .....</b>	6
<b>Situación profesional 1: Librería Cabildo .....</b>	7
<b>SP1 / H1: Registros y Agregados de Datos .....</b>	8
<b>SP1 / Autoevaluación 1 .....</b>	42
<b>SP1 / Ejercicio resuelto .....</b>	45
<b>SP1 / Ejercicio por resolver .....</b>	53
<b>SP1 / Evaluación de paso .....</b>	56
<b>Situación profesional 2: Frigorífico CARCOR S.A. .....</b>	59
<b>SP2 / H1:Cortes de control y reportes .....</b>	61
<b>SP2 / Autoevaluación 1 .....</b>	86
<b>SP2 / Ejercicio resuelto .....</b>	89
<b>SP2 / Ejercicio por resolver .....</b>	93
<b>SP2 / Evaluación de paso .....</b>	94
<b>Situación profesional 3: Biblioteca CÓRDOBA .....</b>	96
<b>SP3 / H1:Procesamiento de múltiples tablas relacionadas. Modelo de tablas relacionales .....</b>	100
<b>SP3 / Autoevaluación 1 .....</b>	109
<b>SP3 / H2:Integración algorítmica de vectores y matrices .....</b>	111
<b>SP3 / Autoevaluación 2 .....</b>	115
<b>SP3 / Ejercicio resuelto .....</b>	117
<b>SP3 / Ejercicio por resolver .....</b>	123
<b>REFERENCIAS 1 .....</b>	126
<b>SP3 / Evaluación de paso .....</b>	127
<b>Situación profesional 4: Empleos S.A. .....</b>	129
<b>SP4 / H1:Actualización de archivos .....</b>	130
<b>SP4 / Autoevaluación 1 .....</b>	151
<b>SP4 / H2:Organización de archivos .....</b>	153
<b>SP4 / Autoevaluación 2 .....</b>	158
<b>SP4 / Ejercicio resuelto .....</b>	161

SP4 / Ejercicio por resolver .....	167
SP4 / Evaluación de paso .....	168
<b>Cierre .....</b>	<b>170</b>
<b>Bibliografía .....</b>	<b>171</b>

## AGRADECIMIENTOS

Agradecemos a todos aquellos que han aportado su investigación, su experiencia y su tiempo para la elaboración de este TID.

---

## AUTORES



Pablo Javier Romo

- Analista de Sistemas de Computación, egresado del Colegio Universitario IES.
  - Preprogramador de aplicaciones visuales en la empresa CDSI Argentina S.A.
  - Docente del Colegio Universitario IES en las materias: *Estructura de Datos, Matemática Discreta, Programación Lógica I y Programación Lógica II.*
- 

## EQUIPO DE PRODUCCIÓN

### Producción y dirección general

- Director general: Alberto Rabbat
- Directora Académica: María Fernanda Sin
- Vicedirectora Académica: María Teresa de las Casas

### Planificación y coordinación general

- Coordinadora de estudios a distancia: Érica Bongiovanni

### Producción Multimedial

- Sebastian Benito
- Nicolás Irusta
- Sabrina Monteverde
- Facundo Moreno
- Diego Oliva

## Producción Académica

- Ana Giró
- Telmo Torres
- Paula Gamba

## Coordinación de sistemas

- Marcela Giraudo
- Marco Moretti

## Diseño y Desarrollo

- M. Rosario Figueroa
  - G. Alejandro Zabala
- 

## USO DE MARCAS

Cláusulas de uso de marcas y derechos de autor de terceros.

A. Uso atípico de marca ajena: Exclusión de los usos no comerciales del Derecho

Marcario.PERSPECTIVAS S.A. en su carácter de titular de los derechos intelectuales sobre la presente obra declara por esta vía que el uso que realiza de marcas comerciales de terceros lo es sólo a los fines informativos y didácticos, para mejor comprensión de los lectores y alumnos del contenido de la obra, siendo el mismo de carácter atípico (uso atípico de marca ajena) y lícito. Este uso, a tenor de la jurisprudencia vigente queda fuera del ius prohibendi, que detenta el titular de cada marca registrada, atento no ser el mismo de carácter comercial en relación al producto que distinguen las referidas marcas, y por ende de índole marcario.-

B. Uso de derechos de autor en videos, diskettes, imágenes y audio: Libre utilización -Uso privado- de obras protegidas. PERSPECTIVAS S.A. en su carácter de titular de los derechos intelectuales sobre la presente obra declara por esta vía que el uso que realiza de determinadas grabaciones (audio y video), e imágenes (fotografías) de terceros, lo es a los fines informativos y didácticos, para mejor comprensión de los alumnos del contenido de la obra, siendo el mismo de carácter privado y no comercial, y desde ya respetando el derecho de cita, esto es declarando en toda ocasión la cita o fuente (obra y autor) de la cual se toman los fragmentos de obras de terceros para incorporarlos a la presente (Convenio de Berna, Acta de París, 1971 – Art. 10, § 2 y § 3).-

C. Modificación de obras literarias por el titular de los derechos patrimoniales. PERSPECTIVAS S.A. en su carácter de titular de los derechos intelectuales (patrimoniales) sobre la presente obra, aclara que ha autorizado a Pablo ROMO, a su modificación respecto de la obra original, publicada por Bongiovanni, Erica; Farneti, Alejandra; Obregón, Diego en abril 2007, bajo N° de ISBN: 978-987-600-062-8, constituyéndose estos últimos en autores morales de la obra referida y modificada. Se declara a todo efecto, que los derechos intelectuales se ceden y mantienen a favor de su titular PERSPECTIVAS S.A.

---

## COPYRIGHT

Pablo ROMO.

**Programación Lógica 2** / Pablo ROMO; coordinado por María Teresa de las Casas; dirigido por José Alberto Rabbat. - 1a ed. - Córdoba : IES Siglo 21, 2012.

CD-ROM.

ISBN 978-987-600-185-4

1. Programación. I. Casas, María Teresa de las, coord. II. José Alberto Rabbat, dir. III. Título

CDD 005.3

1er Edición

© 2012 - Editorial IES Siglo 21

Buenos Aires 563

TE: 54-351-4211717

5000 - Córdoba

Queda hecho el depósito que establece la Ley 11723

Libro de edición argentina

No se permite la reproducción parcial o total, el almacenamiento, el alquiler, la transmisión o la transformación de este libro, en cualquier forma o por cualquier medio, sea electrónico o mecánico, mediante fotocopias, digitalización u otros métodos, sin el permiso previo y escrito del editor. Su infracción está penada por las leyes 11723 y 25446.

Se terminó de replicar durante el mes de febrero de 2012 en el departamento de Logística en Editorial IES Siglo 21.



# ¿Cómo está organizado este texto?

Usted está en presencia de este texto que los autores proponen para la comprensión y estudio de la asignatura. Ha sido preparado y diseñado para facilitarle el acceso al conocimiento, a partir de una secuencia uniforme cuyo punto de partida es la práctica profesional cotidiana y no la teoría alejada de la realidad.

Está organizado de la siguiente manera:

- **Introducción.** Indica qué papel desempeña la asignatura dentro de la carrera y los conceptos básicos que usted conocerá.
- **Esquema.** Muestra los enlaces que unen los conceptos centrales de la asignatura entre sí.
- **Situación profesional.** Lo ubica frente a un problema de la práctica profesional cotidiana que puede ser resuelto, ya que existe al menos una solución para ello, a través de conocimientos específicos que en cada caso se aportan.
- **Herramientas.** Son los conocimientos necesarios para resolver la situación profesional planteada.
- **Autoevaluación.** Para que usted compruebe si ha comprendido correctamente lo que se explicó en una herramienta, los autores proponen la resolución de actividades y le ofrecen las respuestas.
- **Ejercicio resuelto.** Bajo este título encontrará una manera de resolver los problemas de práctica profesional planteados, con la selección de las herramientas pertinentes.
- **Ejercicio por resolver.** Ahora le toca a usted. Es el momento de aplicar las herramientas a una Situación profesional nueva o similar a la ya expuesta. Todas las dudas que le aparezcan podrán ser planteadas a su docente/tutor.
- **Evaluación de paso.** Para que usted compruebe si ha comprendido correctamente lo que se explicó en las distintas herramientas que hasta el momento se han presentado, los autores proponen la resolución de actividades y le ofrecen las respuestas.
- **Bibliografía.** Se indican los textos, revistas y links de consulta a los que podrá recurrir para complementar o ampliar algunos temas.

# Introducción

Hasta el momento usted ha dado los primeros pasos en el mundo de la programación. Ha incorporado el conocimiento necesario para realizar el análisis de una situación determinada y en base a ello ha podido, a través de la lógica computacional, ofrecer soluciones prácticas y creativas ante distintas y variadas situaciones profesionales.

En esta materia dará un paso más en su formación como profesional informático. Incorporará el conocimiento necesario para que los datos generados por sus proyectos perduren en el tiempo. Trabajaremos sobre el manejo y mantenimiento de archivos, pudiendo consultar, agregar, eliminar y modificar registros correspondientes a cualquier tipo de información que desee manejar. Tendrá la habilidad de generar distintos tipos de reportes impresos o por pantalla en base a uno o más archivos. Incorporará métodos avanzados de acceso directo a registros. Y por último, se le brindarán herramientas necesarias para el ordenamiento de los registros contenidos en un archivo.

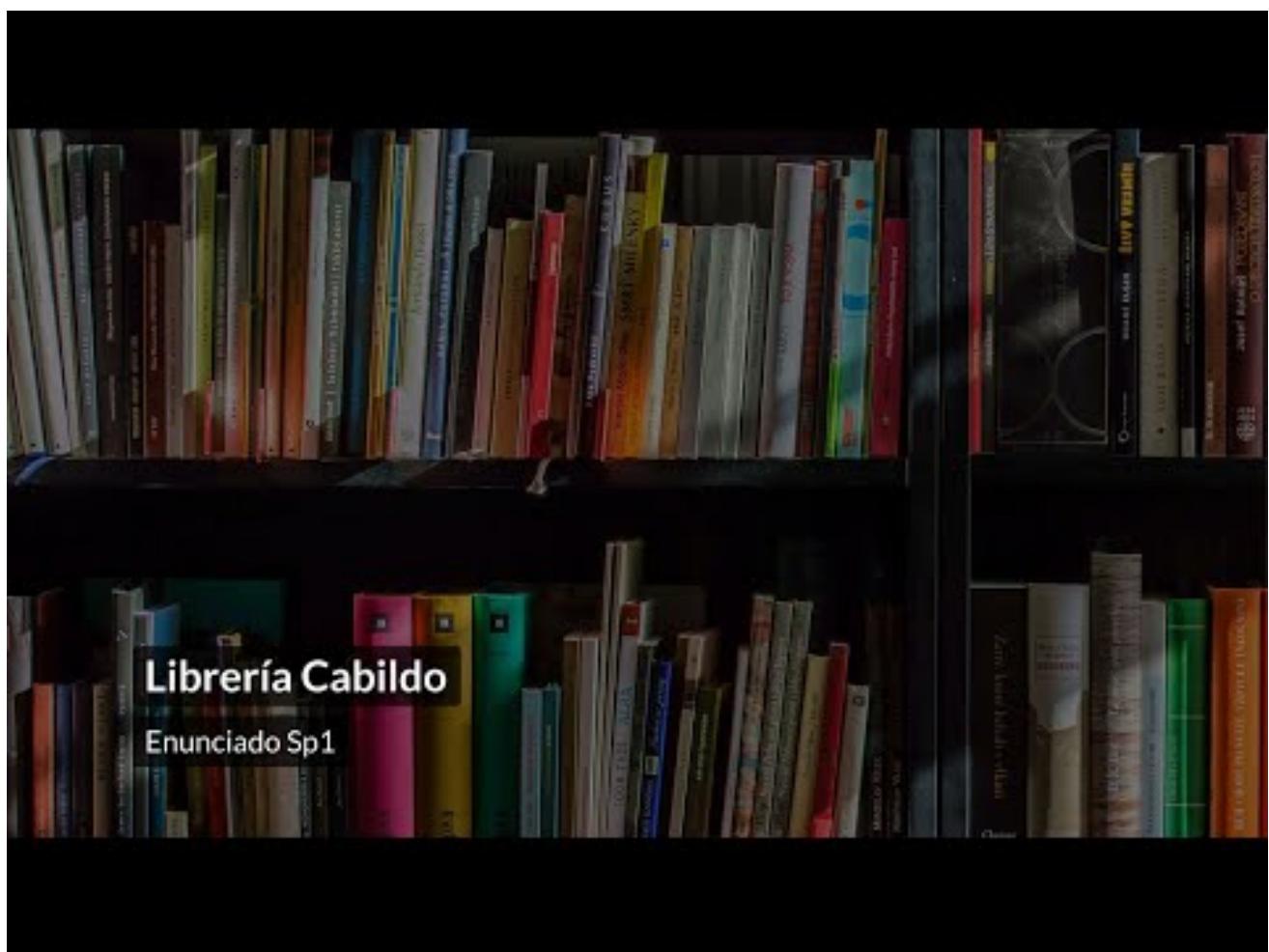
Con estas herramientas, usted estará en condiciones de diseñar todo tipo de sistemas informáticos, ya que contará con la habilidad, práctica y conocimientos necesarios para plantear una solución lógica aprovechando lo que el avance tecnológico nos ofrece por estos días.

Es hora de los hechos así que comencemos y vayamos hacia nuestro objetivo.

**El autor**



# Situación profesional 1: Librería Cabildo



Video "Enunciado SP1" | Elaboración Propia. DEPROE - Colegio Universitario IES

La "Librería Cabildo", que cuenta con un sistema de facturación, le solicita a usted que diseñe una interfaz y el programa que le posibilite la obtención de los siguientes listados:

- Listado de las cuentas corrientes que muestre: el número de la cuenta, el nombre del cliente y el saldo.
- Listado de cuentas que superen los \$1.000 de saldo, donde deberá imprimirse el número de la cuenta, el nombre del cliente y el saldo.
- Listado de cuentas "deudoras", en el que aparecerá impreso el número del cliente, el nombre y el saldo. Al final del listado deberá mostrarse el total adeudado a la empresa.
- Informe de Saldo Total de las cuentas y el monto promedio de las mismas.

# SP1 / H1: Registros y Agregados de Datos

Presentaremos esta herramienta de la siguiente manera:

- a) Registros
- b) Vector de registros
- c) Agregados de Datos (AD)

## a) Registros

En materias anteriores, hemos visto ejemplos de programas que requieren el empleo de datos simples, de vectores y matrices, pero ninguna de estas estructuras es suficiente para resolver lo planteado en la Situación profesional.

En caso que se vea en la necesidad de almacenar y procesar datos tan disímiles como: Número de Cuenta, Nombre del cliente y Saldo actualizado de la cuenta, trabajar con una matriz es imposible. Fundamentalmente porque los valores a almacenar corresponden a tipos de datos diferentes:

- **Número de Cuenta:** Tipo numérico.
- **Nombre de Cliente:** Tipo Alfanumérico o Cadena de Caracteres.
- **Saldo Actualizado:** Tipo Numérico.

En situaciones como esta hace falta utilizar una estructura de datos que permita almacenar valores de diferente tipo. Esta estructura es la que se denomina "registro".

Para almacenar los datos mencionados en el ejemplo se puede definir un registro con tres campos. En cada campo se almacena un atributo diferente:

- **Número de cuenta**
- **Nombre del cliente**
- **Saldo actualizado**

Un programador puede imaginarse un registro del siguiente modo:



Registro

1254	GONZALES, María Inés	1.258,30
------	----------------------	----------

Campo

1254	GONZALES, María Inés	1.258,30
------	----------------------	----------

Variable

1254	GONZALES, María Inés	1.258,30
------	----------------------	----------

Un registro es un conjunto de datos que puede ser heterogéneo. Cada elemento del registro se denomina campo y puede ser un dato simple (variable) o estructurado (vector, registro).

Cuando decimos que el registro puede llegar a ser heterogéneo nos referimos a que algunos lo son y otros no, a diferencia de los vectores que nunca pueden ser heterogéneos.

**NOTA:** solamente veremos registros cuyos campos tengan datos simples.

Cuando se utiliza un registro en un programa debe definirse previamente. Para la definición de un registro se debe tener en cuenta que es necesario asignarle un Nombre que lo identifique, además de especificar el nombre y Tipo de Dato de cada uno de sus datos miembro, o campos.

**Nombre:** se le asigna un nombre al registro y también a cada uno de los campos. Siempre conviene que los nombres sean significativos.

Observe en el siguiente ejemplo:

El registro se denomina:  
**CuentasCorrientes.**

### Registro **CuentasCorrientes**

1254	GONZALES, María Inés	1.258,30
------	----------------------	----------

**Código**

**Nombre**

**Saldo**

El campo que almacena el número de cuenta se denomina: **Código.**

El campo que almacena el nombre del cliente se denomina: **Nombre.**

El campo que almacena el saldo actualizado se denomina: **Saldo.**

Tipo de dato: cada campo del registro tiene un tipo de dato determinado. Este puede ser: numérico, de cadena (o alfanumérico), lógico, de fecha o de hora.

Observe el siguiente ejemplo de definición de un registro:

**Registro Cliente**

**Campo Código Tipo Numérico**

**Campo Nombre Tipo Cadena**

**Campo Saldo Tipo Numérico**

**Fin Registro**

### Implementación de Variables Estructuradas

Debemos tener en cuenta que la Definición de un Registro (tal como la anterior) define un "Tipo de Dato" nuevo, que no se compone de un dato simple, sino de varios agrupados, cada uno con su propio tipo de dato y nombre. En otras palabras: definir un registro es como crear el "molde" para fabricar con él instancias reales de datos. El registro no puede, per se, contener información; sino que permite "construir una variable" con dicha estructura, y almacenar en esa variable el respectivo valor para cada uno de sus campos. De este modo, podemos crear una variable con el Tipo del Registro definido, así:

**Variable MiCliente Tipo Registro Cliente**

Es importante remarcar que las Variables de Tipo Registro (también llamadas Variables Estructuradas) no tienen dimensiones como las matrices o los vectores. Para acceder a cada elemento se debe colocar el nombre del registro, un punto y el nombre del campo.

Cliente.Codigo #Es el valor almacenado en el campo "Código"

Cliente.Nombre #Es el valor almacenado en el campo "Nombre"

Cliente.Saldo #Es el valor almacenado en el campo "Saldo"

Entonces, para almacenar datos para un cliente en particular con la estructura indicada, la sintaxis es la siguiente:

```
MiCliente.Código = 123
```

```
MiCliente.Nombre = "Juan Pablo"
```

```
MiCliente.Saldo = 500
```

En general, para acceder a un determinado campo de una variable estructurada, la sintaxis usada es la siguiente:

```
<NombreVariableEstructurada>.<NombreCampo> = Valor
```

#### Ejemplo: Registro de Clientes

Suponga que se le solicita un programa que almacene los datos de un cliente. El programa trabajará con la siguiente interfaz gráfica:

```
Programa CargaDatos
    #Definición de Datos Globales
    Registro Cliente
        Campo Código Tipo Numérico
        Campo Nombre Tipo Cadena
        Campo Saldo Tipo Numérico
    Fin Registro

    Variable Micliente Tipo Registro Cliente

    #Definición del Formulario
    Formulario frmCargarDatos
        Marco mrcCliente
            Caja de texto txtNombre
        Fin Marco

        Marco mrcCuenta
            Caja de texto txtCodigo
            Caja de texto txtSaldo
        Fin Marco

        Botón de Comando cmdIngresar
    Fin Formulario

    #Desarrollo de los Procedimientos de Evento
    Procedimiento cmdCargar:Click
        #Asigna el valor de la caja de texto al campo
        MiCliente.Nombre = txtNombre
        MiCliente.Código = txtCodigo
        MiCliente.Saldo = txtSaldo
    Fin Procedimiento

Fin Programa
```

#### PSEUDO: Registro de clientes

```
Programa CargaDatos
    #Definición de Datos Globales
    Registro Cliente
        Campo Código Tipo Numérico
        Campo Nombre Tipo Cadena
```

```

    Campo Saldo Tipo Numérico
Fin Registro

Variable MiCliente Tipo Registro Cliente

#Definición del Formulario
Formulario frmCargarDatos
    Marco mrcCliente
        Caja de texto txtNombre
    Fin Marco
    Marco mrcCuenta
        Caja de texto txtCodigo
        Caja de texto txtSaldo
    Fin Marco
    Botón de Comando cmdIngresar
Fin Formulario

#Desarrollo de los Procedimientos de Evento
Procedimiento cmdCargar:Click
    #Asigna el valor de la caja de texto al campo
    MiCliente.Nombre = txtNombre
    MiCliente.Codigo = txtCodigo
    MiCliente.Saldo = txtSaldo
Fin Procedimiento
Fin Programa

```

Si nos remitimos a la Situación profesional planteada, vemos que el registro que utilizamos es el correspondiente a la cuenta corriente de un cliente de la librería Cabildo y se definiría de la siguiente manera:

#### # Definición del Registro

##### Registro Cuenta

Variable Numérica CodCta

Variable Alfanumérica Nombre

Variable Numérica Saldo

Fin Registro

## b) Vector de Registros

Habrá situaciones en donde necesite almacenar datos de registros temporalmente para realizar operaciones y obtener resultados. En estas situaciones es conveniente declarar un vector de registros que le permitirá tener en memoria los datos que necesita hasta tanto dejen de ser útiles.

#### Ejemplo: Listado de Clientes

Tomemos el caso en el que se cuenta con un vector precargado de clientes, y deseamos emitir un listado de los mismos. Tendremos un programa como el siguiente:

```
Programa MostrarDatos
    # Definición de Datos Globales
    Registro Cliente
        Campo Codigo Tipo Numérico
        Campo Nombre Tipo Cadena
        Campo Saldo Tipo Numérico
    Fin Registro
```

**Vector Clientes(10) Tipo Registro Cliente**

```
# Desarrollo de los Procedimientos
Procedimiento MostrarDatos
```

**Listado de clientes**

Video "[Ejemplo: Listado de clientes](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Observe que se declara un registro llamado Cliente, con 3 variables.

A continuación, procedemos a la declaración de un vector de clientes, que en cada posición almacenará un cliente diferente.

# Por último, daremos formato al listado y con una estructura repetitiva mostraremos el contenido del vector de clientes.

En nuestra Situación profesional no haremos uso de un vector de registros ya que para la resolución de la misma utilizaremos un AD.

### c) Agregado de Datos. Casos de uso y características.

Es importante tener en cuenta que un registro permite almacenar la información de una sola entidad. Para procesar grandes volúmenes de información es necesario trabajar con conjuntos de registros. Existe un modo de procesar conjuntos de registros idénticos, y consiste en grabarlos en una "Tabla" o "Agregado de Datos".

1254	GONZALEZ, María Inés	1.258,30
1356	GOMEZ, Lucas	0,00
1458	MARTINEZ, José	125,54
1490	CHIARAMELLO, Jorge	8.452,30
1510	MICOLINI, Julieta	5.034,00
1567	ARGUELLO, Mirian	1.290,00
1689	BASUALDO, Patricia	30,24
1690	VELAZQUEZ, Enrique	109,24
1589	AGUIRRE, Marcos	0,00
1783	MALDONADO, Silvio	0,00

Tabla o agregado de datos

Video "[Tabla o Agregado de datos](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Una "Tabla" o "Agregado de Datos" es un conjunto homogéneo de registros. Esto significa que todos los registros de un agregado tienen la misma estructura, es decir, están constituidos por los mismos campos. Lo que no significa que todos contengan la misma información.

Normalmente los Agregados de Datos se almacenan en Archivos u otro tipo de recurso de almacenamiento no volátil y de alta capacidad, a los efectos de poder registrar en ellos la información necesaria y poder recuperarla en el momento que sea requerido.

Para que un programa pueda procesar un agregado de datos es imprescindible indicar el nombre del archivo y el tipo de registro que lo constituye.

Observe el siguiente ejemplo:

Tenemos un agregado de datos de clientes compuesto por un conjunto de datos tipo registro con nombre Cliente.

#### Agregado de Datos Clientes Tipo Registro Cliente

El programador puede imaginarse el **Agregado de Datos** del siguiente modo:

## Agregado de Datos

PL2

Agregado de Datos Clientes > Campos > Registros Clientes

	Código	Nombre	Saldo
1	1254	GONZALEZ, María Inés	1.258,30
2	1356	GOMEZ, Lucas	0,00
3	1458	MARTINEZ, José Ignacio	125,54
4	1490	CHIARAMELLO, Jorge	8.452,30
5	1510	MICOLINI, Julieta	5.034,00
6	1567	ARGUELLO, Mirian	1.290,00
7	1589	AGUIRRE, Marcos	0,00
8	1689	BASUALDO, Patricia	30,24
9	1690	VELAZQUEZ, Enrique	109,24
10	1783	MALDONADO, Silvio	0,00

Agregado de Datos Clientes  
Tipo Registro Cliente

Agregado de Datos Clientes > Campos > Registros Clientes

	Código	Nombre	Saldo
1	1254	GONZALEZ, María Inés	1.258,30
2	1356	GOMEZ, Lucas	0,00
3	1458	MARTINEZ, José Ignacio	125,54
4	1490	CHIARAMELLO, Jorge	8.452,30
5	1510	MICOLINI, Julieta	5.034,00
6	1567	ARGUELLO, Mirian	1.290,00
7	1589	AGUIRRE, Marcos	0,00
8	1689	BASUALDO, Patricia	30,24
9	1690	VELAZQUEZ, Enrique	109,24
10	1783	MALDONADO, Silvio	0,00

Código	Nombre	Saldo
1	GONZALEZ, María Inés	1.258,30
2	GOMEZ, Lucas	0,00
3	MARTINEZ, José Ignacio	125,54
4	CHIARAMELLO, Jorge	8.452,30
5	MICOLINI, Julieta	5.034,00
6	ARGUELLO, Mirian	1.290,00
7	<b>AGUIRRE, Marcos</b>	<b>0,00</b>
8	BASUALDO, Patricia	30,24
9	VELAZQUEZ, Enrique	109,24
10	MALDONADO, Silvio	0,00

## 1. Acceso a Agregado de Datos

Seguramente usted se está preguntando cómo se hace para acceder a todos los registros del **Agregado de Datos**, por ejemplo en el caso de tener que imprimirllos.

Los agregados de datos generalmente son archivos grabados en disco y para acceder a la información almacenada en ellos es necesario realizar las siguientes operaciones:

- Apertura
- Lectura
- Cierre

Para comprender mejor la necesidad de utilizar estas operaciones le preguntamos a usted: si un compañero del curso le entrega un libro, ¿qué hace usted para estudiar el contenido?

Seguramente usted responderá que:

- Primero abre el libro.
- Después lee el contenido para comprenderlo y analizarlo.
- Por ultimo lo cierra.

El ordenador, para procesar los registros de un **Agregado de Datos** hace lo mismo que usted: abre, lee y cierra.

Veremos a continuación estas acciones en detalle:

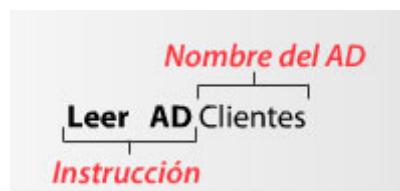
- **Apertura**

En el momento que se abre el archivo, la información que está almacenada, queda disponible para el ordenador. Para abrir el agregado de datos se utiliza la instrucción: "Abrir", que va seguida por el nombre del agregado de datos.



- **Lectura**

Por cada lectura que el ordenador realiza, la información almacenada en un registro pasa a memoria y se accede a ella utilizando el nombre del registro. Por cada lectura sólo se accede a un registro, por eso, para acceder a todos los registros se deberán realizar tantas lecturas como registros estén almacenados. Debe tener presente que si se realiza una lectura antes que la apertura se produce un error de programa. Para realizar la lectura se utiliza la instrucción: "leer", seguida por el nombre del Agregado de Datos.



- **Cierre**

En el momento que se cierra un agregado de datos deja de estar disponible la información almacenada en él. En caso que se intente una lectura posterior al cierre se produce un error de programa. Para realizar el cierre del agregado de datos se utiliza la instrucción "Cerrar", también seguida por el nombre del mismo.



La emisión de reportes es la más compleja de las actividades sobre Agregado de Datos por lo que su algoritmia resulta de interés para esta asignatura. Sobre el particular de las otras operaciones se avanzará en cada una de las materias específicas de programación.

En nuestra Situación profesional, como en cualquier otra en la que manejemos AD, haremos uso de estas operaciones para poder obtener el contenido de los registros que contenga.

## **2. Algoritmos de Recorrido**

Para acceder a un registro almacenado en un agregado de datos es necesario realizar una lectura, posterior a la apertura del mismo.

## Agregado de Datos Clientes

### Tipo Registro Cliente

Código	Nombre	Saldo
1254	GONZALEZ, María Inés	1.258,30
1356	GOMEZ, Lucas	0,00
1458	MARTINEZ, José Ignacio	125,54
1490	CHIARAMELLO, Jorge	8.452,30
1510	MICOLINI, Julieta	5.034,00
1567	ARGUELLO, Mirian	1.290,00
1589	AGUIRRE, Marcos	0,00
1689	BASUALDO, Patricia	30,24
1690	VELAZQUEZ, Enrique	109,24
Algoritmo de recorrido	ALDONADO, Silvio	0,00

**EOF**    ← *Final de archivo*

Video "[Algoritmo de recorrido](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Suponga que el siguiente es el agregado de datos con el que se trabaja:

Cuando se ejecuta la instrucción “Leer AD Clientes”, el registro “cliente”, que está memoria, tomará los siguientes valores:

Si se vuelve a ejecutar la instrucción “Leer AD Clientes”, el registro “cliente”, tomará los valores correspondientes al segundo registro del Agregado de Datos:

Y así sucesivamente... hasta llegar al registro que se necesita.

Remarquemos que: los agregados de datos se acceden en forma secuencial. Esto significa que no se puede leer el tercer registro o el último sin leer los anteriores previamente. Se debe comenzar leyendo el primero, luego el segundo, y así sucesivamente hasta llegar al registro que se necesita.

De ese modo se va recorriendo el agregado de datos, tal como se hace con los vectores. La diferencia reside en que el vector requiere de un índice que se incrementa de uno en uno, y los agregados de datos se recorren con sucesivas lecturas. Cada instrucción de lectura avanza un registro en el recorrido.

Cuando se han leído todos los registros del agregado de datos se llega al final del archivo, conocido como EOF: “End of file”. Esto significa que si se pretende realizar un listado de todos los registros almacenados, se deberá leer mientras no se llegue al final del archivo:

# Mientras no sea el fin del agregado de datos clientes

Mientras ( NOT Clientes.EOF )

.....

.....

**Fin mientras**

Cuando se llega al final del archivo secuencial se debe finalizar el listado y cerrar el agregado de datos.

**Cerrar AD Clientes #Se cierra el agregado de datos "Clientes"**

Ejemplo: Listado completo de Clientes (algoritmo de recorrido)

Código	Nombre	Saldo
1254	GONZALEZ, María Inés	1.258,30
1356	GOMEZ, Lucas	0,00
1458	MARTINEZ, José Ignacio	125,54
1490	CHIARAMELLO, Jorge	8.452,30
1510	MICOLINI, Julieta	5.034,00
1567	ARGUELLO, Mirian	1.290,00
1589	AGUIRRE, Marcos	0,00
1689	BASUALDO, Patricia	30,24
1690	VELAZQUEZ, Enrique	109,24
1783	MALDONADO, Silvio	0,00

Listado completo de clientes

Video "Ejemplo: Listado completo de clientes" | Elaboración Propia. DEPROE - Colegio Universitario IES

Veremos a continuación, un programa que imprime el contenido completo de un agregado de datos que almacena el código y nombre de los clientes y sus respectivos saldos.

En primera instancia declaramos una estructura registro con nombre cliente que contenga los campos Código, nombre y saldo como atributos.

A continuación declaramos un agregado de datos Clientes de tipo Cliente y en el procedimiento que emite el reporte, procedemos al manejo del agregado. En primera instancia lo abrimos y leemos el primer registro. Dentro de una estructura repetitiva, imprimimos en primera instancia el primer elemento que ya lo leímos fuera del ciclo, y posteriormente, leemos el siguiente registro. Así sucesivamente hasta

encontrar el final de archivo. Por último, cerramos el agregado de Clientes y finalizamos el programa

Si hacemos una comparación con nuestra Situación profesional, daremos con que el listado de las cuentas corrientes que muestre el número de la cuenta, el nombre del cliente y el saldo, pedido coincide con lo resuelto en este punto. Pudiendo obtener el mismo a través de un código similar al planteado en este ejemplo.

Para obtenerlo se imprimirán todos los registros hasta llegar al final del agregado de datos.

Listado de Cuentas Corrientes		
Código	NOMBRE	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00
12546	ORDÓÑEZ Paola Lola	560,01
13343	PAEZ Rodolfo	( - 350,00 )
15546	GARCIA Carlos	220,56

Se desarrolla el procedimiento del mismo:

Algoritmo de recorrido - SP PL2

```
Procedimiento ListadoGeneral()
    Imprimir "Listado de Cuentas Corrientes" Salto de Línea
    Imprimir "CódigoNombreSaldo" Salto de Línea
    #Apertura del Agregado de Datos
    Abrir AD Cuentas
    #Lectura del primer registro del Agregado de Datos
    Leer AD Cuentas
    #Se lee mientras no se llegue al fin de archivo
    Mientras ( NOT Cuentas.EOF )
        Imprimir Cuenta.CodCta Cuenta.Nombre Cuenta.Saldo Salto de Línea
        #Se lee mientras no se llegue al fin de archivo
        Leer AD Cuentas
    Fin Mientras
    #Cierre del Agregado de Datos
    Cerrar AD Cuentas
Fin Procedimiento
```

Estructura Repetitiva

### 3. Filtro de Campos de un Agregado de Datos

Habrá casos en donde solo necesite obtener algunos de los registros de un agregado de datos y leer secuencialmente los registros es inevitable, pero se puede colocar una alternativa dentro del recorrido. De este modo se verifica una condición deseada para generar la selección.

#### Ejemplo: Listado de clientes con saldo 0(cero)

El siguiente es un programa que emite un listado de los clientes que tienen saldo igual a cero.

# Definición del Registro  
Registro Cliente

### Listado de clientes con saldo 0

Código	Nombre	Saldo
1356	GOMEZ, Lucas	0,00
1589	AGUIRRE, Marcos	0,00
1783	MALDONADO, Silvio	0,00

Listado de clientes con saldo cero

Video "Ejemplo: Listado de clientes con saldo 0" | Elaboración Propia. DEPROE - Colegio Universitario IES

En primera instancia, declaramos una estructura registro con nombre cliente que contenga los campos Código, nombre y saldo como atributos. A continuación declaramos un agregado de datos Clientes de tipo Cliente, Ya en el procedimiento que emite el reporte, procedemos al manejo del agregado en primera instancia lo abrimos y leemos el primer registro Dentro de una estructura repetitiva, imprimimos en primera instancia el primer elemento que ya

lo leímos fuera del ciclo solo si el saldo es igual a cero y, posteriormente, leemos el siguiente registro e imprimimos si corresponde. Así sucesivamente, hasta encontrar el final de archivo. Por último, cerramos el agregado de Clientes y finalizamos el programa.

Observe que en este pseudocódigo de ejemplo, la lectura del AD dentro del bucle denominado "Mientras.....Fin Mientras" se realiza "fuera" del condicional "Si". Esto se debe a que vamos a querer leer el siguiente registro, hayamos o no impreso el anterior. Si colocáramos la operación de lectura junto con la de impresión, en el interior del condicional, sucedería que con el primer cliente cuyo saldo no sea igual a cero, el programa entraría en un bucle infinito, pues nunca más se ejecutaría la instrucción de lectura del AD y por tanto jamás se avanzaría sobre el AD, con lo que la condición del mientras (NOT Clientes.EOF) sería eternamente cierta y el bucle jamás finalizaría.

De la misma manera que obtenemos los clientes con saldo cero en este ejemplo, podemos obtener el Listado de cuentas que superen los \$1.000 de saldo (tal como nos pide la Situación profesional). Para ello aplicaremos un filtro similar al usado en el código antes resuelto.

Listado de las cuentas corrientes que tienen saldo mayor a mil pesos:

## Cuentas con saldo mayor a \$1.000

PL2

*El siguiente es el modelo del listado que se quiere imprimir:*

### Listado de Cuentas con Saldo Mayor a \$ 1000

Código	NOMBRE	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00

**Procedimiento** MayoresMil()

```
Imprimir "Listado de Cuentas con Saldo Mayor a $ 1000" Salto de Línea
Imprimir "CódigoNombreSaldo" Salto de Línea
# Apertura del Agregado de Datos
Abrir AD Cuentas
# Lectura del primer registro del Agregado de Datos
Leer AD Cuentas
# Se lee mientras no se llegue al fin de archivo
Mientras ( NOT Cuentas.EOF )
    # Se imprimen sólo los registros que cumplen la condición
    Si ( Cuenta.Saldo > 1000 )
        Imprimir Cuenta.CodCta Cuenta.Nombre Cuenta.Saldo Salto de Línea
    Fin Si
    # Lectura del siguiente registro
    Leer AD Cuentas
Fin Mientras
# Cierre del Agregado de Datos
Cerrar AD Cuentas
Fin Procedimiento
```

## 4. Búsqueda en un Agregado de Datos

Es posible realizar búsquedas dentro de un agregado de datos. Debe tener en cuenta que en los agregados de datos existe un campo (o conjunto de campos) al que se lo denomina **clave**. La clave es un campo que contiene un valor que no se repite en toda la Tabla o Agregado de Datos, y que por tanto identifica a cada registro de modo único. En general, los Agregados de Datos están siempre ordenados por ese campo clave, salvo que se especifique lo contrario.

Considere el agregado del ejemplo anterior. Muchos clientes pueden tener el mismo saldo, incluso existe la posibilidad de que dos clientes tengan el mismo nombre, pero nunca podrán tener el mismo código, pues este último es el campo clave que los identifica. La clave es única como el número de documento de un individuo.

- En caso de tener que realizar una búsqueda de datos por un **campo no clave**, el algoritmo anterior es adecuado ya que, en definitiva, se está realizando una búsqueda de los clientes que deben cero pesos.

- En caso de realizar una búsqueda por un **campo clave**, se deben tener en cuenta lo siguiente considerando que el agregado de datos esta ordenado:
  - 1. No es necesario recorrer el agregado de datos hasta llegar al final.
  - 2. Se realizan las lecturas necesarias hasta encontrar el dato buscado.

Explicaremos las búsquedas a partir de los siguientes problemas de ejemplo:

#### Ejemplo: Búsqueda de un Cliente

El siguiente programa, permitirá consultar el nombre y el saldo de un cliente a partir de su número de cuenta:

```

# Definición del Agregado de Datos
Agregado de Datos Clientes Tipo Registro Cliente
# Definición del Formulario
Formulario frmConsulta
  Marco CuentaCorriente
    Caja de texto txtCódigo
  Fin Marco
  Marco Resultado
    Etiqueta lblNombre
    Etiqueta lblSaldo
  Fin Marco

```

Video "[Ejemplo: Búsqueda de un cliente](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Al igual que en los casos anteriores, en primer instancia declaramos una estructura registro con nombre cliente que contenga los campos Código, nombre y saldo como atributos. A continuación declaramos un agregado de datos Clientes de tipo Cliente. Ya en el procedimiento de consulta, procedemos al manejo del agregado. Y abrimos el primer registro. Y leemos el primer registro. Buscamos el dato recorriendo el agregado de datos. Una vez que salimos de la estructura "Mientras....Fin Mientras", verificamos que

realmente encontramos el dato buscado. De ser así mostramos el código de cliente y el saldo del mismo. En caso contrario, mostramos un mensaje al usuario indicando que el código ingresado no pertenece a ningún cliente del agregado de datos Clientes. Por último, cerramos el agregado de datos y concluimos con el programa.

## PSEUDO: Busqueda cliente

### Programa ConsultarCuenta

# Definición de Datos Globales

# Definición del Registro

Registro Cliente

    Campo Código Tipo Numérico

    Campo Nombre Tipo Cadena

    Campo Saldo Tipo Numérico

Fin Registro

# Definición del Agregado de Datos

Agregado de Datos Clientes Tipo Registro Cliente

# Definición del Formulario

Formulario frmConsulta

    Marco CuentaCorriente

        Caja de texto txtCódigo

    Fin Marco

    Marco Resultado

        Etiqueta lblNombre

        Etiqueta lblSaldo

    Fin Marco

    Botón de Comando cmdConsultar

Fin Formulario

# Desarrollo de los Procedimientos de Evento

### Procedimiento cmdConsultar:Click

    # Apertura del Agregado de Datos

    Abrir AD Clientes

    # Lectura del primer registro del Agregado de Datos

    Leer AD Clientes

    # Se mantiene en el bucle mientras no encuentre el dato y no

    # llegue al final del agregado de datos

    Mientras ( NOT Clientes.EOF AND Cliente.Codigo <> txtCodigo )

        # Lectura del siguiente registro

        Leer AD Clientes

    Fin Mientras

    # Si encontró el código muestra los datos

    Si ( Cliente.Codigo = txtCodigo )

        lblNombre = Cliente.Nombre

        lblSaldo = Cliente.Saldo

    # Si no encontró el código muestra un mensaje notificando al

    # usuario

    Si No

        Mensaje

            Título: "Cliente Inexistente"

            Icono: Advertencia

            Texto: "El código ingresado no corresponde a ningún  
                  cliente."

        Fin Mensaje

    Fin Si

    # Cierre del Agregado de Datos

    Cerrar AD Clientes

Fin Procedimiento

[Fin programa](#)

Observe que la estructura repetitiva tiene **dos condiciones de corte**. Eso significa que es necesario preguntar por cuál de las dos condiciones se salió del bucle.

Puede ser:

- Porque se encontró lo que se buscaba, o;
- Porque ya no hay más registros sin leer en el agregado de datos (se llegó al final de archivo).

Precisamente por eso, el condicional que sigue luego de la estructura repetitiva nos permite saber si el cliente fue encontrado, o no; actuando en consecuencia.

## 5. Búsqueda en un AD mejorado

Si se busca un código no almacenado seguirá leyendo hasta el final. ¿No cree que se podría mejorar? Suponga que el siguiente es el agregado de datos tratado y se busca el código: 1500.

Tipo Registro Cliente		
	Código	Nombre
Leer	1254	GONZALEZ, María Inés
	1356	GOMEZ, Lucas
	1458	MARTINEZ, José Ignacio
	1490	CHIARAMELLO, Jorge
	1510	MICOLINI, Julieta
	1567	ARGUELLO, Mirian
	1589	AGUIRRE, Marcos
	1689	BASUALDO, Patricia
	1690	VELAZQUEZ, Enrique
	Búsqueda de un AD	MALDONADO, Silvio
EOF ← Final de archivo		

Video "Búsqueda de un AD" | Elaboración Propia. DEPROE - Colegio Universitario IES

Una vez que se realizaron cinco lecturas, ya no es necesario seguir leyendo. Note que el registro marcado contiene el código 1510, un valor mayor al buscado. Los siguientes registros tienen valores más grandes que éste, precisamente por ser Código el campo clave del AD. Ya no hay posibilidad de encontrar lo que se busca.

Observe el siguiente ejemplo donde se soluciona este inconveniente (Atención: solo se desarrolla el

procedimiento de evento):

#### Ejemplo: Búsqueda de un Cliente con salida anticipada

```
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdConsultar:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    # Lectura del primer registro del Agregado de Datos
    Leer AD Clientes
    # Sale del mientas cuando encuentra un dato mayor o igual al
    buscado, o cuando llega al final.
    Mientras ( NOT Clientes.EOF AND Cliente.Codigo < txtCodigo )
        # Lectura del siguiente registro
        Leer AD Clientes
    Fin Mientras
```

Búsqueda de un cliente con salida anticipada

Video "[Ejemplo: Búsqueda de un cliente con salida anticipada](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Procedimiento que busca un cliente por código. Sale del mientas cuando encuentra un dato mayor o igual al buscado, o cuando llega al final. Si encontró el código muestra los datos. Si no encontró el código muestra un mensaje notificando al usuario

En nuestra Situación profesional no tenemos ninguna búsqueda individual, pero en caso de querer buscar una cuenta corriente en particular podríamos usar un algoritmo como el descripto anteriormente.

#### 6. Estructuras de recorrido a utilizar

Usted se plantea que cuando se abre el agregado de datos se realiza la primera lectura bajo ninguna condición, después se ingresa a la estructura repetitiva. ¿No es más eficiente utilizar la estructura "Hacer...Hasta"?

Los tres procedimientos vistos se pueden resolver perfectamente con la estructura "Hacer...Hasta", pero hay casos donde es adecuada y otros donde no lo es.

A continuación se desarrollan los tres procedimientos vistos con la estructura mencionada y hacemos algunos comentarios con respecto a los mismos.

#### Ejemplo: Impresión del contenido completo de un AD de clientes

En el procedimiento que imprime el contenido completo de un agregado de datos que almacena el código y

nombre de los clientes y sus respectivos saldos.

Se lo puede programar de dos modos, pero ninguno es más adecuado que utilizando un "Mientras".

### Método #1

En este caso se realizó el programa del mismo modo que si se utilizará un mientras: una lectura antes y otra dentro de la repetitiva.

```
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdImprimir:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    # Lectura del primer registro del Agregado de Datos
    Leer AD Clientes
    Hacer
        Imprimir Cliente,Codigo Cliente,Nombre Cliente.Saldo,Salto de Línea
        # Lectura del siguiente registro
        Leer AD Clientes
    # Sale cuando llega al final de archivo
    Hasta (Clientes.EOF)
    # Cierre del Agregado de Datos
    Cerrar AD Clientes
Fin Procedimiento
```

¿Qué sucederá si el AD que queremos procesar está vacío? La primera lectura, antes que la repetitiva, ya generará la condición de EOF, pero como el Hacer...Hasta verifica la condición al final, lo mismo entrará al bucle. Entonces ¿Qué imprimiría la línea de impresión si no hay ningún registro con datos cargados?

### Método #2

En ese caso se utiliza la estructura como corresponde, con sólo una lectura en la parte interna. Pero se presenta el inconveniente que primero lee y luego imprime. Eso significa que llegará un momento en el que se llegará al final de archivo e imprimirá lo mismo. Para solucionar esto se puede utilizar una alternativa. De todos modos este, no es el algoritmo adecuado.

```
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdImprimir:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    Hacer
        # Lectura del registro
        Leer AD Clientes
        Si (NOT Clientes.EOF)
            Imprimir Cliente.Codigo,Cliente.Nombre, Cliente,Saldo, Salto de Línea
        Fin Si
    # Sale cuando llega al final de archivo
    Hasta (Clientes.EOF)
    # Cierre del Agregado de Datos
    Cerrar AD Clientes
Fin Procedimiento
```

### Ejemplo: Listado de Clientes con saldo igual a cero (usando nueva estructura).

Vemos nuevamente un procedimiento que emite un listado de los clientes que tienen saldo igual a cero pero en este caso utilizando una nueva estructura de recorrido.

Como en esta situación la alternativa debe utilizarse para decidir si se imprime o no, solo se incorpora una condición para que el programa funcione correctamente con el "Hacer...Hasta".

### Listado de Clientes con saldo cero (Nueva estructura) PL2

```
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdImprimir:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    Hacer
        # Lectura del registro
        Leer AD Clientes
        Si ( NOT Clientes.EOF AND Cliente.Saldo = 0 )
            Imprimir Cliente.Codigo, Cliente.Nombre, Cliente.Saldo, SL
        Fin Si
        # Sale cuando llega al final del archivo
        Hasta ( Clientes.EOF )
        # Cierre del Agregado de Datos
        Cerrar AD Clientes
    Fin Procedimiento
```

Condición

### Ejemplo: Búsqueda de un Cliente (usando nueva estructura)

Veamos este procedimiento que busca un cliente por código para mostrar el nombre y el saldo del mismo, utilizando la nueva estructura vista.

En este caso el procedimiento tiene una sola línea de programa dentro de la repetitiva: la lectura del registro. Por consiguiente, este sí es un caso donde es más adecuado utilizar un "Hacer... Hasta" que un "Mientras".

```

# Desarrollo de los Procedimientos de Evento
Procedimiento cmdConsultar:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    Hacer
        # Lectura del registro
        Leer AD Clientes
        # Sale de la repetitiva cuando encuentra un código de cliente mayor
        # o igual al buscado, o cuando llega al final del Agregado de Datos
        Hasta ( Clientes.EOF OR Cliente.Codigo >= txtCodigo )

        # Si encontró el código muestra los datos
        Si ( Cliente.Codigo = txtCodigo )
            lblNombre = Cliente.Nombre
            lblSaldo = Cliente.Saldo
            # Si no encontró el código muestra un mensaje notificando al usuario
            Si No
                Mensaje
                    Título: "Cliente Inexistente"
                    Icono: Advertencia
                    Texto: "El código ingresado no corresponde a ningún cliente."
                Fin Mensaje
            # Cierre del Agregado de Datos
            Cerrar AD Clientes
        Fin Procedimiento

```

En nuestra Situación profesional, emitiremos un listado en base a la búsqueda de cuentas deudoras y lo resolveremos de la siguiente manera:

#### Listado de cuentas deudoras

El siguiente es el modelo del listado que se quiere imprimir:

Listado de Cuentas Corrientes		
Código	NOMBRE	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00
12546	ORDÓÑEZ Paola Lola	560,01
13343	PAEZ Rodolfo	( - 350,00 )
15546	GARCIA Carlos	220,56

En este caso el procedimiento es muy parecido al anterior, solo que cambia la condición y se requiere de un acumulador. Este último es conveniente definirlo como un dato local ya que es el único procedimiento que lo

utiliza.

## Listado de cuentas deudoras

PL2

```
Procedimiento CuentasDeudoras()
    Variable Numérica Total = 0
    Imprimir "Listado de Cuentas Deudoras" , Salto de Línea
    Imprimir "CódigoNombreSaldo", Salto de Línea
    # Apertura del Agregado de Datos
    Abrir AD Cuentas
    # Lectura del primer registro del Agregado de Datos
    Leer AD Cuentas
    # Se lee mientras no se llegue al fin de archivo
    Mientras ( NOT Cuentas.EOF )
        # Las cuentas son saldo mayor a cero son deudoras
        Si ( Cuenta.Saldo > 0 )
            # Se imprimen y acumulan sólo los registros que cumplen la condición
            Imprimir Cuenta.CodCta, Cuenta.Nombre, Cuenta.Saldo, Salto de Línea
            Total = Total + Cuenta.Saldo
        Fin Si
        # Lectura del siguiente registro
        Leer AD Cuentas
    Fin Mientras
    # Cierre del Agregado de Datos
    Cerrar AD Cuentas
    # Se imprime el total obtenido
    Imprimir "Total Cuentas Deudoras .... $" , Total
Fin Procedimiento
```

### 7. Intervalos de Registros

Es posible imprimir los registros que estén dentro de un intervalo. Pero lo primero que hay que considerar cuando se realiza el algoritmo es si los registros están ordenados por el campo que se quiere controlar o no.

Analizaremos las diferentes situaciones por medio de ejemplos:

#### Ejemplo: Listado de clientes en un intervalo de números de cuenta

Si se le pide desarrollar un programa que debe imprimir los clientes que están comprendidos dentro de un intervalo de números de cuenta (código), considerando que el agregado de datos está ordenado por este campo clave.

Existen dos posibles soluciones, una es más simple, resuelve el problema pero lee todo el archivo. La otra es un poco más compleja de programar, pero a la hora de ejecutarse es más eficiente.

Analizaremos ambas soluciones:

Código	Nombre	Saldo
1254	GONZALEZ, María Inés	1.258,30
1356	GOMEZ, Lucas	0,00
1458	MARTINEZ, José Ignacio	125,54
1490	CHIARAMELLO, Jorge	8.452,30
1510	MICOLINI, Julieta	5.034,00
1567	ARGUELLO, Mirian	1.290,00
1589	AGUIRRE, Marcos	0,00
1689	BASUALDO, Patricia	30,24
1783	VILLAZON, EZ, Enrique	109,24
1783	MALDONADO, Silvio	0,00

Video "Dos posibles soluciones part 1" | Elaboración Propia. DEPROE - Colegio Universitario IES  
[https://www.youtube.com/watch?v=LDqTYZS\\_GnU](https://www.youtube.com/watch?v=LDqTYZS_GnU)

#### PSEUDO: part1

```

Programa ImprimirIntervalo
# Definición de Datos Globales
# Definición del Registro
Registro Cliente
    Variable Numérica Código
    Variable Alfanumérica Nombre
    Variable Numérica Saldo
Fin Registro

# Definición del Agregado de Datos
Agregado de Datos Clientes Tipo Registro Cliente
# Definición del Formulario
Formulario frmImprimir
    Marco mrcRangoCuentas
        Caja de Texto txtDesde
        Caja de Texto txtHasta
    Fin Marco
    Botón de Comando cmdImprimir

```

Fin Formulario

```
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdImprimir:Click
    # Apertura del Agregado de Datos
    Abrir AD Clientes
    # Lectura del primer registro del Agregado de Datos
    Leer AD Clientes
    Imprimir "Listado de Clientes", Salto de línea

    Mientras ( NOT Clientes.EOF )
        # Si el código está dentro del intervalo, se imprime el registro
        Si ( Cliente.Codigo >= txtDesde AND Cliente.Codigo <= txtHasta )
            Imprimir Cliente.Codigo,Cliente.Nombre,Cliente.Saldo,SL
        Fin Si
        # Lectura del siguiente registro
        Leer AD Clientes
    Fin Mientras

    # Cierre del Agregado de Datos
    Cerrar AD Clientes
Fin Procedimiento
```

Fin Programa

## Dos posibles soluciones

PL2

Si se le pide desarrollar un programa que debe imprimir los clientes que están comprendidos dentro de un intervalo de números de cuenta (código), considerando que el agregado de datos está ordenado por este campo clave.



Existen dos posibles soluciones, una es más simple, resuelve el problema pero lee todo el archivo. La otra es un poco más compleja de programar, pero a la hora de ejecutarse es más eficiente.

Analizaremos ambas soluciones:

```

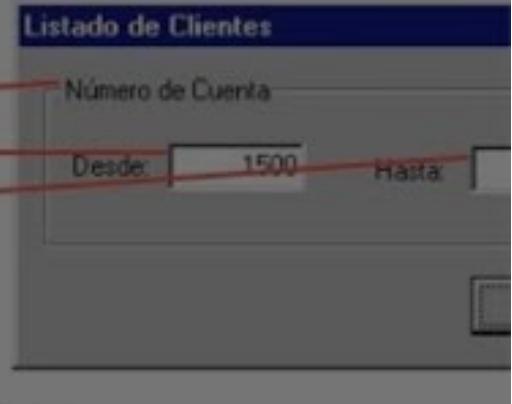
Variable Numérica      Saldo
Fin Registro

# Definición del Agregado de Datos
Agregado de Datos Clientes Tipo Registro Cliente

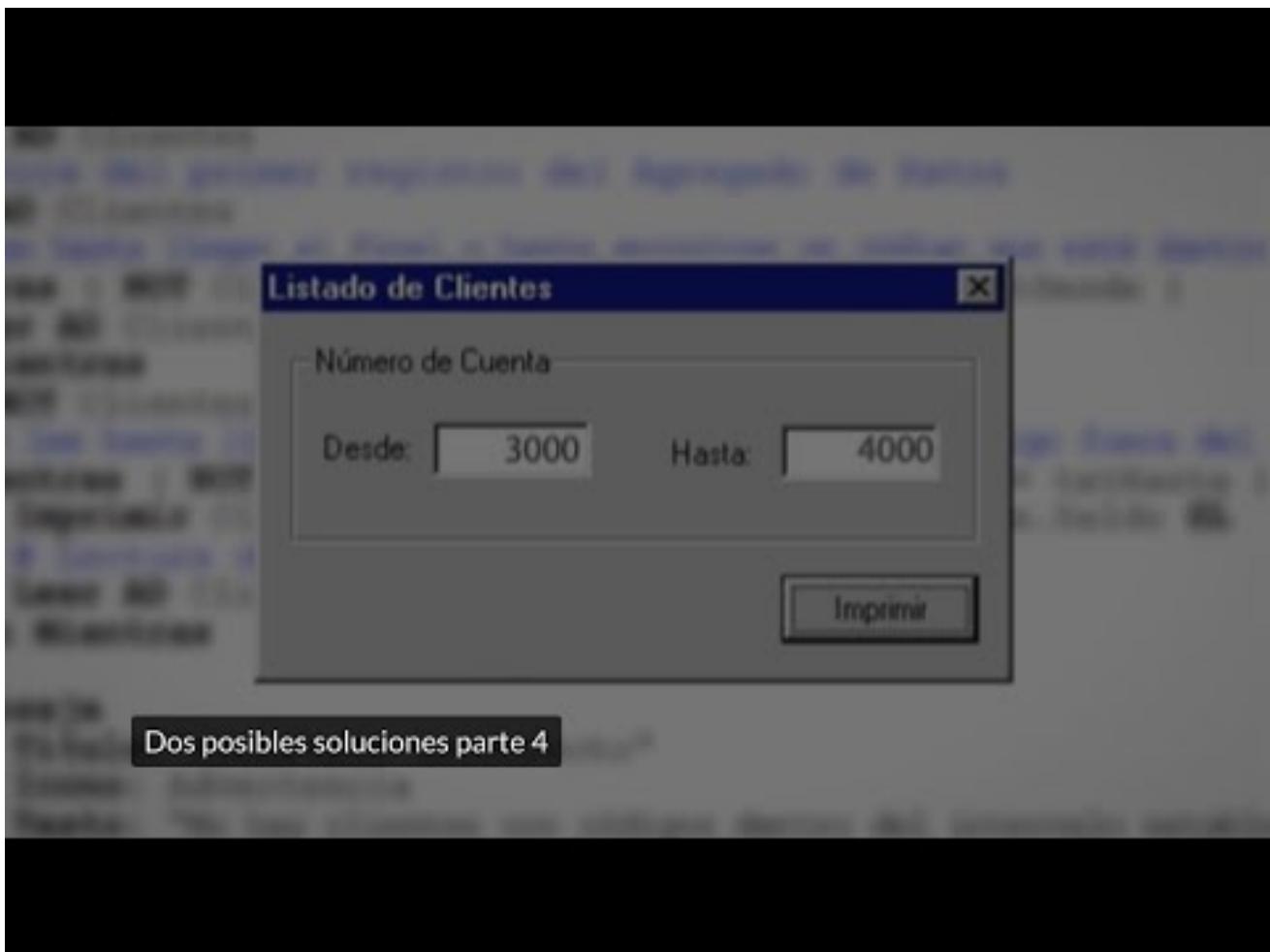
# Definición del Formulario
Formulario frmImprimir
  Marco mrcRangoCuentas           Número de Cuenta
    Caja de Texto txtDesde          Desde: 
    Caja de Texto txtHasta          Hasta: 
  Fin Marco
  Botón de Comando cmdImprimir
Fin Formulario           Dos posibles soluciones parte 3

# Desarrollo de los Procedimientos de Evento

```



Video "Dos posibles soluciones part 3" | Elaboración Propia. DEPROE - Colegio Universitario IES  
<https://www.youtube.com/watch?v=85Y1zkDw7xk>



Video "Dos posibles soluciones part 4" | Elaboración Propia. DEPROE - Colegio Universitario IES  
<https://www.youtube.com/watch?v=LapOXf09fiU>

## 8. El orden de los datos determina la estructura del algoritmo

A continuación se desarrollan algunos ejemplos de programas donde se puede verificar que la lógica del programa depende del ordenamiento del agregado de datos.

Solo se desarrollan los procedimientos de eventos. Esto es una síntesis que tiene como objetivo la comparación y análisis de los diferentes modos en que se puede explorar un agregado de datos y no son los únicos modos de hacerlo.

### Ejemplo: Empresa de turismo

Una empresa de turismo cuenta con un agregado de datos conformado por registros que tienen la siguiente estructura:

```

Estructura Registro Viaje
  Variable Numérica Fecha          #fecha del viaje
  Variable Alfanumérica Origen    #Ciudad de origen
  Variable Alfanumérica Destino   #Ciudad de destino
  Variable Numérica Cantidad     #Cantidad de pasajes vendidos
  Variable Numérica Precio       #Precio del pasaje
Fin Registro

```

```

Agregado de Datos Viajes Tipo Registro Viaje

```

El agregado de datos está ordenado por Fecha como primer criterio de orden y Origen como segundo criterio. Se requieren los siguientes programas:

### Caso 1

Programa que muestre en pantalla los viajes que se realizarán en un intervalo de fechas definido por el usuario. Tenga en cuenta que el **usuario elige las Fechas** y el **Agregado de Datos** está ordenado por **Fecha**. Esto permite saltar los registros que no son necesarios y luego imprimir mientras se lean registros que están dentro del intervalo.

```

Procedimiento cmdEmitir:Click
  Variable Numérica f = 1      # Variable que se utiliza para subindicar la
                                # fila de la grilla.
  Abrir AD Viajes
  Leer AD Viajes
  # Repetitiva que se utiliza para "pasar" los registros no necesarios.
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha < txtDesde )
    Leer AD Viajes
  Fin Mientras

  # Repetitiva que se utiliza para mostrar la información de los registros
  # necesarios.
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha <= txtHasta )
    Grilla[f][1] = Viaje.Fecha
    Grilla[f][2] = Viaje.Origen
    Grilla[f][3] = Viaje.Destino
    Grilla[f][4] = Viaje.Cantidad
    Grilla[f][5] = Viaje.Precio
    Grilla[f][6] = Viaje.Precio * Viaje.Cantidad
    f = f + 1
    Leer AD Viajes
  Fin Mientras
  Cerrar AD Viajes
Fin Procedimiento

```

### Caso 2

Programa que emita un listado detallado de los viajes que se realizarán hacia un **Destino** en particular. El

listado deberá visualizarse en pantalla. Tenga en cuenta que: **el usuario elige un Destino pero el agregado de datos está ordenado por Fecha**. Esto significa que no se podrá saltar un bloque de los registros que no sean necesarios, ya que no están todos juntos. Los viajes a un destino en particular pueden estar intercalados a lo largo de todo el agregado de datos. Por consiguiente será necesario leer completamente el agregado de datos y mostrar sólo los registros necesarios.

```
Procedimiento cmdEmitir:Click
    Variable Numérica f = 1 # Variable que se utiliza para subindicar la
    fila de la grilla.
    Abrir AD Viajes
    Leer AD Viajes

    # Repetitiva que recorre todo el agregado de datos. Con una alternativa
    se controla que los registros cumplan con una condición dada para ser
    mostrados.
    Mientras ( NOT Viajes.EOF )
        Si ( Viaje.destino = lstDestino )
            Grilla[f][1] = Viaje.Fecha
            Grilla[f][2] = Viaje.Origen
            Grilla[f][3] = Viaje.Cantidad
            Grilla[f][4] = Viaje.Precio
            Grilla[f][5] = Viaje.Precio * Viaje.Cantidad
            f = f + 1
        Fin Si
        Leer AD Viajes
    Fin Mientras

    Cerrar AD Viajes
Fin Procedimiento
```

### Caso 3

Programa que emita un listado (en pantalla) que muestre la cantidad de pasajes vendidos y el monto cobrado por día. En este caso el usuario no selecciona ninguna opción, pero el requerimiento indica que se debe mostrar una fecha, la cantidad de pasajes vendidos en esa fecha y el monto recaudado. En una fecha en particular pueden salir varios viajes para diferentes destinos, por lo que será necesario acumular las cantidades y los montos. El requerimiento es: **por Fecha, y el Agregado de Datos está ordenado por Fecha**. Esto significa que se podrán acumular todos los importes de un día en particular, para luego mostrarlos; después se hará lo mismo con los de los días siguientes.

```

Procedimiento cmdEmitir:Click
    Variable Numérica f = 1 # Variable que se utiliza para subindicar la
    fila de la grilla.
    Variable Numérica AuxFecha # Variable que se utiliza para controlar el
    cambio de fecha.
    Variable Numérica CantPasajes = 0 # Variable que se utiliza como
    acumulador.
    Variable Numérica TotImporte = 0 # Variable que se utiliza para acumular
    los importes.
    Abrir AD Viajes
    Leer AD Viajes
    Mientras ( NOT Viajes.EOF )
        AuxFecha = Viaje.Fecha
        CantPasajes = 0
        TotImporte = 0
        Mientras ( NOT Viajes.EOF AND Viaje.Fecha = AuxFecha )
            CantPasajes = CantPasajes + Viaje.CantPasajes
            TotImporte = TotImporte + (Viaje.Cantidad * Viaje.Precio)
            Leer AD Viajes
        Fin Mientras
        Grilla[f][1] = AuxFecha
        Grilla[f][2] = CantPasajes
        Grilla[f][3] = TotImporte
        f = f + 1
    Fin Mientras
    Cerrar AD viajes
Fin Procedimiento

```

### Casos 1 y 2 combinados

Programa que emita un listado detallado de los viajes que se realizarán en un período de **fechas** hacia un **destino** en particular. El listado deberá visualizarse en pantalla.

```

Procedimiento cmdEmitir:Click
  Variable Numérica f = 1 # Variable que se utiliza para subindicar la
  fila de la grilla.
  Abrir AD Viajes
  Leer AD Viajes
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha < txtDesde )
    Leer AD Viajes
  Fin Mientras
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha <= txtHasta )
    Si ( Viaje.destino = lstDestino )
      Grilla[f][1] = Viaje.Fecha
      Grilla[f][2] = Viaje.Origen
      Grilla[f][3] = Viaje.Cantidad
      Grilla[f][4] = Viaje.Precio
      Grilla[f][5] = Viaje.Precio * Viaje.Cantidad
      f = f + 1
    Fin Si
    Leer AD Viajes
  Fin Mientras
  Cerrar AD Viajes
Fin Procedimiento

```

### Casos 2 y 3 combinados

Programa que emita un listado que muestre la cantidad de pasajes vendidos y el monto cobrado por día de los viajes que se realicen hacia un **destino** en particular.

```

Procedimiento cmdEmitir:Click
    Variable Numérica f = 1 # Variable que se utiliza para subindicar la
    fila de la grilla
    Variable Numérica AuxFecha # Variable que se utiliza para controlar el
    cambio de fecha
    Variable Numérica CantPasajes = 0 # Variable que se utiliza como
    acumulador
    Variable Numérica TotImporte = 0 # Variable que se utiliza para
    almacenar el precio
    Abrir AD Viajes
    Leer AD Viajes
    Mientras ( NOT Viajes.EOF )
        AuxFecha = Viaje.Fecha
        TotImporte = 0
        CantPasajes = 0
        Mientras ( NOT Viajes.EOF AND Viaje.Fecha = AuxFecha )
            Si ( Viaje.Destino = 1stDestino )
                CantPasajes = CantPasajes + Viaje.Cantidad
                TotImporte= TotImporte + (Viaje.Cantidad * Viaje.Precio)
            Fin Si
            Leer AD Viajes
        Fin Mientras
        Grilla[f][1] = AuxFecha
        Grilla[f][2] = CantPasajes
        Grilla[f][3] = TotImporte
        f = f + 1
    Fin Mientras
    Cerrar AD Viajes
Fin Procedimiento

```

### Casos 1 y 3 combinados

Programa que emita un listado que muestre la cantidad de pasajes vendidos y el monto cobrado por día de los viajes que se realicen en un período de **Fechas** determinado por el usuario.

```

Procedimiento cmdEmitir:Click
  Variable Numérica f = 1 # Variable que se utiliza para subindicar la
  fila de la grilla
  Variable Numérica AuxFecha # Variable que se utiliza para controlar el
  cambio de fecha
  Variable Numérica CantPasajes = 0 # Variable que se utiliza como
  acumulador
  Variable Numérica TotImporte = 0 # Variable que se utiliza para
  almacenar el precio
  Abrir AD Viajes
  Leer AD Viajes
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha < txtDesde )
    Leer AD Viajes
  Fin Mientras

  Mientras ( NOT Viajes.EOF AND Viaje.Fecha <= txtHasta )
    AuxFecha = Viaje.Fecha
    TotImporte = 0
    CantPasajes = 0

    Mientras ( NOT Viajes.EOF AND Viaje.Fecha = AuxFecha )
      CantPasajes = CantPasajes + Viaje.Cantidad
      TotImporte = TotImporte + (Viaje.Cantidad * Viaje.Precio)
      Leer AD Viajes

    Fin Mientras
    Grilla[f][1] = AuxFecha
    Grilla[f][2] = CantPasajes
    Grilla[f][3] = TotImporte
    f = f + 1
  Fin Mientras
  Cerrar AD Viajes
Fin Procedimiento

```

### Casos 1, 2 y 3 combinados

Programa que emita un listado que muestre la cantidad de pasajes vendidos y el monto cobrado por día de los viajes que se realicen en un período de **Fechas** determinado por el usuario, pero solo de los viajes que se dirigen a un **Destino** en particular.

```

Procedimiento cmdEmitir:Click
  Variable Numérica f = 1 # Variable que se utiliza para subindicar la
  fila de la grilla
  Variable Numérica AuxFecha # Variable que se utiliza para controlar el
  cambio de fecha
  Variable Numérica CantPasajes = 0 # Variable que se utiliza como
  acumulador
  Variable Numérica TotImporte = 0 # Variable que se utiliza para
  almacenar el precio
  Abrir AD Viajes
  Leer AD Viajes
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha < txtDesde )
    Leer AD Viajes
  Fin Mientras
  Mientras ( NOT Viajes.EOF AND Viaje.Fecha <= txtHasta )
    AuxFecha = Viaje.Fecha
    TotImporte = 0
    CantPasajes = 0
    Mientras ( NOT Viajes.EOF AND Viaje.Fecha = AuxFecha )
      Si ( Viaje.Destino = lstDestino )
        CantPasajes = CantPasajes + Viaje.Cantidad
        TotImporte = TotImporte + (Viaje.Cantidad * Viaje.Precio)
      Fin Si
      Leer AD Viajes
    Fin Mientras
    Grilla[f][1] = AuxFecha
    Grilla[f][2] = CantPasajes
    Grilla[f][3] = TotImporte
    f = f + 1
  Fin Mientras
  Cerrar AD Viajes
Fin Procedimiento

```



¡Vamos a comprobar cuánto aprendiste!

**1. Indique la opción correcta**

La clave de un registro es:

- La clave es un campo que contiene un valor que no se repite.
- La clave es el campo por el cual se comienza la lectura de un AD.
- La clave debe ser un campo numérico.

**2. Indique la opción correcta**

Marque cual de las siguientes afirmaciones corresponde a Agregado de Datos.

- Es un conjunto de registros de personas.
- Es un conjunto de registros idénticos.
- Es un conjunto de archivos en memoria.

**3. Indique la opción correcta**

La siguiente fracción de código de un procedimiento muestra el nombre del cliente guardado en el primer registro de un agregado de datos Clientes que contiene registros de tipo Cliente.

- Verdadero
- Falso

**4. Indique la opción correcta**

La siguiente fracción de código de un procedimiento, muestra el nombre del cliente guardado en el cuarto (índice 4) registro de un agregado de datos Clientes que contiene registros de tipo Cliente.

- Verdadero
- Falso

**5. Indique la opción correcta**

La siguiente fracción de código de un procedimiento, recorre todo el AD Clientes y muestra el nombre de los clientes contenidos en cada registro.

- Verdadero

Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

La clave de un registro es:

- La clave es un campo que contiene un valor que no se repite.
- La clave es el campo por el cual se comienza la lectura de un AD.
- La clave debe ser un campo numérico.

## 2. Indique la opción correcta

Marque cual de las siguientes afirmaciones corresponde a Agregado de Datos.

- Es un conjunto de registros de personas.
- Es un conjunto de registros idénticos.
- Es un conjunto de archivos en memoria.

## 3. Indique la opción correcta

La siguiente fracción de código de un procedimiento muestra el nombre del cliente guardado en el primer registro de un agregado de datos Clientes que contiene registros de tipo Cliente.

- Verdadero
- Falso

## 4. Indique la opción correcta

La siguiente fracción de código de un procedimiento, muestra el nombre del cliente guardado en el cuarto (índice 4) registro de un agregado de datos Clientes que contiene registros de tipo Cliente.

- Verdadero
- Falso

## 5. Indique la opción correcta

La siguiente fracción de código de un procedimiento, recorre todo el AD Clientes y muestra el nombre de los clientes contenidos en cada registro.

- Verdadero
- Falso

# SP1 / Ejercicio resuelto

La Situación profesional plantea la necesidad de:

Desarrollar la interfaz y el pseudocódigo que posibilite la obtención de los siguientes listados:

- Listado de cuentas y sus respectivos saldos.
- Listado de cuentas que superen los \$ 1.000 de saldo.
- Listado de cuentas "deudoras" con una línea de Total Adeudado a la Empresa.
- Informe de Saldo Total de las cuentas y el monto promedio de las mismas.

Para generar los listados que se solicitan la información debe estar almacenada en un agregado de datos. Dicho agregado de datos debe tener como mínimo los siguientes campos: Número de Cuenta, Nombre del cliente y Saldo. Como el enunciado del problema no determina cuál es el agregado de datos, consideremos que el siguiente ejemplo es el agregado que se debe procesar:

AD Cuentas			PL2
AD Cuentas			
CodCta	Nombre	Saldo	
11345	GONZALEZ Juan Carlos	2.350,34	
11389	PEREZ Agustín Antonio	1.555,00	
12546	ORDOÑEZ Paola Lola	560,01	
13343	PAEZ Rodolfo	(- 350,00)	
15546	GARCÍA Carlos	220,56	

EOF

El registro en el que definimos los campos para el Agregado de Datos es el siguiente:

## # Definición del Registro

### Registro Cuenta

Variable Numérica CodCta

Variable Alfanumérica Nombre

Variable Numérica Saldo

### Fin Registro

## # Definición del Agregado de Datos

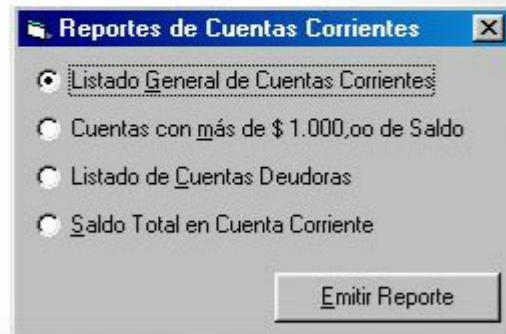
### Agregado de Datos Cuentas Tipo Registro Cuenta

Considerando que los cuatro se resuelven mediante la emisión de listados, y que los mismos no están parametrizados (no requieren el ingreso de ningún parámetro por parte del usuario), la interfaz de usuario requerida resulta muy elemental. Apenas bastaría con un formulario MDI en el que hubiera un menú con cuatro opciones correspondientes a cada uno de los listados solicitados.

Una alternativa con mayores características de interfaz de usuario (GUI) podría ser la siguiente:

A partir de la selección de los botones de opción se generará el listado que corresponda. Analizaremos cada procedimiento por separado, y finalmente se realizará el programa completo.

- Listado general de las cuentas corrientes**
- Cuentas con más de \$1.000 de Saldo**
- Listado de Cuentas Deudoras**
- Saldo Total en Cuenta Corriente**
- Programa completo**



- **Listado general de las cuentas corrientes**

El siguiente es el modelo del listado que se quiere imprimir:

Listado de Cuentas Corrientes		
Código	Nombre	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00
12546	ORDÓÑEZ Paola Lola	560,01
13343	PAEZ Rodolfo	( - 350,00 )
15546	GARCIA Carlos	220,56

Para obtenerlo se imprimirán todos los registros hasta llegar al final del agregado de datos. Se desarrolla el procedimiento del mismo:

```

Procedimiento ListadoGeneral()
  Imprimir "Listado de Cuentas Corrientes" Salto de Línea
  Imprimir "Código Nombre Saldo" Salto de Línea
  # Apertura del Agregado de Datos
  Abrir AD Cuentas
  # Lectura del primer registro del Agregado de Datos
  Leer AD Cuentas
  # Se lee mientras no se llegue al fin de archivo
  Mientras ( NOT Cuentas.EOF )
    Imprimir Cuenta.CodCta Cuenta.Nombre Cuenta.Saldo Salto de Línea
    # Se lee mientras no se llegue al fin de archivo
    Leer AD Cuentas
  Fin Mientras
  # Cierre del Agregado de Datos
  Cerrar AD Cuentas
Fin Procedimiento

```

- Cuentas con más de \$1.000 de Saldo

El siguiente es el modelo del listado que se quiere imprimir:

Listado de Cuentas con Saldo Mayor a \$ 1000		
Código	Nombre	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00

```

Procedimiento MayoresMil()
  Imprimir "Listado de Cuentas con Saldo Mayor a $1000" Salto de Línea
  Imprimir "Código Nombre Saldo" Salto de Línea
  # Apertura del Agregado de Datos
  Abrir AD Cuentas
  # Lectura del primer registro del Agregado de Datos
  Leer AD Cuentas
  # Se lee mientras no se llegue al fin de archivo
  Mientras ( NOT Cuentas.EOF )
    # Se imprimen sólo los registros que cumplen la condición
    Si ( Cuenta.Saldo > 1000 )
      Imprimir Cuenta.CodCta Cuenta.Nombre Cuenta.Saldo Salto de Línea
    Fin Si
    # Lectura del siguiente registro
    Leer AD Cuentas
  Fin Mientras
  # Cierre del Agregado de Datos
  Cerrar AD Cuentas
Fin Procedimiento

```

- Listado de Cuentas Deudoras

El siguiente es el modelo del listado que se quiere imprimir:

Listado de Cuentas Deudoras		
Código	Nombre	Saldo
11345	GONZALEZ Juan Carlos	2.350,34
11389	PEREZ Agustín Antonio	1.555,00
12546	ORDÓÑEZ Paola Lola	560,01
15546	GARCIA Carlos	220,56
<b>Total Adeudado</b>		<b>4.685,91</b>

En este caso el procedimiento es muy parecido al anterior, solo que cambia la condición y se requiere de un acumulador. Este último es conveniente definirlo como un dato local ya que es el único procedimiento que lo utiliza.

```
Procedimiento CuentasDeudoras()
    Variable Numérica Total = 0
    Imprimir "Listado de Cuentas Deudoras", Salto de Línea
    Imprimir "Código Nombre Saldo", Salto de Línea
    #Apertura del Agregado de Datos
    Abrir AD Cuentas
    #Lectura del primer registro del Agregado de Datos
    Leer AD Cuentas
    #Se lee mientras no se llegue al fin de archivo
    Mientras ( NOT Cuentas.EOF )
        #Las cuentas son saldo mayor a cero son deudoras
        Si ( Cuenta.Saldo > 0 )
            #Se imprimen y acumulan sólo los registros que cumplen la condición
            Imprimir Cuenta.CodCta, Cuenta.Nombre, Cuenta.Saldo, Salto de Línea
            Total = Total + Cuenta.Saldo
        Fin Si
        #Lectura del siguiente registro
        Leer AD Cuentas
    Fin Mientras
    #Cierre del Agregado de Datos
    Cerrar AD Cuentas
    #Se imprime el total obtenido
    Imprimir "Total Cuentas Deudoras .... $ " ,Total
Fin Procedimiento
```

- Saldo total en Cuenta Corriente

En este caso no se requiere un listado, sino que se muestra la información en pantalla. Puede ser en un cuadro de mensaje. Observe el siguiente procedimiento.

```

Procedimiento TotalYPromedio()
  Variable Numérica Total = 0
  Variable Numérica Cantidad = 0
  Variable Numérica Promedio = 0
  Abrir AD Cuentas
  Leer AD Cuentas
  Mientras ( NOT Cuentas.EOF )
    # Acumula el total adeudado en la variable Total
    Total = Total + Cuenta.Saldo
    # Calcula la cantidad de cuentas en la variable Cantidad
    Cantidad = Cantidad + 1
    Leer AD Cuentas
  Fin Mientras
  Cerrar AD Cuentas
  # Calcula el promedio
  Promedio = Total / Cantidad
  Mensaje
    Titulo: "Total Adeudado en Cuenta Corriente"
    Icono: Información
    Texto: "El total adeudado es de $" & Total &
           "Con un promedio de deuda por cliente de $" & Promedio
  Fin Mensaje
Fin Procedimiento

```

- **Programa completo**

A continuación se desarrolla el programa completo, con definición de datos, definición de las interfaces y desarrollo de los procedimientos de eventos y procedimientos definidos por el programador.

```

Programa ReportesCtasCtes
    # Definición de Datos Globales
    # Definición del Registro
Registro Cuenta
    Variable Numérica      CodCta
    Variable Alfanumérica Nombre
    Variable Numérica      Saldo
Fin Registro
# Definición del Agregado de Datos
Agregado de Datos Cuentas Tipo Registro Cuenta
# Definición Interfaz de Usuario
Interfaz ReportesCtasCtes
    Formulario frmReportesCtaCte
        Botón de Opción optListadoGeneral
        Botón de Opción optMasDe1000
        Botón de Opción optCuentasDeudoras
        Botón de Opción optSaldoTotal
        Botón de Comando cmdReporte
    Fin Formulario
# Desarrollo de los Procedimientos de Evento
Procedimiento cmdReporte:Click
    Según Sea
        Cuando optListadoGeneral = VERDADERO
            Ejecutar Procedimiento ListadoGeneral
        Cuando optMasDe1000 = VERDADERO
            Ejecutar Procedimiento MayoresMil
        Cuando optCuentasDeudoras = VERDADERO
            Ejecutar Procedimiento CuentasDeudoras
        Cuando optSaldoTotal = VERDADERO
            Ejecutar Procedimiento TotalYPromedio
    Fin Según Sea
    Fin Procedimiento
Fin Interfaz
# Desarrollo de los Procedimientos definidos por el Programador
Procedimiento ListadoGeneral ( )
    Abrir AD Cuentas
    Leer AD Cuentas
    Imprimir "Listado General De Cuentas Corrientes"
    Imprimir Salto de Línea
    Imprimir Salto de Línea
    Imprimir "Código     Nombre     Saldo"
    Imprimir Salto de Línea
    Mientras ( NOT Cuentas.EOF )
        Imprimir Cuenta.CodCta, Cuenta.Nombre, Cuenta.Saldo, Salto de
    Línea
        Leer AD Cuentas
    Fin Mientras
    Cerrar AD Cuentas
Fin Procedimiento

```

```

Procedimiento MayoresMil ( )
  Abrir AD Cuentas
  Leer AD Cuentas
  Imprimir "Listado Cuentas con más de 1000 de Saldo"
  Imprimir Salto de Línea
  Imprimir Salto de Línea
  Imprimir "Código Nombre Saldo"
  Imprimir Salto de Línea
  Mientras ( NOT Cuentas.EOF )
    Si ( Cuenta.Saldo > 1000 )
      Imprimir Cuenta.CodCta , Cuenta.Nombre , Cuenta.Saldo
      Imprimir Salto de Línea
    Fin Si
    Leer AD Cuentas
  Fin Mientras
  Cerrar AD Cuentas
Fin Procedimiento

Procedimiento CuentasDeudoras ( )
  Variable Numérica Total = 0
  Abrir AD Cuentas
  Leer AD Cuentas
  Imprimir "Listado De Cuentas Corrientes Deudoras"
  Imprimir Salto de Línea
  Imprimir Salto de Línea
  Imprimir "Código Nombre Saldo"
  Imprimir Salto de Línea
  Mientras ( NOT Cuentas.EOF )
    Si ( Cuenta.Saldo > 0 )
      Imprimir Cuenta.CodCta , Cuenta.Nombre , Cuenta.Saldo
      Imprimir Salto de Línea
      Total = Total + Cuenta.Saldo
    Fin Si
    Leer AD Cuentas
  Fin Mientras
  Cerrar AD Cuentas
  Imprimir "Total Cuentas Deudoras .... $ " Total
  Imprimir Salto de Línea
Fin Procedimiento

```

```

Procedimiento TotalYPromedio( )
  Variable Numérica Total = 0
  Variable Numérica Cantidad = 0
  Variable Numérica Promedio = 0
  Abrir AD Cuentas
  Leer AD Cuentas
  Mientras ( NOT Cuentas.EOF )
    # Acumula el total adeudado en la variable Total
    Total = Total + Cuenta.Saldo
    # Calcula la cantidad de cuentas en la variable Cantidad
    Cantidad = Cantidad + 1
    Leer AD Cuentas
  Fin Mientras
  Cerrar AD Cuentas
  # Calcula el promedio
  Promedio = Total / Cantidad
  Mensaje
    Título: "Total Adeudado en Cuenta Corriente"
    Icono: Información
    Texto: "El total adeudado es de $ " & Total &
           "Con un promedio de deuda por cliente de $ " & Promedio
  Fin Mensaje
  Fin Procedimiento
Fin Programa

```

# SP1 / Ejercicio por resolver

El COMFER, encargado de controlar los programas de televisión, requiere un informe detallado de la cantidad de audiencia que tiene cada programa.

Dicho informe deberá mostrar por cada canal:

- Cantidad de audiencia por cada programa y tipo de programa.
- Cantidad de audiencia del canal.
- También se deberán mostrar los totales generales de audiencia detallado por tipo de programa.

El diseño del informe debe respetar las características mostradas en el siguiente ejemplo:

Diseño del informe				PL2
Listado General de Audiencia por Canal y Programa				
Canal: 8	Programa	Tipo	Audiencia	
	Hola Susana	Entretenimiento	20.000	
	El Chavo	Entretenimiento	5.000	
	Teleocio Noticias	Informativo	35.000	
	<b>Total de Audiencia:</b>		<b>60.000</b>	
Canal: 10	Programa	Tipo	Audiencia	
	Por Amor	Entretenimiento	6.000	
	El Precio Justo	Entretenimiento	7.500	
	Venite	Interés General	8.900	
	Crónica 10	Informativo	28.050	
	<b>Total de Audiencia:</b>		<b>54.450</b>	
Canal: 12	Programa	Tipo	Audiencia	
	Noticiero Doce	Informativo	31.600	
	Hora Cero	Interés General	1.000	
	<b>Total de Audiencia:</b>		<b>32.600</b>	
<b>Total General de Audiencia:</b>				<b>143.050</b>
<b>Audiencia de Programas de Entretenimiento:</b>				<b>38.500</b>
<b>Audiencia de Programas de Interés General:</b>				<b>9.900</b>
<b>Audiencia de programas Informativos:</b>				<b>94.650</b>

## Listado General de Audiencia por Canal y Programa

Cantidad de audiencia por cada programa y tipo de programa	Programa	Tipo	Audiencia
	Hola Susana	Entretenimiento	20.000
	El Chavo	Entretenimiento	5.000
	Teleocho Noticias	Informativo	35.000
	<b>Total de Audiencia:</b>		<b>60.000</b>
Canal: 12	Programa	Tipo	Audiencia
	Por Amor	Entretenimiento	6.000
	El Precio Justo	Entretenimiento	7.500
	Venite	Interés General	8.900
	Crónica 10	Informativo	28.050
	<b>Total de Audiencia:</b>		<b>54.450</b>
	<b>Total General de Audiencia:</b>		<b>143.050</b>
	<b>Audiencia de Programas de Entretenimiento:</b>		<b>38.500</b>
	<b>Audiencia de Programas de Interés General:</b>		<b>9.900</b>
	<b>Audiencia de programas Informativos:</b>		<b>94.650</b>

Totales generales de audiencia detallados por tipo de programa

Además, deberá tener en cuenta que:

- El total de audiencia se debe calcular por canal.
- Los totales de audiencia por tipo de programa, se calculan en forma general (sin importar el canal).
- En caso que el informe ocupe más de una página, cada hoja tiene la capacidad de imprimir como máximo 72 líneas.

Para realizar dicho informe es necesario trabajar con el siguiente agregado de datos:

## AD Audiencia

Canal	Programa	Tipo	Audiencia
8	Hola Susana	E	20000
8	El Chavo	E	5000
8	Teleocho Noticias	G	35000
10	Por Amor	E	6000
10	El Precio Justo	E	7500
10	Venite	G	8900
10	Crónica 10	I	28050
12	Noticiero Doce	I	31600
12	Hora Cero	G	1000

EOF

**Canal:** Número que identifica a cada canal**Programa:** Nombre del programa televisivo.**Tipo:** Clasificación del Programa**E** = Entretenimiento**I** = Informativo**G** = Interés General**Audiencia:** Cantidad de personas que miran el programa.



¿Te animás a medir cuánto aprendiste?

**1. Indique la opción correcta**

Gracias al índice de un agregado de datos, podemos acceder directamente a cualquier posición del mismo.

- Verdadero
- Falso

**2. Indique la opción correcta**

Cuando se han leído todos los registros del agregado de datos se llega al final del archivo, conocido como EOF.

- Verdadero
- Falso

**3. Indique la opción correcta**

La siguiente declaración corresponde a la de un vector de registros de tipo cliente.

**Estructura Vector Cliente Clientes[10]**

- Verdadero

- Falso

**4. Indique la opción correcta**

La siguiente fracción de código de un procedimiento, recorre todo el AD Clientes y muestra el nombre de los clientes contenidos en cada registro.

## Procedimiento Nn

```
.....  
Abrir AD Clientes  
Leer AD Clientes  
Mientras ( NOT Clientes.EOF)  
    Imprimir Cliente.Nombre  
    Leer AD Clientes  
Fin Mientras  
Cerrar AD Clientes  
Fin Procedimiento
```

Verdadero

Falso

### 5. Indique la opción correcta

En un AD, la clave es un campo que contiene un valor que no se repite.

Verdadero

Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Gracias al índice de un agregado de datos, podemos acceder directamente a cualquier posición del mismo.

- Verdadero
- Falso

## 2. Indique la opción correcta

Cuando se han leído todos los registros del agregado de datos se llega al final del archivo, conocido como EOF.

- Verdadero
- Falso

## 3. Indique la opción correcta

La siguiente declaración corresponde a la de un vector de registros de tipo cliente.

- Verdadero
- Falso

## 4. Indique la opción correcta

La siguiente fracción de código de un procedimiento, recorre todo el AD Clientes y muestra el nombre de los clientes contenidos en cada registro

- Verdadero
- Falso

## 5. Indique la opción correcta

En un AD, la clave es un campo que contiene un valor que no se repite.

- Verdadero
- Falso

## Situación profesional 2: Frigorífico CARCOR S.A.

El frigorífico CARCOR S.A. faena ganado que se compra en distintas ferias y que puede estar destinado a consumo o exportación.

Cuenta con un sistema informático que se utiliza para registrar todas las actividades comerciales que se realizan y emitir diferentes tipos de reportes. Uno de los agregados de datos del sistema almacena los kilos comprados a cada feria y el importe pagado por kilo.

El agregado tiene la siguiente estructura:

AD: Compras						
Código	Nombre	Destino	Fecha	Kilos	Precio	
1011	San Salvador	Consumo	22/01/15	5000	13,30	
1011	San Salvador	Consumo	02/02/15	3000	11,00	
1011	San Salvador	Consumo	15/03/15	6000	10,50	
1011	San Salvador	Exportación	12/01/15	1200	11,70	
1011	San Salvador	Exportación	07/02/15	7000	09,70	
1047	Romanutti S.A.	Consumo	10/03/15	1800	08,90	
1047	Romanutti S.A.	Exportación	07/02/15	1000	09,30	
1047	Romanutti S.A.	Exportación	23/03/15	2000	11,20	
EOF						

Se le solicita que desarrolle el programa en pseudocódigo que emita un reporte que refleje la siguiente información:

- a) Kilogramos comprados a cada feria y total pagado.
- b) Total de kilogramos destinados a consumo por feria y total pagado.
- c) Total de kilogramos destinados exportación y total pagado. También por feria.
- d) Total de kilogramos comprados a cada feria y total pagado.
- e) Total general de kilogramos comprados y total pagado.

El reporte solicitado debe respetar el modelo siguiente:

*Informe detallado de compras realizadas por feria*

**Feria: 1011 - San Salvador**

Destino	Fecha	Kilos	Importe
Consumo	22/05/15	5000	\$66.500,00
	02/02/15	3000	\$33.000,00
	15/03/15	6000	\$63.000,00
	Total:	14000	\$162.500,00
Exportación	12/01/15	1200	\$14.040,00
	07/02/15	7000	\$67.900,00
	Total:	8200	\$81.940,00
Total correspondiente a la feria:		22200	\$244.440,00
<b>Feria: 1047 - Romanutti S.A.</b>			
Destino	Fecha	Kilos	Importe
Consumo	10/03/15	1800	\$16020,00
	Total:	1800	\$16020,00
Exportación	07/02/15	1000	\$9.300,00
	23/03/15	2000	\$22.400,00
	Total:	3000	\$31.700,00
Total correspondiente a la feria:		4800	\$47.720,00
<b>Total General:</b>		<b>27000 \$292.160,00</b>	

# SP2 / H1: Cortes de control y reportes

Presentaremos esta herramienta de la siguiente manera:

- a) Tratamiento de datos ordenados
- b) Diagramas de Warnier
- c) Modelos de reportes y salidas impresas

## a) Tratamiento de datos ordenados

Cuando se procesan agregados de datos es fundamental interpretar correctamente el problema y analizar los datos con los que se cuenta. Es de suma importancia considerar el criterio con el que se ordenaron los registros. Usted se preguntará por qué, pues:

*El orden de los datos determina la estructura del algoritmo*

Muchas veces sucede que, en el conjunto de registros, existen datos que se repiten. Por ejemplo, si se cuenta con un agregado de datos que almacena los movimientos de las cuentas corrientes de una entidad bancaria compuesto por registros que almacenan: el número de la sucursal, la fecha del movimiento, el número de la cuenta corriente, el tipo de movimiento (depósito o extracción) y el importe, deberá tener en cuenta lo siguiente:

- Existen muchos movimientos por sucursal. Esto indica que habrá muchos registros con el mismo número de sucursal.
- Existen muchos movimientos por fecha. Por lo que habrá muchos registros con la misma fecha.
- Existen muchos movimientos para una misma cuenta. Por lo que se verán varios registros con el mismo número de cuenta.
- Existen muchos depósitos y muchas extracciones. Por consiguiente se encontrarán muchos registros con el mismo tipo de movimiento.

## Movimiento de las cuentas corrientes ➔ PL2

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
1	01/02/15	0001	E	0032658	3000,50
1	01/02/15	0015	D	0000031	5000,00
1	02/02/15	0001	D	0000032	6000,00
1	02/02/15	0001	D	0000034	5000,00
1	02/02/15	0001	E	0951001	2500,00
1	02/02/15	0015	D	0000033	11000,00
1	02/02/15	0015	E	0236590	3200,00
1	02/02/15	0015	E	0562308	5300
2	01/02/15	0010	D	0000050	6000,00
2	01/02/15	0010	D	0000052	5000,00
2	02/02/15	0030	D	0000051	3500,00
2	02/02/15	0030	E	0659821	6000,00
2	02/02/15	0030	E	0985421	1500,00

Al igual que este ejemplo en nuestra Situación profesional tenemos muchas actividades por feria, en distintas fechas y con distintos destinos.

## Actividades de la SP ➔ PL2

### AD: Compras

Código	Nombre	Destino	Fecha	Kilos	Precio
1011	San Salvador	Consumo	22/01/15	5000	13,30
1011	San Salvador	Consumo	02/02/15	3000	11,00
1011	San Salvador	Consumo	15/03/15	6000	10,50
1011	San Salvador	Exportación	12/01/15	1200	11,70
1011	San Salvador	Exportación	07/02/15	7000	09,70
1047	Romanutti S.A.	Consumo	10/03/15	1800	08,90
1047	Romanutti S.A.	Exportación	07/02/15	1000	09,30
1047	Romanutti S.A.	Exportación	23/03/15	2000	11,20

EOF

Considere que se le solicita un programa que imprima en un reporte la cantidad de movimientos almacenados, el importe total de todos los depósitos y el importe total de todas las extracciones.

Seguramente usted tiene la respuesta, y es que se debe recorrer todo el agregado de datos y por cada registro que se lee se incrementa un contador y dependiendo del tipo de movimiento se incremente uno de los acumuladores necesarios:

```

Procedimiento cmdImprimirReporte:Click
  Variable numérica Cantidad = 0    # Inicializo la variable que se utilizará como contador
  Variable numérica Depósito = 0    # Inicializo la variable que se utilizará como acumulador
  Variable numérica Extracción = 0 # Inicializo la variable que se utilizará como acumulador
  Abrir AD movimientos
  Leer AD movimientos
  Imprimir "Informe de los movimientos realizados" Salto de línea
  Mientras ( NOT movimientos.EOF )
    Si ( movimientos.tipo = "D")
      # Se acumulan los depósitos
      Depósito = Depósito + movimiento.importe
    Si no
      # Se acumulan las extracciones
      Extracción = Extracción + movimientos.importe
    fin si
    # Se incrementa el contador
    Cantidad = Cantidad + 1
    Leer AD movimientos
  Fin mientras
  # Se imprimen los resultados
  Imprimir "Cantidad de movimiento:" Cantidad Salto de línea
  Imprimir "Total Depositado:" Depósito Salto de línea
  Imprimir "Total Extraido:" Extracción Salto de línea
  Cerrar AD movimientos
Fin procedimiento

```

Observe el manejo de las variables en el algoritmo. Se presentan claramente tres instancias:

- **Inicialización de variables**

```

Procedimiento cmdImprimirReporte:Click
  Variable numérica Cantidad = 0    # Inicializo la variable que se utilizará como contador
  Variable numérica Depósito = 0    # Inicializo la variable que se utilizará como acumulador
  Variable numérica Extracción = 0 # Inicializo la variable que se utilizará como acumulador

  Inicialización de variables: Es el momento en que la variable toma un valor inicial.
  Los contadores y acumuladores generalmente se inicializan en cero. En este programa
  coincide la inicialización de las variables con el inicio del programa, pero veremos en otro
  ejemplo que no siempre es así. Siempre se inicializan las variables antes de ingresar a la
  estructura repetitiva donde son modificadas.

  Depósito = Depósito + movimiento.importe
  Si no
    # Se acumulan las extracciones
    Extracción = Extracción + movimientos.importe
  fin si
  # Se incrementa el contador
  Cantidad = Cantidad + 1
  Leer AD movimientos
  Fin mientras
  # Se imprimen los resultados
  Imprimir "Cantidad de movimiento:" Cantidad Salto de línea
  Imprimir "Total Depositado:" Depósito Salto de linea
  Imprimir "Total Extraido:" Extracción Salto de linea
  Cerrar AD movimientos
Fin procedimiento

```

- **Generación de resultados**

```

Procedimiento cmdImprimirReporte:Click
    Variable numérica Cantidad = 0      # Inicializo la variable que se utilizará como contador
    Variable numérica Depósito = 0      # Inicializo la variable que se utilizará como acumulador
    Variable numérica Extracción = 0    # Inicializo la variable que se utilizará como acumulador

Generación de resultados: Es el momento en el que las variables sufren cambios.
Puede existir una estructura repetitiva o no, pero siempre se realizan estos cambios en el
núcleo del programa, dentro de la estructura repetitiva.

Mientras ( NOT movimientos.EOF )
    Si ( movimientos.tipo = "D")
        # Se acumulan los depósitos
        Depósito = Depósito + movimiento.importe
    Si no
        # Se acumulan las extracciones
        Extracción = Extracción + movimientos.importe
    fin si
    # Se incrementa el contador
    Cantidad = Cantidad + 1
    Leer AD movimientos
Fin mientras
# Se imprimen los resultados
Imprimir "Cantidad de movimiento:" Cantidad Salto de línea
Imprimir "Total Depositado:" Depósito Salto de línea
Imprimir "Total Extraído:" Extracción Salto de línea
Cerrar AD movimientos
Fin procedimiento

```

- **Salida de datos**

```

Procedimiento cmdImprimirReporte:Click
    Variable numérica Cantidad = 0      # Inicializo la variable que se utilizará como contador
    Variable numérica Depósito = 0      # Inicializo la variable que se utilizará como acumulador
    Variable numérica Extracción = 0    # Inicializo la variable que se utilizará como acumulador
    Abrir AD movimientos
    Leer AD movimientos
    Imprimir "Informe de los movimientos realizados" Salto de línea
    Mientras ( NOT movimientos.EOF )
        Si ( movimientos.tipo = "D")
            # Se acumulan los depósitos
            Depósito = Depósito + movimiento.importe
        Si no
            # Se acumulan las extracciones
            Extracción = Extracción + movimientos.importe
        fin si
        # Se incrementa el contador
    fin mientras
    # Se imprimen los resultados
    Imprimir "Cantidad de movimiento:" Cantidad Salto de línea
    Imprimir "Total Depositado:" Depósito Salto de línea
    Imprimir "Total Extraído:" Extracción Salto de línea
    Cerrar AD movimientos
Fin procedimiento

```

**Salida de datos:** Es el momento en que se muestra el resultado obtenido en el programa,
ya sea en pantalla o en impresora. **Siempre se realiza al salir de la estructura repetitiva.**

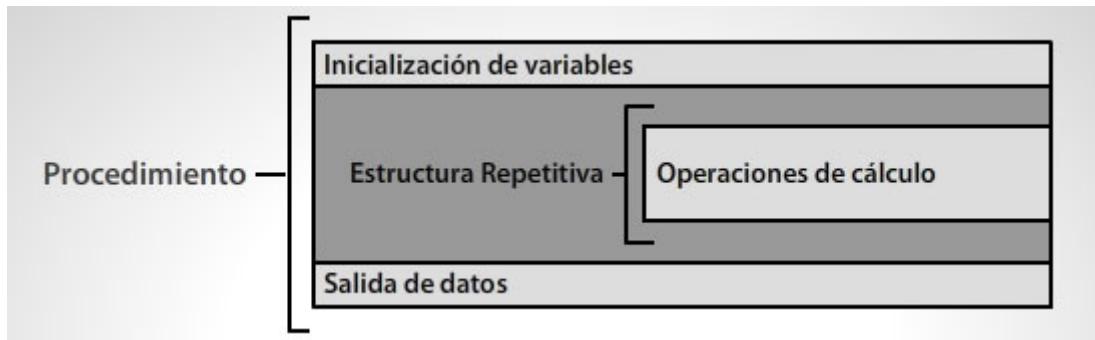
## b) Diagramas de Warnier

Los diagramas de Warnier son también conocidos como construcción lógica de programas/construcción lógica de sistemas. Este método ayuda al diseño de estructuras de programas identificando la salida y resultado del

procedimiento. Trabaja hacia atrás para determinar los pasos y combinaciones de entrada necesarios para producirlos. Los sencillos métodos gráficos usados en los diagramas de Warnier hacen evidentes los niveles en un sistema y más claros los movimientos de los datos en dichos niveles.

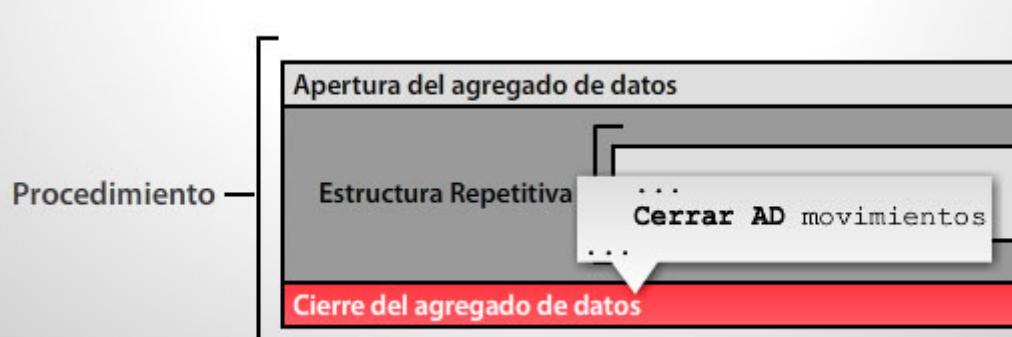
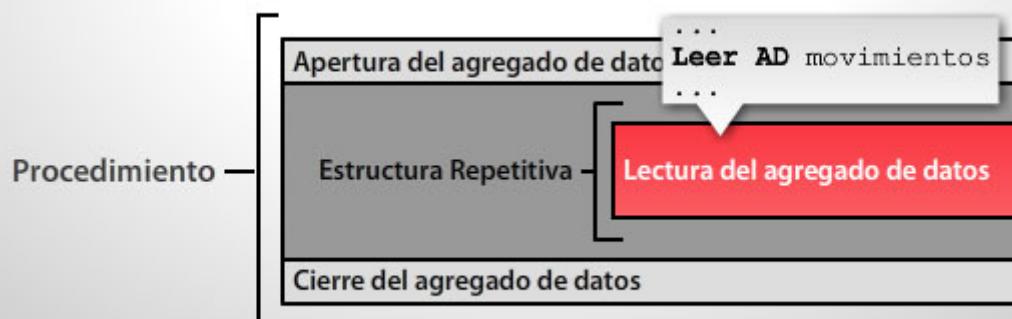
Estos diagramas muestran los procesos y la secuencia en que se realizan. Cada proceso se define de una manera jerárquica, es decir, consta de conjuntos de subprocesos que lo definen, en cada nivel. El proceso se muestra en una llave que agrupa a sus componentes. Puesto que un proceso puede tener muchos subprocesos distintos, un diagrama de Warnier usa un conjunto de llaves para mostrar cada nivel del sistema.

Observe en el siguiente esquema la representación de estas tres instancias:



Observe que estas tres instancias coinciden, además, en ubicación con respecto a la estructura repetitiva, con los de apertura, lectura y cierre del agregado de datos:





Todas estas instancias deben ser muy tenidas en cuenta cuando se trabaja en un caso más complejo. Considere que se necesita realizar un programa que emita un reporte de la cantidad total de movimientos realizados, el importe depositado y el extraído en cada sucursal de la entidad bancaria y, como cierre del informe, la cantidad y totales generales.

Deberá considerar que, el agregado de datos (que no es más que un conjunto de registros), se puede separar claramente en subconjuntos. El agregado de datos de nuestra Situación profesional está conformado precisamente por dos subconjuntos:



**El conjunto de los registros correspondientes a la sucursal 1**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
1	01/02/15	0001	E	0032658	3000,50
1	01/02/15	0015	D	0000031	5000,00
1	02/02/15	0001	D	0000032	6000,00
1	02/02/15	0001	D	0000034	5000,00
1	02/02/15	0001	E	0951001	2500,00
1	02/02/15	0015	D	0000033	11000,00
1	02/02/15	0015	E	0236590	3200,00
1	02/02/15	0015	E	0562308	5300

### El conjunto de los registros correspondientes a la sucursal 2

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
2	01/02/15	0010	D	0000050	6000,00
2	01/02/15	0010	D	0000052	5000,00
2	02/02/15	0030	D	0000051	3500,00
2	02/02/15	0030	E	0659821	6000,00
2	02/02/15	0030	E	0985421	1500,00

Como el agregado de datos está ordenado por número de sucursal, será posible realizar un programa que recorra todos los registros del primer subconjunto a medida que se cuentan las lecturas y, luego, los del segundo subconjunto. Tenga presente que de todos modos el agregado de datos deberá recorrerse plenamente. Esto significa que se deberá llegar al final del agregado de datos.

En el caso de nuestra situación tendremos la misma diferenciación en cuanto a los mercados.

Cuando se trata el conjunto de registros del agregado de datos, una sola estructura repetitiva es suficiente.

```

Mientras ( NOT movimientos.EOF)
    Sucursal = movimientos.sucursal
    Cantidad = 0 # Inicializo la variable que se utilizará como contador
    Depósito = 0 # Inicializo la variable que se utilizará como acumulador
    Extracción = 0 # Inicializo la variable que se utilizará como acumulador

    Mientras ( NOT movimientos.EOF AND movimientos.sucursal = Sucursal)
        Si ( movimientos.tipo = "D")
            Depósito = Depósito + movimiento.importe
        Si no
            Extracción = Extracción + movimiento.importe
        fin si
        Cantidad = Cantidad + 1
        Leer AD movimientos
    Fin mientras

    Imprimir SucursalCantidadDepósitosExtracciones & Salto de línea
    CantidadGral = CantidadGral + Cantidad
    Depósito = Depósito + Extracción
    CantidadGral = CantidadGral + Cantidad
Fin mientras

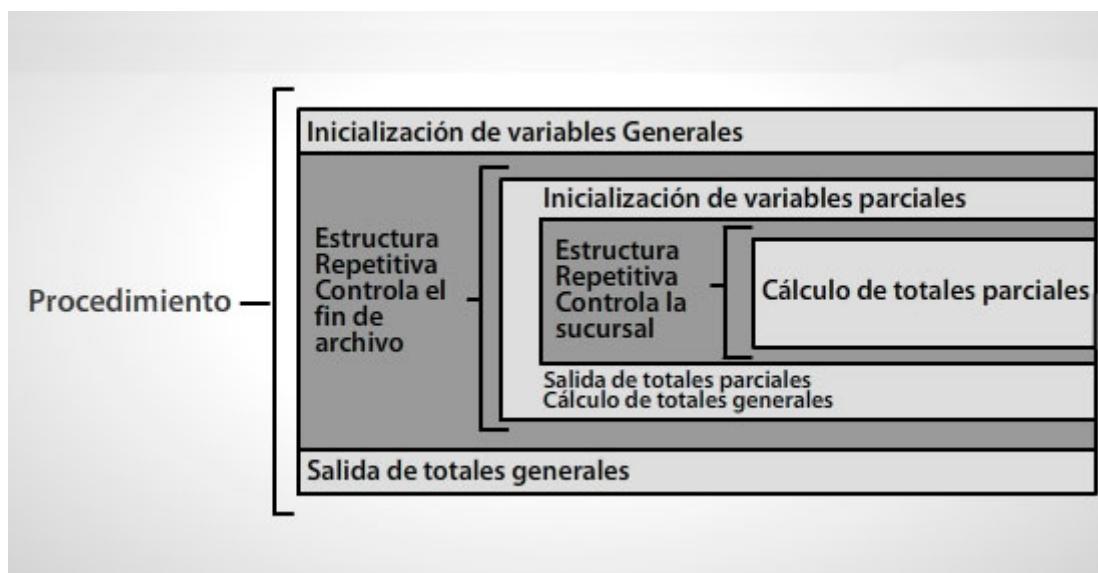
```

Video "[Estructura repetitiva: Subconjuntos](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

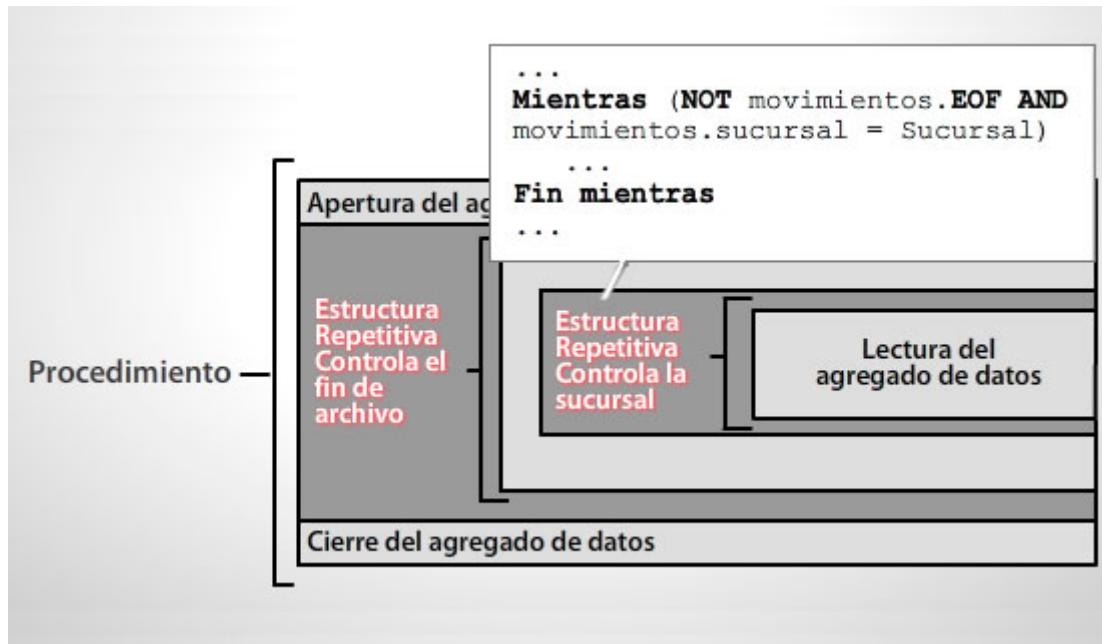
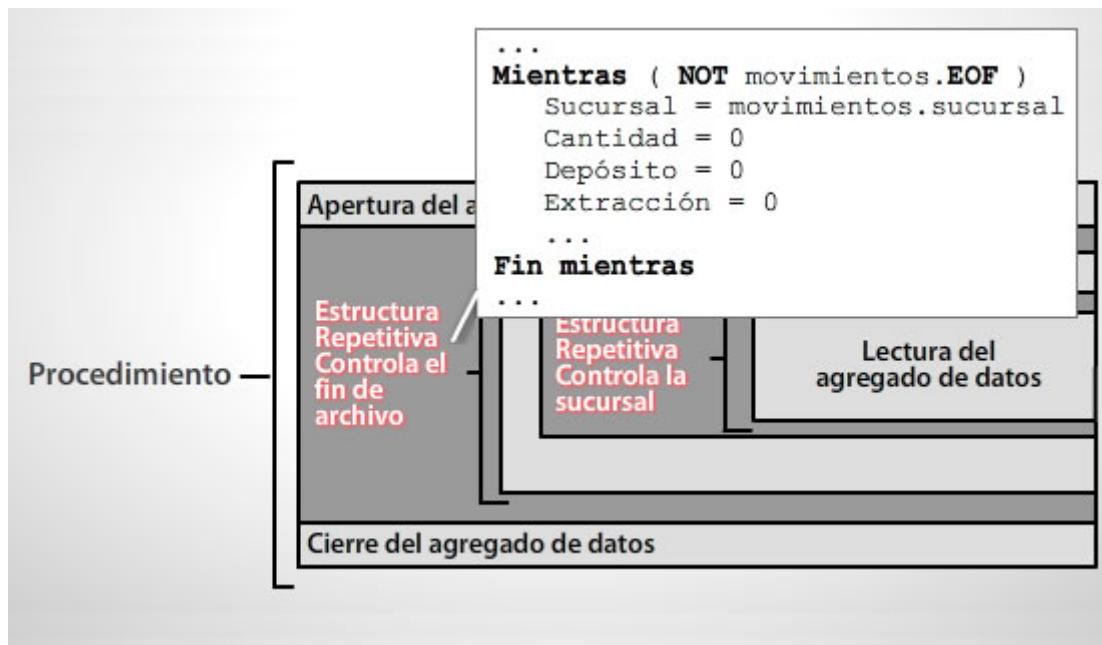
- La primera repetitiva permite recorrer todo el agregado de datos.
- La segunda repetitiva garantiza realizar lecturas y cálculos en la medida que correspondan a una sucursal. Cuando se cambia de sucursal se deben mostrar los totales.

Este programa parece muy diferente al anterior. Sin embargo, respeta lo dicho con respecto a las instancias de inicialización, cálculo y salida de resultados. La única diferencia es que se utilizaron dos estructuras repetitivas, que están una dentro de la otra. Veamos el esquema para lograr una mejor comprensión de lo planteado:

- **Esquema:** Se utilizaron dos estructuras repetitivas, que están una dentro de la otra



- **Ejemplo:** Con respecto a las operaciones con el agregado de datos notará que la apertura y cierre se mantienen al principio y fin del procedimiento respectivamente. La lectura está en la parte interna del mismo.



### c) Modelos de Reportes y Salidas impresas

¿Qué se debería tener en cuenta si se necesita realizar un programa que emita un reporte que detalle lo siguiente?

- Total general depositado y total general extraído en el banco.
- Total depositado y total extraído por sucursal.

- Total depositado y total extraído por fecha.
- Total depositado y total extraído por cuenta.

Lo primero a considerar es la información que se desea obtener con el funcionamiento del programa. A esto se lo denomina: *análisis de la salida de datos*. Cuando el programa sólo requiere un resultado, como se vio en el primer ejercicio no es necesario, pero cuando son muchos los cálculos que intervienen es fundamental determinar cuántos y cuáles son los resultados que se espera obtener.

Para esto vamos a trabajar con un modelo de ejemplo de la salida de datos:

**Modelo de Reportes y Salidas impresas** PL2

<b>Informe diario de los movimientos</b>					
Sucursal	Día	Cuenta	Depositado	Extraído	
1	01/02/15	1	15000,23	3000,50	→ Acumulador de extracciones por cuenta
		15	5000,00	0,00	→ Acumulador de depósitos por cuenta
		<b>Total del día</b>	<b>20000,23</b>	<b>3000,50</b>	→ Acumulador de extracciones por día → Acumulador de depósitos por día
	02/02/15	1	11000,00	2500,00	
		15	11000,00	8500,00	
		<b>Total del día</b>	<b>22000,00</b>	<b>11000,00</b>	
<b>Total de la Sucursal</b>			<b>42000,23</b>	<b>14000,50</b>	→ Acumulador de extracciones por sucursal → Acumulador de depósitos por sucursal
2	01/02/15	10	11000,00	0,00	
		<b>Total del día</b>	<b>11000,00</b>	<b>0,00</b>	
	02/02/15	30	3500,00	7500,00	
		<b>Total del día</b>	<b>3500,00</b>	<b>7500,00</b>	
<b>Total de la Sucursal</b>			<b>14500,00</b>	<b>7500,00</b>	
<b>Total General</b>			<b>56500,23</b>	<b>21500,50</b>	→ Acumulador general de extracciones → Acumulador general de depósitos

En el caso de nuestra Situación profesional, lo que queremos obtener es el siguiente reporte:

**Informe detallado de compras realizadas por feria****Feria: 1011 - San Salvador**

Destino	Fecha	Kilos	Importe	
Consumo	22/05/15	5000	\$66.500,00	Importe de la compra
	02/02/15	3000	\$33.000,00	
	15/03/15	6000	\$63.000,00	
	Total:	14000	\$162.500,00	
Exportación	12/01/15	1200	\$14.040,00	Suma de los kilos por destino
	07/02/15	7000	\$67.900,00	
	Total:	8200	\$81.940,00	Suma de los importes por destino
<b>Total correspondiente a la feria:</b>		<b>22200</b>	<b>\$244.440,00</b>	Suma de los importes por feria
				Suma de los kilos por feria
<b>Feria: 1047 - Romanutti S.A.</b>				
Destino	Fecha	Kilos	Importe	
Consumo	10/03/15	1800	\$16020,00	
	Total:	1800	\$16020,00	
Exportación	07/02/15	1000	\$9.300,00	
	23/03/15	2000	\$22.400,00	
	Total:	3000	\$31.700,00	
<b>Total correspondiente a la feria:</b>		<b>4800</b>	<b>\$47.720,00</b>	
<b>Total General:</b>		<b>27000</b>	<b>\$292.160,00</b>	Suma general de los importes Suma general de kilos

Deberá considerar, además, que el agregado de datos se puede separar en más subconjuntos que los analizados hasta el momento. Es más, estos subconjuntos están incluidos en los ya definidos tal como vemos en el esquema siguiente:

**AD movimientos****Subconjunto 1**

1 01/02/15 0001  
D 0000030 15000,23  
  
 1 01/02/15 0001 1 01/02/15 00015  
E 0032658 3000,50 D 0000031 5000,00  
  
 1 02/02/15 0001  
D 0000032 6000,00  
  
 1 02/02/15 0001 1 02/02/15 0001  
E 0951001 2500,00 D 0000034 5000,00  
  
 1 02/02/15 0015 1 02/02/15 0015  
E 0236590 3200,00 D 0000033 11000,00  
  
 1 02/02/15 0015  
E 0562308 5300

**Subconjunto 2**

2 01/02/15 0010  
D 0000050 6000,00  
  
 2 01/02/15 0010  
D 0000052 5000,00  
  
 2 02/02/15 0010  
E 0659821 6000,00  
  
 2 02/02/15 0010 2 02/02/15 0010  
D 0000051 3500,00 E 0985421 1500,00

**El conjunto de los registros correspondientes a la sucursal 1**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
1	01/02/15	0001	E	0032658	3000,50
1	01/02/15	0015	D	0000031	5000,00
1	02/02/15	0001	D	0000032	6000,00
1	02/02/15	0001	D	0000034	5000,00
1	02/02/15	0001	E	0951001	2500,00
1	02/02/15	0015	D	0000033	11000,00
1	02/02/15	0015	E	0236590	3200,00
1	02/02/15	0015	E	0562308	5300

**El conjunto de los registros correspondientes a la sucursal 2**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
2	01/02/15	0010	D	0000050	6000,00
2	01/02/15	0010	D	0000052	5000,00
2	02/02/15	0030	D	0000051	3500,00
2	02/02/15	0030	E	0659821	6000,00
2	02/02/15	0030	E	0985421	1500,00

Movimientos de la **sucursal 1** del **día 01/02/15**

**Conjunto de movimientos de la sucursal 1 correspondientes al día 01/02/15**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
1	01/02/15	0001	E	0032658	3000,50
1	01/02/15	0015	D	0000031	5000,00

Movimientos de la **sucursal 1** del **día 02/02/15**

**Conjunto de movimientos de la sucursal 1 correspondientes al día 02/02/15**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	02/02/15	0001	D	0000032	6000,00
1	02/02/15	0001	D	0000034	5000,00
1	02/02/15	0001	E	0951001	2500,00
1	02/02/15	0015	D	0000033	11000,00
1	02/02/15	0015	E	0236590	3200,00
1	02/02/15	0015	E	0562308	5300

Movimientos de la **sucursal 2** del **día 01/02/15**

**Conjunto de movimientos de la sucursal 2 correspondientes al día 01/02/15**

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
2	01/02/15	0010	D	0000050	6000,00
2	01/02/15	0010	D	0000052	5000,00

Conjunto de movimientos de la sucursal 2 correspondientes al día 02/02/15						
Sucursal	Fecha	Cuenta	Tipo	Número	Importe	
2	02/02/15	0030	D	0000051	3500,00	
2	02/02/15	0030	E	0659821	6000,00	
2	02/02/15	0030	E	0985421	1500,00	

Cada subconjunto que se trate requiere de una estructura repetitiva diferente. Tenga en cuenta lo siguiente:

- El agregado de datos tiene **muchas** sucursales.
- Cada sucursal tiene movimientos en **muchas** fechas.
- Por cada fecha existen movimientos de **muchas** cuentas.
- Por cada cuenta existen **muchos** movimientos.
- Cada movimiento tiene **un** único número e importe.

Lo descrito se puede entender mejor analizando el siguiente esquema:

## Esquema: Estructura repetitiva diferente

PL2



- 1- El agregado de datos tiene **muchas** sucursales
- 2- Cada sucursal tiene movimientos en **muchas** fechas
- 3- Por cada fecha existen movimientos de **muchas** cuentas
- 4- Por cada cuenta existen **muchos** movimientos
- 5- Cada movimiento tiene **un** único número e importe

Observe que este esquema presenta cuatro llaves, que están representando los subconjuntos, más adelante podrá verificar que cada llave corresponde a una estructura repetitiva.

Todo lo representado en el esquema anterior se verifica perfectamente analizando el agregado de datos que tenemos de ejemplo.

## Representación: Estructura repetitiva diferente

PL2

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
			E	0032658	3000,50
		0015	D	0000031	5000,00
	02/02/15	0001	D	0000032	6000,00
				0000034	5000,00
		0015	E	0951001	2500,00
			D	0000033	11000,00
			E	0236590	3200,00
				0562308	5300
2	01/02/15	0010	D	0000050	6000,00
	0000052	5000,00			
	02/02/15	0030	D	0000051	3500,00
			E	0659821	6000,00
				0985421	1500,00

No deje de observar el modo en el que están ordenados los datos ya que, el ordenamiento, está respetando la secuencia mencionada:

1º criterio de orden

2º criterio de orden

3º criterio de orden

4º criterio de orden



Interacción

## Representación: Estructura repetitiva diferente

PL2

Sucursal	1º criterio de orden			Importe
1	mc cuadro 1	Los registros están ordenados por número de sucursal. Esto significa que primero se leen los de la sucursal número 1, y después los de la sucursal número 2.		
		01/02/15	0001	15000,23
			0015	3000,50
		02/02/15	0010	5000,00
			0030	6000,00
			0010	5000,00
			0030	6000,00
2	mc cuadro 1	01/02/15	0010	6000,00
			0030	5000,00
		02/02/15	0010	3500,00
			0030	6000,00
			0030	1500,00

No deje de observar el modo en el que están ordenados los datos ya que, el ordenamiento, está respetando la secuencia mencionada:

1º criterio de orden

2º criterio de orden

3º criterio de orden

4º criterio de orden



Interacción

## Representación: Estructura repetitiva diferente

PL2

Sucursal	Fecha	2º criterio de orden			
1	01/02/15	A su vez, existe un segundo criterio de orden, los registros están ordenados por fecha.			
	02/02/15	Pero como este no es el criterio de orden principal se puede observar el ordenamiento en forma fraccionada. Esto significa que todos los registros de una sucursal estarán juntos y ordenados por fecha. A continuación los de la siguiente sucursal ordenados también por fecha.			
2	01/02/15	0010	D	0000050	6000,00
				0000052	5000,00
	02/02/15		D	0000051	3500,00
		0030	E	0659821	6000,00
				0985421	1500,00

No deje de observar el modo en el que están ordenados los datos ya que, el ordenamiento, está respetando la secuencia mencionada:

1º criterio de orden

2º criterio de orden

3º criterio de orden

4º criterio de orden



Interacción

## Representación: Estructura repetitiva diferente

PL2

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
			E	0032658	3000,50
		0015	D	0000031	5000,00
	02/02/15				6000,00
					5000,00
2	3º criterio de orden				2500,00
	Existe un tercer criterio de orden, que agrupa por cada fecha los movimientos de cada cuenta.				1000,00
	Observe en el agregado de datos que los movimientos correspondientes a la sucursal número 1 del día 02/02/02 están ordenados, primero aparecen los de la cuenta 0001 y a continuación los de la cuenta 0015.				3200,00
					5300
					6000,00

No deje de observar el modo en el que están ordenados los datos ya que, el ordenamiento, está respetando la secuencia mencionada:

1º criterio de orden

2º criterio de orden

3º criterio de orden

4º criterio de orden



Interacción

Sucursal	Fecha	Cuenta	Tipo	Número	Importe
1	01/02/15	0001	D	0000030	15000,23
			E	0032658	3000,50
		0015	D	0000031	5000,00
	02/02/15	0001	D	0000032	6000,00
				0000034	5000,00
			E	0951001	2500,00

**4º criterio de orden**

Finalmente, los movimientos están ordenados por un cuarto criterio de orden que agrupa todos los depósitos por un lado y las extracciones por otro. Observe en el agregado de datos que los movimientos de la cuenta 0001 realizados el día 02/02/02 presenta primero los depósitos y luego las extracciones.

No deje de observar el modo en el que están ordenados los datos ya que, el ordenamiento, está respetando la secuencia mencionada:

1º criterio de orden

2º criterio de orden

3º criterio de orden

4º criterio de orden



Interacción

Por esta razón el algoritmo que se realice requerirá de cuatro estructuras repetitivas:

- Una que controle el final del archivo.
- Otra que controle las sucursales.
- Otras que controles las fechas.
- Otras que controlen las cuentas.

Por cada una de las estructuras repetitivas deberá respetarse las tres instancias de:

- Inicialización de variables.
- Operaciones de cálculo.
- Salida de resultados.

#### Pseudocódigo de Estructura repetitiva diferente

```

Procedimiento cmdImprimirReporte:Click
    Variable numérica DepósitoGral = 0      #Inicializo el acumulador general
    Variable numérica ExtracciónGral = 0    #Inicializo el acumulador general
    Variable numérica DepósitoSuc = 0       #Inicializo el acumulador por sucursal
    Variable numérica ExtracciónSuc = 0     #Inicializo el acumulador por sucursal
    Variable numérica DepósitoFec = 0       #Inicializo el acumulador por fecha
    Variable numérica ExtracciónFec = 0     #Inicializo el acumulador por fecha
    Variable numérica DepósitoCue = 0       #Inicializo el acumulador por cuenta
    Variable numérica ExtracciónCue = 0     #Inicializo el acumulador por cuenta

    Variable numérica Sucursal   #Variable auxiliar que se utiliza como control interno
    Variable numérica Fecha     #Variable auxiliar que se utiliza como control interno
    Variable numérica Cuenta    #Variable auxiliar que se utiliza como control interno

    Abrir AD movimientos
    Leer AD movimientos
    Imprimir "Informe diario de los movimientos" Salto de línea
    Imprimir "Sucursal Día Cuenta Depósitos Extracciones" & Salto de línea

    Mientras (NOT movimientos.EOF)
        Sucursal = movimientos.sucursal
        Imprimir Sucursal
        DepósitoSuc = 0
        ExtracciónSuc = 0

        Mientras (NOT movimientos.EOF AND movimientos.sucursal = Sucursal)
            Fecha = movimientos.fecha
            Imprimir Fecha
            DepósitoFec = 0
            ExtracciónFec = 0

            Mientras (NOT movimientos.EOF AND movimientos.fecha = Fecha)
                Cuenta = movimientos.Cuenta
                Imprimir Cuenta
                DepósitoCue = 0
                ExtracciónCue = 0

                Mientras (NOT movimientos.EOF AND movimientos.Cuenta = Cuenta)
                    Si (movimientos.tipo = "D")
                        DepósitoCue = DepósitoCue + movimiento.importe
                    Si no
                        ExtracciónCue = ExtracciónCue + movimientos.importe
                    fin si
                    Leer AD movimientos
                Fin mientras

                Imprimir DepósitoCue ExtracciónCue Salto de Línea
                DepósitoFec = DepósitoFec + DepósitoCue
                ExtracciónFec = ExtracciónFec + ExtracciónCue
            Fin mientras

            Imprimir "Total del dia:" & DepósitoFec & ExtracciónFec & Salto de Línea
            DepósitoSuc = DepósitoSuc + DepósitoFec
            ExtracciónSuc = ExtracciónSuc + ExtracciónFec
        Fin mientras

        Imprimir "Total de la Sucursal:" & DepósitoSuc & ExtracciónSuc & Salto de Línea
        ExtracciónGral = ExtracciónGral + ExtracciónSuc
        DepósitoGral = DepósitoGral + DepósitoSuc
    Fin mientras

    Imprimir "Total General:" DepósitosGral ExtraccionesGral Salto de línea
    Cerrar AD movimientos
Fin procedimiento

```

## **Resaltar estructura repetitiva**

```
Procedimiento cmdImprimirReporte:Click
    Variable numérica DepósitoGral = 0           #Inicializo el acumulador general
    Variable numérica ExtracciónGral = 0          #Inicializo el acumulador general
    Variable numérica DepósitoSuc = 0             #Inicializo el acumulador por sucursal
    Variable numérica ExtracciónSuc = 0            #Inicializo el acumulador por sucursal
    Variable numérica DepósitoFec = 0              #Inicializo el acumulador por fecha
    Variable numérica ExtracciónFec = 0             #Inicializo el acumulador por fecha
    Variable numérica DepósitoCue = 0              #Inicializo el acumulador por cuenta
    Variable numérica ExtracciónCue = 0             #Inicializo el acumulador por cuenta

    Variable numérica Sucursal      #Variable auxiliar que se utiliza como control interno
    Variable numérica Fecha        #Variable auxiliar que se utiliza como control interno
    Variable numérica Cuenta       #Variable auxiliar que se utiliza como control interno

    Abrir AD movimientos
    Leer AD movimientos
    Imprimir "Informe diario de los movimientos" Salto de linea
    Imprimir "Sucursal   Día   Cuenta   Depósitos   Extracciones" & Salto de linea
```

**Estructura repetitiva que controla el final del archivo**

```
Mientras (NOT movimientos.EOF)
    Sucursal = movimientos.sucursal
    Imprimir Sucursal
    DepósitoSuc = 0
    ExtracciónSuc = 0

    Mientras (NOT movimientos.EOF AND movimientos.sucursal = Sucursal)
        Fecha = movimientos.fecha
        Imprimir Fecha
        DepósitoFec = 0
        ExtracciónFec = 0

        Mientras (NOT movimientos.EOF AND movimientos.fecha = Fecha)
            Cuenta = movimientos.Cuenta
            Imprimir Cuenta
            DepósitoCue = 0
            ExtracciónCue = 0

            Mientras (NOT movimientos.EOF AND movimientos.Cuenta = Cuenta)
                Si (movimientos.tipo = "D")
                    DepósitoCue = DepósitoCue + movimiento.importe
                Si no
                    ExtracciónCue = ExtracciónCue + movimientos.importe
                fin si
                Leer AD movimientos
            Fin mientras

            E.R. que controla las cuentas
            Imprimir DepósitoCue ExtracciónCue Salto de Línea
            DepósitoFec = DepósitoFec + DepósitoCue
            ExtracciónFec = ExtracciónFec + ExtracciónCue
        Fin mientras

        Imprimir "Total del día:" & DepósitoFec & ExtracciónFec & Salto de Línea
        DepósitoSuc = DepósitoSuc + DepósitoFec
        ExtracciónSuc = ExtracciónSuc + ExtracciónFec
    Fin mientras

    Imprimir "Total de la Sucursal:" & DepósitoSuc & ExtracciónSuc & Salto de Línea
    ExtracciónGral = ExtracciónGral + ExtracciónSuc
    DepósitoGral = DepósitoGral + DepósitoSuc
Fin mientras

Imprimir "Total General:" DepósitosGral ExtraccionesGral Salto de línea
Cerrar AD movimientos
Fin procedimiento
```

**Resaltar las tres instancias**

```

Procedimiento cmdImprimirReporte:Click
  Variable numérica DepósitoGral = 0           #Inicializo el acumulador general
  Variable numérica ExtracciónGral = 0         #Inicializo el acumulador general
  Variable numérica DepósitoSuc = 0            #Inicializo el acumulador por sucursal
  Variable numérica ExtracciónSuc = 0          #Inicializo el acumulador por sucursal
  Variable numérica DepósitoFec = 0            #Inicializo el acumulador por fecha
  Variable numérica ExtracciónFec = 0          #Inicializo el acumulador por fecha
  Variable numérica DepósitoCue = 0            #Inicializo el acumulador por cuenta
  Variable numérica ExtracciónCue = 0          #Inicializo el acumulador por cuenta

  Variable numérica Sucursal    #Variable auxiliar que se utiliza como control interno
  Variable numérica Fecha      #Variable auxiliar que se utiliza como control interno
  Variable numérica Cuenta     #Variable auxiliar que se utiliza como control interno

Abrir AD movimientos
Leer AD movimientos
Imprimir "Informe diario de los movimientos" Salto de línea
Imprimir "Sucursal   Día   Cuenta   Depósitos   Extracciones" & Salto de línea

Mientras (NOT movimientos.EOF)
  Sucursal = movimientos.sucursal
  Imprimir Sucursal
  DepósitoSuc = 0
  ExtracciónSuc = 0

  Mientras (NOT movimientos.EOF AND movimientos.sucursal = Sucursal)
    Fecha = movimientos.fecha
    Imprimir Fecha
    DepósitoFec = 0
    ExtracciónFec = 0

    Mientras (NOT movimientos.EOF AND movimientos.fecha = Fecha)
      Cuenta = movimientos.Cuenta
      Imprimir Cuenta
      DepósitoCue = 0
      ExtracciónCue = 0

```

**Salida de resultados**

**Inicialización de variables**

```

Mientras (NOT movimientos.EOF AND movimientos.Cuenta = Cuenta)
    Si ( movimientos.tipo = "D")
        DepósitoCue = DepósitoCue + movimiento.importe
    Si no
        ExtracciónCue = ExtracciónCue + movimientos.importe
    fin si
    Leer AD movimientos
Fin mientras

Imprimir DepósitoCue ExtracciónCue Salto de Línea
DepósitoFec = DepósitoFec + DepósitoCue
ExtracciónFec = ExtracciónFec + ExtracciónCue
Fin mientras

Imprimir "Total del dia:" & DepósitoFec & ExtracciónFec & Salto de Línea
DepósitoSuc = DepósitoSuc + DepósitoFec
ExtracciónSuc = ExtracciónSuc + ExtracciónFec

Fin mientras
Imprimir "Total de la Sucursal:" & DepósitoSuc & ExtracciónSuc & Salto de Línea
ExtracciónGral = ExtracciónGral + ExtracciónSuc
DepósitoGral = DepósitoGral + DepósitoSuc
Fin mientras

Imprimir "Total General:" DepósitosGral ExtraccionesGral Salto de línea
Cerrar AD movimientos
Fin procedimiento

```

En nuestra Situación profesional podemos subdividir en subconjuntos de destino:

- **Subconjunto original**

Resaltar el conjunto de los registros correspondientes a la feria:

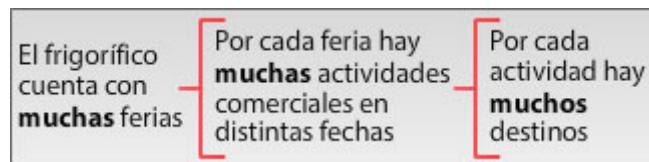


Código	Código	Destino	Fecha	Kilos	Precio
1011 con destino de Consumo	1011	San Salvador	Consumo	22/01/15	5000 13,30
1011 con destino de Exportación	1011	San Salvador	Consumo	02/02/15	3000 11
1047 con destino de Consumo	1011	San Salvador	Consumo	15/03/15	6000 10,50
1047 con destino de Exportación	1011	San Salvador	Exportación	12/01/15	1200 11,70
	1011	San Salvador	Exportación	07/02/15	7000 9,70
	1047	Romanutti S.A.	Consumo	10/03/15	1800 8,90
	1047	Romanutti S.A.	Exportación	07/02/15	1000 9,30
	1047	Romanutti S.A.	Exportación	23/03/15	2000 11,20

Cada subconjunto que se trate requiere de una estructura repetitiva diferente. Tenga en cuenta lo siguiente:

- El agregado de datos tiene **muchas** ferias.
- Cada feria tiene actividades comerciales en **muchas** fechas.
- Por cada actividad hay **muchos** destinos.

Lo descrito se puede entender mejor analizando el siguiente esquema:



- Análisis del AD

Todo lo representado en el **esquema** se verifica perfectamente analizando el agregado de datos que tenemos de ejemplo.

Código	Código	Destino	Fecha	Kilos	Precio
1011	San Salvador	Consumo	22/01/15	5000	13,30
			02/02/15	3000	11
			15/03/15	6000	10,50
		Exportación	12/01/15	1200	11,70
			07/02/15	7000	9,70
1047	Romanutti S.A.	Consumo	10/03/15	1800	8,90
		Exportación	07/02/15	1000	9,30
			23/03/15	2000	11,20

- Los registros están ordenados por número de feria.
- Luego por destino de la actividad comercial.
- Y por último término se ordena por fecha.

El algoritmo que se realizaremos requerirá de cuatro estructuras repetitivas:

- Una que controle el final del archivo.
- Otra que controle las ferias.
- Otras que controles los destinos.
- Otras que controle las fechas.

Por cada una de las estructuras repetitivas deberá respetarse las tres instancias de:

- Inicialización de variables.
- Operaciones de cálculo.
- Salida de resultados.

La comparación con el AD Original sería:

Código	Código	Destino	Fecha	Kilos	Precio
1011	San Salvador	Consumo	22/01/15	5000	13,30
1011	San Salvador	Consumo	02/02/15	3000	11
1011	San Salvador	Consumo	15/03/15	6000	10,50
1011	San Salvador	Exportación	12/01/15	1200	11,70
1011	San Salvador		07/02/15	7000	9,70
1047	Romanutti S.A.	Consumo	10/03/15	1800	8,90
1047	Romanutti S.A.	Exportación	07/02/15	1000	9,30
1047	Romanutti S.A.	Exportación	23/03/15	2000	11,20

Para realizar un reporte claro y con los resultados solicitados es necesario respetar los siguientes pasos:

**Paso 1:**

Analizar la salida de datos para determinar qué cálculos son necesarios.

**Paso 2:**

Analizar los conjuntos de datos y verificar si el orden de los datos coincide con la jerarquía de los conjuntos.

**Paso 3:**

Desarrollar el algoritmo teniendo en cuenta que:

- El agregado de datos:
  - Se abre al principio del procedimiento.
  - Se lee en el núcleo del procedimiento.
  - Se cierra junto antes de terminar el procedimiento.
- Los acumuladores y contadores generales:
  - Se inicializan al comenzar el procedimiento.
  - Se calculan en un nivel interior (dentro de la repetitiva general).
  - Se muestran al salir de la repetitiva.
  - Se muestran antes de finalizar el procedimiento.
- Los acumuladores y contadores parciales:
  - Se inicializan al comenzar la repetitiva propia del subconjunto al que pertenecen.
  - Se calculan en dentro de la repetitiva.
  - Se muestran antes de finalizar el procedimiento.

Estos tres pasos también serán de utilidad para resolver nuestra Situación profesional ya que, siguiendo los mismos, podremos obtener los acumulados de kilos vendidos por feria, por fecha, por destino y para toda salida que surja de nuestro análisis.



¿Estás listo para un desafío?

**1. Indique la opción correcta**

Los diagramas de Warnier muestran:

- Los informes que se emiten de un sistema.
- Los procesos y la secuencia en que se realizan.
- La cantidad de campos de un agregado de datos.

**2. Indique la opción correcta**

En un algoritmo que imprime un reporte, las variables se inicializan:

- Antes de ingresar a la estructura repetitiva donde son modificadas.
- Al comenzar el programa.
- Cada vez que se lee un registro del agregado de datos.

**3. Indique la opción correcta**

En un algoritmo de emisión de reportes, las variables se inicializan cuando se ingresa a la estructura repetitiva en donde serán modificadas.

- Verdadero
- Falso

**4. Indique la opción correcta**

La generación de resultados en un algoritmo de generación de emisión de reportes, se realiza dentro de la estructura repetitiva.

- Verdadero
- Falso

**5. Indique la opción correcta**

La salida de datos en un algoritmo de emisión de reportes se realiza a medida que se obtienen los mismos.

- Verdadero

Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Los diagramas de Warnier muestran:

- Los informes que se emiten de un sistema.
- Los procesos y la secuencia en que se realizan.
- La cantidad de campos de un agregado de datos.

## 2. Indique la opción correcta

En un algoritmo que imprime un reporte, las variables se inicializan:

- Antes de ingresar a la estructura repetitiva donde son modificadas.
- Al comenzar el programa.
- Cada vez que se lee un registro del agregado de datos.

## 3. Indique la opción correcta

En un algoritmo de emisión de reportes, las variables se inicializan cuando se ingresa a la estructura repetitiva en donde serán modificadas.

- Verdadero
- Falso

## 4. Indique la opción correcta

La generación de resultados en un algoritmo de generación de emisión de reportes, se realiza dentro de la estructura repetitiva.

- Verdadero
- Falso

## 5. Indique la opción correcta

La salida de datos en un algoritmo de emisión de reportes se realiza a medida que se obtienen los mismos.

- Verdadero
- Falso

## SP2 / Ejercicio resuelto

Según el enunciado de la SP2 a usted se le solicitó: el desarrollo de un el programa en pseudocódigo que emita un reporte que refleje determinada información.

### Información solicitada, para el desarrollo:

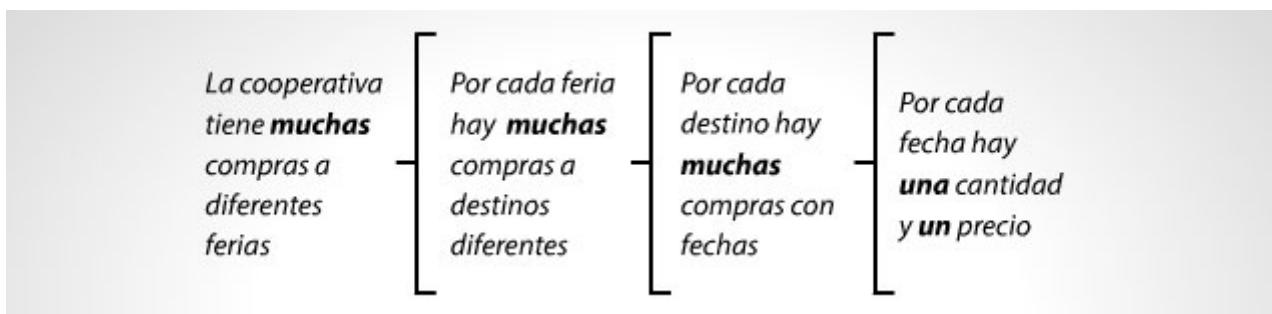
- Kilogramos comprados a cada feria y total pagado.
- Total de kilogramos destinados a consumo por feria y total pagado.
- Total de kilogramos destinados exportación y total pagado. También por feria.
- Total de kilogramos comprados a cada feria y total pagado.
- Total general de kilogramos comprados y total pagado.

Veamos a continuación el proceso del desarrollo solicitado:

- **Paso uno:** analizar la salida para determinar que operaciones son necesarias.

<b>Informe detallado de compras realizadas por feria</b>				
<b>Feria: 1011 - San Salvador</b>				
Destino	Fecha	Kilos	Importe	
Consumo	22/05/15	5000	\$66.500,00	Importe de la compra
	02/02/15	3000	\$33.000,00	
	15/03/15	6000	\$63.000,00	
	<b>Total:</b>	<b>14000</b>	<b>\$162.500,00</b>	Suma de los kilos por destino
Exportación	12/01/15	1200	\$14.040,00	
	07/02/15	7000	\$67.900,00	
	<b>Total:</b>	<b>8200</b>	<b>\$81.940,00</b>	Suma de los importes por destino
<b>Total correspondiente a la feria:</b>		<b>22200</b>	<b>\$244.440,00</b>	Suma de los importes por feria Suma de los kilos por feria
<b>Feria: 1047 - Romanutti S.A.</b>				
Destino	Fecha	Kilos	Importe	
Consumo	10/03/15	1800	\$16020,00	
	<b>Total:</b>	<b>1800</b>	<b>\$16020,00</b>	
Exportación	07/02/15	1000	\$9.300,00	
	23/03/15	2000	\$22.400,00	
	<b>Total:</b>	<b>3000</b>	<b>\$31.700,00</b>	
<b>Total correspondiente a la feria:</b>		<b>4800</b>	<b>\$47.720,00</b>	
<b>Total General:</b>		<b>27000</b>	<b>\$292.160,00</b>	Suma general de los importes Suma general de kilos

- **Paso dos:** analizar los conjuntos y el orden de los datos para determinar las repetitivas necesarias.



- La cooperativa tiene **muchas** compras a diferentes ferias

### **Informe detallado de compras realizadas por feria**

Feria: 1011 - San Salvador

Destino	Fecha	Kilos	Importe
Consumo	22/05/15	5000	\$66.500,00
	02/02/15	3000	\$33.000,00
	15/03/15	6000	\$63.000,00
<b>Total:</b>		<b>14000</b>	<b>\$162.500,00</b>
Exportación	12/01/15	1200	\$14.040,00
	07/02/15	7000	\$67.900,00
	<b>Total:</b>		<b>8200 \$162.500,00</b>

\* Ejemplificamos solo sobre las compras de una de las feria.

- Por cada feria hay **muchas** compras a destinos diferentes

### **Informe detallado de compras realizadas por feria**

Feria: 1011 - San Salvador

Destino	Fecha	Kilos	Importe
Consumo	22/05/15	5000	\$66.500,00
	02/02/15	3000	\$33.000,00
	15/03/15	6000	\$63.000,00
<b>Total:</b>		<b>14000</b>	<b>\$162.500,00</b>
Exportación	12/01/15	1200	\$14.040,00
	07/02/15	7000	\$67.900,00
	<b>Total:</b>		<b>8200 \$162.500,00</b>

\* Ejemplificamos solo sobre las compras de una de las feria.

- Por cada destino hay **muchas** compras con fechas

### **Informe detallado de compras realizadas por feria**

Feria: 1011 - San Salvador

Destino	Fecha	Kilos	Importe
Consumo	22/05/15	5000	\$66.500,00
	02/02/15	3000	\$33.000,00
	15/03/15	6000	\$63.000,00
	<b>Total:</b>	<b>14000</b>	<b>\$162.500,00</b>
Exportación	12/01/15	1200	\$14.040,00
	07/02/15	7000	\$67.900,00
	<b>Total:</b>	<b>8200</b>	<b>\$162.500,00</b>

\* Ejemplificamos solo sobre las compras de una de las feria.

- Por cada fecha hay **una** cantidad y **un** precio

### **Informe detallado de compras realizadas por feria**

Feria: 1011 - San Salvador

Destino	Fecha	Kilos	Importe
Consumo	22/05/15	5000	\$66.500,00
	02/02/15	3000	\$33.000,00
	15/03/15	6000	\$63.000,00
	<b>Total:</b>	<b>14000</b>	<b>\$162.500,00</b>
Exportación	12/01/15	1200	\$14.040,00
	07/02/15	7000	\$67.900,00
	<b>Total:</b>	<b>8200</b>	<b>\$162.500,00</b>

\* Ejemplificamos solo sobre las compras de una de las feria.

- Paso tres:** como el ordenamiento de los datos coincide con la jerarquía de los conjuntos, para desarrollar el algoritmo en pseudocódigo.

```

Procedimiento cmdImprimirReporte:Click
  Variable numérica KilosGral = 0 #Inicializo el acumulador general
  Variable numérica ImportesGral = 0 # Inicializo el acumulador general
  Variable numérica KilosFer = 0# Inicializo el acumulador por feria
  Variable numérica Importes Fer = 0# Inicializo el acumulador por feria
  Variable numérica KilosDes = 0# Inicializo el acumulador por destino
  Variable numérica ImportesDes = 0# Inicializo el acumulador por destino
  Variable numérica Importe = 0# Inicializo la variable de cálculo de precio
  Variable numérica Feria # Variable auxiliar que se utiliza como control interno
  Variable numérica Destino # Variable auxiliar que se utiliza como control interno
  Variable numérica Fecha # Variable auxiliar que se utiliza como control interno
  Abrir AD movimientos
  Leer AD compras
  Imprimir "Informe detallado de compras realizadas por feria" & SL
  Mientras ( NOT compras.EOF)
    Feria = compras.código
    Imprimir "Feria:" compras.codigo compras.nombre & SL
    Imprimir "DestinoFechaKilosImporte"
    KilosFer = 0
    ImportesFer = 0
    Mientras ( NOT compras.EOF AND compras.código = Feria)
      Destino = compras.destino
      Imprimir Destino
      KilosDes = 0
      ImportesDes = 0
      Mientras ( NOT compras.EOF AND compras.destino = Destino)
        Imprimir compras.fechacompras.kilos
        Importe = compras.kilos * compras.precio
        Imprimir Importe & Salto de Línea
        Leer AD compras
      Fin mientras
      Imprimir "Total:" & KilosDes & ImportesDes & SL
      KilosFer = KilosFer + KilosDes
      ImportesFer = ImportesFer + ImportesDes
    Fin mientras
    Imprimir "Total Correspondiente a la feria" & KilosFer & ImportesFer
    Imprimir Salto de Línea
    KilosGral = KilosGral + KilosFer
    ImportesGral = ImportesGral + ImportesFer
  Fin mientras
  Imprimir "Total General: " & KilosGral & ImportesGral & SL
  Cerrar AD compras
Fin procedimiento

```

## SP2 / Ejercicio por resolver

Realizar el algoritmo en pseudocódigo de un procedimiento que imprima el siguiente reporte utilizando el agregado de datos de la Situación profesional:

### Ejercicio por resolver SP2

PL2

#### Informe detallado de compras realizadas por feria

##### Feria: 1011 - San Salvador

Destino	Kilos	Importe
Consumo	14000	\$162.500,00
Exportación	8200	\$81.940,00
Total correspondiente a la feria:	22200	\$244.440,00

- Kilos por Destino
- Importes por Destino

##### Feria: 1047 - Romanutti S.A.

Destino	Kilos	Importe
Consumo	1800	\$16.020,00
Exportación	3000	\$31.700,00
Total correspondiente a la feria:	4800	\$47.720,00

**Total General:** 27000 \$29.216,00

Y... una pregunta: si el agregado de datos de la Situación profesional hubiera estado ordenado por fecha ¿se podría haber resuelto de la misma forma? ¿Por qué?



¡Vamos a comprobar cuánto aprendiste!

**1. Indique la opción correcta**

Un agregado de datos se abre al principio de un procedimiento de emisión de reporte.

- Verdadero
- Falso

**2. Indique la opción correcta**

Los acumuladores generales se muestran al comienzo de un procedimiento de emisión.

- Verdadero
- Falso

**3. Indique la opción correcta**

Un agregado de datos se lee solo una vez en un algoritmo de emisión de reporte.

- Verdadero
- Falso

**4. Indique la opción correcta**

Las variables parciales se declaran fuera del procedimiento de emisión de reporte.

- Verdadero
- Falso

**5. Indique la opción correcta**

Cuando se trata el conjunto de registros de un agregado de datos una sola estructura repetitiva es suficiente para recorrer los mismos. Pero cuando se tratan subconjuntos se requiere de otra estructura repetitiva más.

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Un agregado de datos se abre al principio de un procedimiento de emisión de reporte.

Verdadero

Falso

## 2. Indique la opción correcta

Los acumuladores generales se muestran al comienzo de un procedimiento de emisión.

Verdadero

Falso

## 3. Indique la opción correcta

Un agregado de datos se lee solo una vez en un algoritmo de emisión de reporte.

Verdadero

Falso

## 4. Indique la opción correcta

Las variables parciales se declaran fuera del procedimiento de emisión de reporte.

Verdadero

Falso

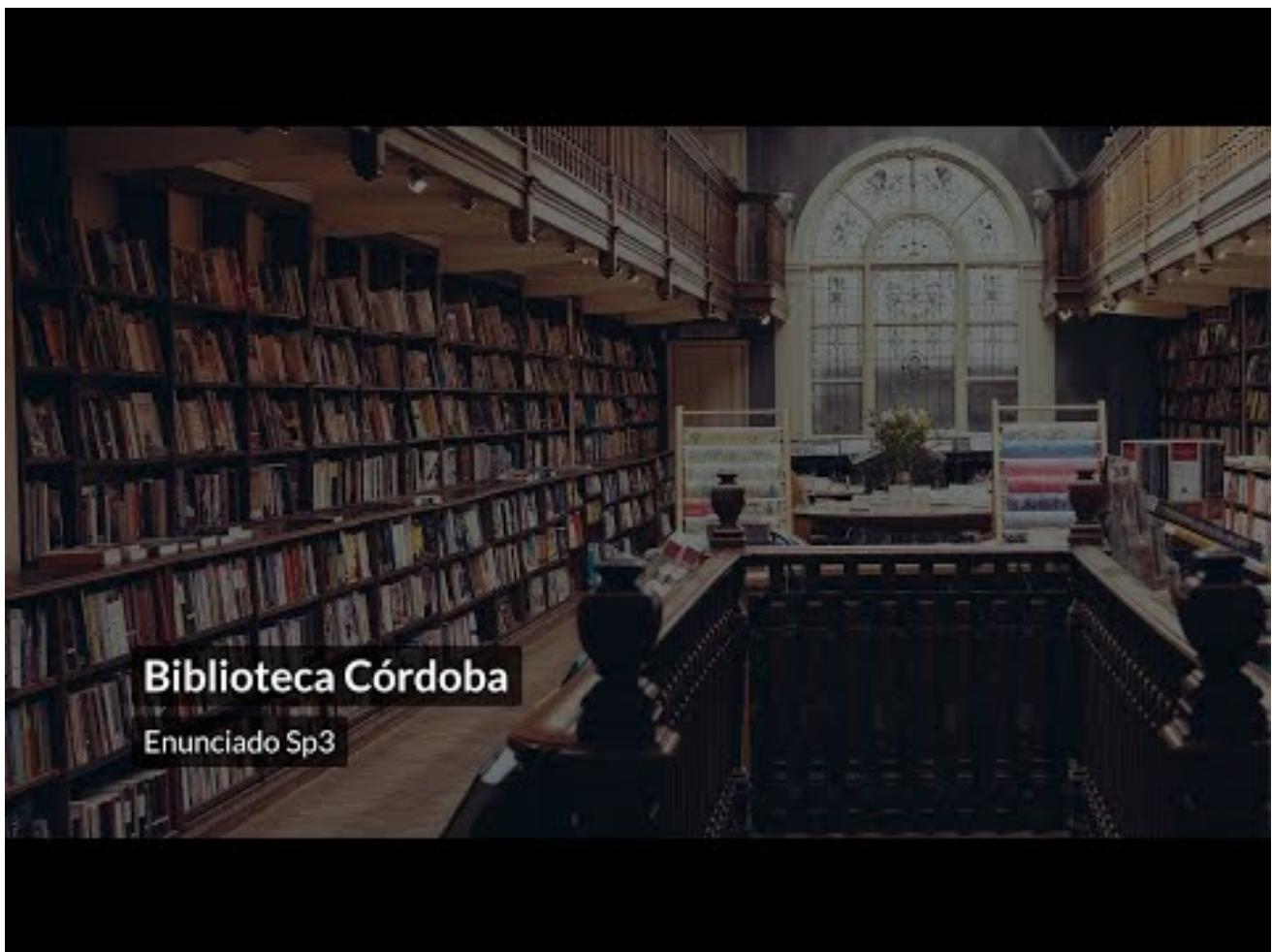
## 5. Indique la opción correcta

Cuando se trata el conjunto de registros de un agregado de datos una sola estructura repetitiva es suficiente para recorrer los mismos. Pero cuando se tratan subconjuntos se requiere de otra estructura repetitiva más.

Verdadero

Falso

# Situación profesional 3: Biblioteca CÓRDOBA



**Biblioteca Córdoba**

Enunciado Sp3

Video "[Enunciado SP3](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

La Biblioteca Córdoba le solicita a usted, pasante de la misma, la confección de diferentes tipos

de listados que pasamos a detallar:

- Listado de los Libros "Sin Devolver".
- Cantidad de libros "Sin Devolver" por Carrera.
- Listado de libros prestados a un estudiante en particular.
- Informe estadístico de préstamos por carrera.

La Biblioteca dispone de los siguientes agregados de datos:

ADCarreras

Código Nombre

I Informática

K Marketing

M Management

P Publicidad

R Recursos

Humanos

EOF

AD Préstamos  
Carrera Fecha Legajo Título Devuelto  
I 01/07/15 1245 Programación Estructurada V  
I 25/08/15 2235 La biblia del C++ F  
I 26/08/15 1245 RM-COBOL 85 Avanzado V  
I 02/09/15 2567 PC Interno II V  
M 03/07/15 3343 Investigación Operativa V  
M 03/07/15 4453 Administración Financiera F  
M 07/09/15 4453 Estructuras de Costos F  
M 04/08/15 2345 El modelo de PERT V  
P 14/08/15 3547 El Modelo Comunicacional Actual F  
P 17/10/15 4698 Los efectos de la Desinformación V  
R 16/08/15 2474 Psicología Evolutiva F  
R 15/09/15 2998 Estructuras Organizacionales V

EOF

Ordenamiento:

- El AD Carreras se encuentra ordenado por el campo Código.
- El AD Préstamos se encuentra ordenado por Carrera como primer campo clave, y Fecha en segundo lugar.

Relaciones:

- El campo Código del AD Carreras se encuentra relacionado con el campo Carreras del AD Préstamos, en una relación de Uno a Muchos. Esto significa que para 1 carrera del

84

AD Carreras puede haber muchos préstamos en el AD Préstamos. De esto se desprende que la relación sea 1 Muchos

Se deberán confeccionar los pseudocódigos necesarios para obtener la siguiente información:

- Listado de todos los libros “Sin Devolver”, clasificados por carrera, indicando la fecha en que cada uno de ellos fue prestado, el legajo del alumno que lo tiene, y el título del libro en cuestión:
  - Cantidad de libros “Sin Devolver” por Carrera, indicando el nombre de la carrera:
  - Listado de libros prestados a un estudiante en particular (especificando el legajo del mismo), indicando Fecha del préstamo y Estado de devolución (devuelto o no):

85

- Informe estadístico de préstamos por carrera, según el siguiente modelo de salida del Reporte:

AD Carreras		AD Préstamos				
Código	Nombre	Carrera	Fecha	Legajo	Título	Devuelto
I	Informática	I	01/07/15	1245	Programación Estructurada	V
K	Marketing	I	25/08/15	2235	La biblia del C++	F
M	Management	I	26/08/15	1245	RM-COBOL 85 Avanzado	V
P	Publicidad	I	02/09/15	2567	PC Interno II	V
R	Recursos Humanos	M	03/07/15	3343	Investigación Operativa	V
EOF		M	03/07/15	4453	Administración Financiera	F
		M	07/09/15	4453	Estructuras de Costos	F
		M	04/08/15	2345	El modelo de PERT	V
		P	14/08/15	3547	El Modelo Comunicacional Actual	F
		P	17/10/15	4698	Los efectos de la Desinformación	V
		R	16/08/15	2474	Psicología Evolutiva	F
		R	15/09/15	2998	Estructuras Organizacionales	V
EOF						

### Listado de libros sin devolver

Informática	Fecha	Legajo	Título
	25/08/15	2235	La Biblia del C++
Management	Fecha	Legajo	Título
	03/07/15	4453	Administración Financiera
	07/09/15	4453	Estructuras de Costos
Publicidad	Fecha	Legajo	Título
	14/08/15	3547	El Modelo Comunicacional Actual
Recursos Humanos	Fecha	Legajo	Título
	16/08/15	2474	Psicología Evolutiva

### Cantidad de libros sin devolver por carrera

Carrera	Cantidad
Informática	1
Management	2
Marketing	0
Publicidad	1
Recursos Humanos	1

**Libros Prestados al Alumno: 4453**

<i>Fecha</i>	<i>Devuelto</i>
03/07/15	No
07/09/15	No

**Informe estadístico de préstamos**

Carrera	Prestados	Devueltos	Indice Devolución
Informática	132	98	74.25 %
Marketing	226	143	63.27 %
Management	103	45	43.69 %
Publicidad	56	20	35.71 %
Recursos Humanos	72	71	98.61 %
<b>TOTALES</b>	<b>589</b>	<b>377</b>	<b>64.01 %</b>

# SP3 / H1: Procesamiento de múltiples tablas relacionadas.

## Modelo de tablas relacionales

Generalmente los programas no trabajan con un solo archivo, sino que trabajan con conjuntos de archivos que están relacionados entre sí. Para poder procesarlos es necesario utilizar una técnica denominada "Apareo de Archivos".

Es necesario trabajar con archivos relacionados ya que no se puede tener toda la información de un sistema almacenada en un único agregado de datos porque se generaría repetición de información.

Imagine que en un archivo de un instituto de educación se han almacenado los datos personales de los alumnos y las notas de las materias. Si toda esta información estuviera en un único agregado de datos, el mismo podría tener los siguientes registros:

AD Alumnos								PL2
DNI	Nombre	Dirección	Carrera	Fecha	Materia	Tipo	Nota	
27368740	ORTIZ, Walter	Colón 310	ASC	21/04/2015	Ingles I	1° Parcial	10	
27368740	ORTIZ, Walter	Colón 310	ASC	14/06/2015	Ingles I	2° Parcial	8	
27368740	ORTIZ, Walter	Colón 310	ASC	12/04/2015	Lógica I	1° Parcial	9	
27368740	ORTIZ, Walter	Colón 310	ASC	01/06/2015	Lógica I	2° Parcial	7	
27368740	ORTIZ, Walter	Colón 310	ASC	30/04/2015	Matemática I	1° Parcial	9	
28360789	GARCIA, Alejandro	Lima 1250	ASC	21/04/2015	Ingles I	1° Parcial	9	
28360789	GARCIA, Alejandro	Lima 1250	ASC	14/06/2015	Ingles I	2° Parcial	7	
28360789	GARCIA, Alejandro	Lima 1250	ASC	12/04/2015	Lógica I	1° Parcial	6	
28360789	GARCIA, Alejandro	Lima 1250	ASC	01/06/2015	Lógica I	2° Parcial	4	
28360789	GARCIA, Alejandro	Lima 1250	ASC	30/04/2015	Matemática I	1° Parcial	3	

El campo DNI es clave única en el en el **AD Alumnos**, y es clave con duplicados en el **AD Exámenes**. En donde **Clave única** es una *clave compuesta por campos* cuyos valores jamás se repiten para dos registros diferentes, es decir, que identifica de manera biunívoca a cada registro de la Tabla o Agregado de Datos.

**Clave con Duplicados :** es una clave compuesta por campos cuyos valores pueden repetirse en más de un registro, por lo que no identifica de manera biunívoca a cada registro.

Una relación 1-> *Muchos* es la establecida entre dos agregados de datos siempre que se verifique que para un registro del primer archivo pueden existir muchos registros en el segundo:

## Relación uno muchos

PL2

A un alumno le corresponden muchos exámenes

AD Alumnos

DNI	Nombre
27368740	ORTÍZ, Walter
28360789	GARCIA, Alejandro
29465780	MALDONADO, Juan
29650423	MUÑOS, Karina
29678201	MARTINEZ, Olga
27798008	LUNA Enrique
...	...

AD Exámenes

DNI	Fecha	Materia
27368740	21/04/2015	Inglés I
27368740	14/06/2015	Inglés I
27368740	12/04/2015	Lógica I
27368740	01/06/2015	Lógica I
28360789	14/06/2015	Inglés I
28360789	12/04/2015	Lógica I
28360789	01/06/2015	Lógica I
28360789	30/04/2015	Matemática I
...	...	...

## Relación uno muchos

PL2

A un alumno le corresponden muchos exámenes

AD Alumnos

DNI	Nombre
27368740	ORTÍZ, Walter
28360789	GARCIA, Alejandro
29465780	MALDONADO, Juan
29650423	MUÑOS, Karina
29678201	MARTINEZ, Olga
27798008	LUNA Enrique
...	...

AD Exámenes

DNI	Fecha	Materia
27368740	21/04/2015	Inglés I
27368740	14/06/2015	Inglés I
27368740	12/04/2015	Lógica I
27368740	01/06/2015	Lógica I
28360789	14/06/2015	Inglés I
28360789	12/04/2015	Lógica I
28360789	01/06/2015	Lógica I
28360789	30/04/2015	Matemática I
...	...	...

Para procesar estos agregados de datos apareados lo que va a tener que tener en cuenta es que por cada lectura que se haga en el primer agregado de datos deberá realizar muchas en el segundo.

En nuestra Situación profesional también hacemos uso de *dos AD* que son el de *Carreras* y el de *Préstamos*, que al igual que los AD del ejemplo anterior se relacionan por un campo clave como es el código de carrera.

**AD Carreras**

Código	Nombre
I	Informática
K	Marketing
M	Management
P	Publicidad
R	Recursos Humanos

**EOF****AD Préstamos**

Carrera	Fecha	Legajo	Título	Devuelto
I	01/07/15	1245	Programación Estructurada	V
I	25/08/15	2235	La biblia del C++	F
I	26/08/15	1245	RM-COBOL 85 Avanzado	V
I	02/09/15	2567	PC Interno II	V
M	03/07/15	3343	Investigación Operativa	V
M	03/07/15	4453	Administración Financiera	F
M	07/09/15	4453	Estructuras de Costos	F
M	04/08/15	2345	El modelo de PERT	V
P	14/08/15	3547	El Modelo Comunicacional Actual	F
P	17/10/15	4698	Los efectos de la Desinformación	V
R	16/08/15	2474	Psicología Evolutiva	F
R	15/09/15	2998	Estructuras Organizacionales	V

**EOF****AD Carreras**

Código	Nombre
I	Informática
K	Marketing
<b>M</b>	Management
P	Publicidad
R	Recursos Humanos

**EOF****AD Préstamos**

Carrera	Fecha	Legajo	Título	Devuelto
I	01/07/15	1245	Programación Estructurada	V
I	25/08/15	2235	La biblia del C++	F
I	26/08/15	1245	RM-COBOL 85 Avanzado	V
I	02/09/15	2567	PC Interno II	V
M	03/07/15	3343	Investigación Operativa	V
M	03/07/15	4453	Administración Financiera	F
M	07/09/15	4453	Estructuras de Costos	F
M	04/08/15	2345	El modelo de PERT	V
P	14/08/15	3547	El Modelo Comunicacional Actual	F
P	17/10/15	4698	Los efectos de la Desinformación	V
R	16/08/15	2474	Psicología Evolutiva	F
R	15/09/15	2998	Estructuras Organizacionales	V

**EOF**

## Relación uno muchos, en la SP

PL2

### AD Carreras

Código	Nombre
I	Informática
K	Marketing
M	Management
P	Publicidad
R	Recursos Humanos

EOF

### AD Préstamos

Carrera	Fecha	Legajo	Título	Devuelto
I	01/07/15	1245	Programación Estructurada	V
I	25/08/15	2235	La biblia del C++	F
I	26/08/15	1245	RM-COBOL 85 Avanzado	V
I	02/09/15	2567	PC Interno II	V
M	03/07/15	3343	Investigación Operativa	V
M	03/07/15	4453	Administración Financiera	F
M	07/09/15	4453	Estructuras de Costos	F
M	04/08/15	2345	El modelo de PERT	V
P	14/08/15	3547	El Modelo Comunicacional Actual	F
P	17/10/15	4698	Los efectos de la Desinformación	V
R	16/08/15	2474	Psicología Evolutiva	F
R	15/09/15	2998	Estructuras Organizacionales	V

EOF

## Relación uno muchos, en la SP

PL2

### AD Carreras

Código	Nombre
I	Informática
K	Marketing
M	Management
P	Publicidad
R	Recursos Humanos

EOF

### AD Préstamos

Carrera	Fecha	Legajo	Título	Devuelto
I	01/07/15	1245	Programación Estructurada	V
I	25/08/15	2235	La biblia del C++	F
I	26/08/15	1245	RM-COBOL 85 Avanzado	V
I	02/09/15	2567	PC Interno II	V
M	03/07/15	3343	Investigación Operativa	V
M	03/07/15	4453	Administración Financiera	F
M	07/09/15	4453	Estructuras de Costos	F
M	04/08/15	2345	El modelo de PERT	V
P	14/08/15	3547	El Modelo Comunicacional Actual	F
P	17/10/15	4698	Los efectos de la Desinformación	V
R	16/08/15	2474	Psicología Evolutiva	F
R	15/09/15	2998	Estructuras Organizacionales	V

EOF

## Ejemplo 1: Materias rendidas por Alumno

<b>Materias rendidas por Alumnos</b>				
<b>Alumno</b>	<b>Fecha</b>	<b>Materia</b>	<b>Tipo Examen</b>	<b>Nota</b>
27.368.740 ORTIZ, Walter	21/04/2015	Inglés I	1º Parcial	10
	14/06/2015	Inglés I	2º Parcial	8
	12/04/2015	Lógica I	1º Parcial	9
	01/06/2015	Lógica I	2º Parcial	7
	30/04/2015	Matemática I	1º Parcial	9
28360789 GARCIA, Alejandro	21/04/2015	Inglés I	1º Parcial	10
	14/06/2015	Inglés I	2º Parcial	8
	12/04/2015	Lógica I	1º Parcial	9
	01/06/2015	Lógica I	2º Parcial	7
	30/04/2015	Matemática I	1º Parcial	9

**Materias rendidas por alumnos**

Video "[Ejemplo 1: Materias rendidas por Alumno](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

El programa que resuelva esta situación deberá tener definidos los registros correspondientes a los dos agregados de datos. A continuación se realiza el procedimiento correspondiente al listado. Observe que en el procedimiento, por cada lectura que se realiza en el agregado de datos Alumnos, se realizaron muchas lecturas en el agregado de datos Exámenes.

Cuando se trabaja con dos agregados de datos, se pueden realizar todas las operaciones vistas hasta el momento, como búsquedas y recorridos.

En nuestra Situación profesional tenemos un caso similar, como los préstamos de libros realizados por carrera, que podemos resolver con un algoritmo visto en el ejemplo anterior.

A continuación veremos un ejemplo más.

El siguiente es el modelo del listado que se quiere imprimir:

### Listado de libros sin devolver

	<b>Fecha</b>	<b>Legajo</b>	<b>Título</b>
Informática	25/08/15	2235	La Biblia del C++
Management	<b>Fecha</b>	<b>Legajo</b>	<b>Título</b>
	03/07/15	4453	Administración Financiera
	07/09/15	4453	Estructuras de Costos
Publicidad	<b>Fecha</b>	<b>Legajo</b>	<b>Título</b>
	14/08/15	3547	El Modelo Comunicacional Actual
Recursos Humanos	<b>Fecha</b>	<b>Legajo</b>	<b>Título</b>
	16/08/15	2474	Psicología Evolutiva

Listado de todos los libros "Sin Devolver", clasificados por carrera, indicando la fecha en que cada uno de ellos fue prestado, el legajo del alumno que lo tiene y el título del libro en cuestión.

```

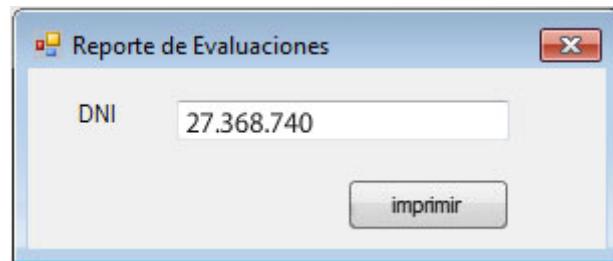
# Definición de Datos Globales
# Definición del Registro
Registro Carrera
    Variable Alfanumérica Código
    Variable Alfanumérica Nombre
Fin Registro
# Definición del Agregado de Datos
Agregado de Datos Carreras Tipo Registro Carrera
# Definición del Registro
Registro Préstamo
    Variable Alfanumérica Carrera
    Variable Fecha Fecha
    Variable Numérica Legajo
    Variable Alfanumérica Titulo
    Variable Lógica Devuelto
Fin Registro
# Definición del Agregado de Datos
Agregado de Datos Préstamos Tipo Registro Préstamo
Procedimiento ListaSinDevolver ( )
    Imprimir "Listado de Libros sin Devolver", SL, SL
    Abrir AD Carreras
    Leer AD Carreras
    Abrir AD Préstamos
    Leer AD Préstamos
    Mientras ( NOT Carrera.EOF )
        Imprimir Carrera.Nombre
        Imprimir "Fecha      Legajo      Titulo", SL
        Mientras ( NOT Préstamo.EOF AND Carrera.Código = Prestamo.Carrera )
            Si ( Préstamo.Devuelto = F )
                Imprimir Préstamo.Fecha
                Imprimir Préstamo.Legajo
                Imprimir Préstamo.Titulo, SL
            Fin Si
            Leer AD Préstamos
        Fin Mientras
        Imprimir SL, SL
        Leer AD Carreras
    Fin Mientras
    Cerrar AD Préstamos
    Cerrar AD Carreras
Fin Procedimiento

```

### Ejemplo 2: Evaluaciones de un Alumno

Realizar un programa que permita emitir un reporte impreso de las evaluaciones de un alumno en particular.

Solamente se programa el procedimiento considerando que se utiliza la siguiente interfaz gráfica:



El siguiente es el modelo del listado a imprimir:

**Exámenes de ORTIZ, Walter (27.368.740)**

<b>Fecha</b>	<b>Materia</b>	<b>Tipo Examen</b>	<b>Nota</b>
21/04/2015	Inglés I	1º Parcial	10
14/06/2015	Inglés I	2º Parcial	8
12/04/2015	Lógica I	1º Parcial	9
01/06/2015	Lógica I	2º Parcial	7
30/04/2015	Matemática I	1º Parcial	9

```

Procedimiento cmdReporte:Click
  Abrir AD Alumnos
  Abrir AD Exámenes
  Leer AD Alumnos
  Leer AD Exámenes
  Mientras ( NOT Alumnos.EOF AND Alumno.Nombre <> txtAlumno)
    # Lee todos los registros hasta encontrar el nombre del alumno
    Leer AD Alumnos
  Fin Mientras

  # Verifica si se encontró el alumno buscado
  Si ( Alumno.Nombre = txtAlumno)
    Imprimir "Exámenes de:", Alumno.Nombre, SL, SL
    # Repetitiva que saltea todos los exámenes que no son del alumno buscado
    Mientras ( NOT Examenes.EOF AND Examen.Dni <> Alumno.Dni )
      Leer AD Examen
    Fin Mientras
    Mientras ( NOT Examenes.EOF AND Examen.Dni = Alumno.Dni )
      # Imprime todos los exámenes del alumno
      Imprimir Examen.Fecha & Examen.Materia & Examen.Tipo & Examen.Nota & SL
      Leer AD Examenes
    Fin Mientras
  Si No
    Mensaje
      Título: "Dato inexistente"
      Icono: Advertencia
      Texto: "El DNI del alumno no está registrado"
    Fin Mensaje
  Fin Si
  Cerrar AD Alumnos
  Cerrar AD Exámenes
Fin Procedimiento

```

En nuestra Situación profesional se nos solicita un "Listado de libros prestados a un estudiante en particular", que encuadra con este ejemplo. En el ejemplo podemos ver con claridad, como lograrlo, siguiendo el mismo procedimiento.



¿Te animás a medir cuánto aprendiste?

**1. Indique la opción correcta**

Clave única es una clave compuesta por números de legajos o números de DNI.

- Verdadero
- Falso

**2. Indique la opción correcta**

Una relación 1 sobre muchos es la establecida entre un agregados de datos y muchos otros.

- Verdadero
- Falso

**3. Indique la opción correcta**

Para procesar agregados de datos apareados, por cada lectura que se haga en el primer agregado de datos deberá realizar muchas en el segundo.

- Verdadero
- Falso

**4. Indique la opción correcta**

Clave con duplicados es una clave compuesta por campos cuyos valores pueden repetirse en más de un registro.

- Verdadero
- Falso

**5. Indique la opción correcta**

La técnica denominada apareo de archivo sirve para trabajar con un conjunto de archivos con el mismo nombre.

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Clave única es una clave compuesta por números de legajos o números de DNI.

Verdadero

Falso

## 2. Indique la opción correcta

Una relación 1 sobre muchos es la establecida entre un agregados de datos y muchos otros.

Verdadero

Falso

## 3. Indique la opción correcta

Para procesar agregados de datos apareados, por cada lectura que se haga en el primer agregado de datos deberá realizar muchas en el segundo.

Verdadero

Falso

## 4. Indique la opción correcta

Clave con duplicados es una clave compuesta por campos cuyos valores pueden repetirse en más de un registro.

Verdadero

Falso

## 5. Indique la opción correcta

La técnica denominada apareo de archivo sirve para trabajar con un conjunto de archivos con el mismo nombre.

Verdadero

Falso

## SP3 / H2: Integración algorítmica de vectores y matrices

Es importante considerar que los programas requieren la utilización de muchas estructuras de datos. Existen muy pocos programas que utilicen solo vectores, o solo matrices o solo agregados de datos. Por este motivo es fundamental que un programador tenga claridad respecto de las diferencias existentes entre las estructuras de almacenamientos y de las operaciones que se realizan con las mismas.

En rigor de verdad, esta herramienta que le presentamos no es completamente nueva ya que, a través de la siguiente tabla, haremos una comparativa de las estructuras de datos vistas para que pueda decidir cómo programar cuando se requiere la utilización de muchas estructuras de datos. Veamos en el siguiente cuadro, las características de cada una de ellas:

### Comparativa de las estructura de datos

PL2

Vectores	Matrices	Agregados de Datos
Homogéneos.	Homogéneos.	Homogéneos (el Registro es una estructura Heterogénea).
Requieren un índice.	Requieren dos índices (uno para apuntar la fila y otro para la columna).	No utilizan índices, el recorrido es secuencial y se accede a cada campo del registro por su nombre.
Se accede a los datos en forma directa, por medio de un índice.	Se accede a los datos en forma directa, por medio de los índices.	Se accede a los datos secuencialmente en sucesivas lecturas.
Para recorrerlo se requiere una estructura repetitiva.	Para recorrerla completamente requiere de dos estructuras repetitivas. El barrido puede hacerse por filas o por columnas.	Para recorrerlo completamente se requiere de una estructura repetitiva, hasta hallar el Fin del Agregado de Datos (EOF).

Para tener mayor claridad, analicemos lo visto con el siguiente ejemplo:

#### Ejemplo: Empresa de cotizaciones

Una empresa que realiza el seguimiento de la evolución de las cotizaciones de diferentes monedas en diferentes bolsas de valores del mundo, recibe diariamente un AD con los valores iniciales de dichas cotizaciones y, posteriormente, debe procesar otro AD donde se reflejan las modificaciones de las diferentes cotizaciones porcentualmente y de manera evolutiva durante el día bursátil.

Las Monedas y las Bolsas de Valores se encuentran codificadas numéricamente, a fin de poder disponer los valores en una matriz de 10 filas por 12 columnas, correspondiendo las filas a las diferentes monedas en cotización (Moneda 1 - U\$S - , Moneda 2 - Yens - , ... , Moneda 10 - Pesos -) y las columnas de diferentes bolsas de valores (Bolsa 1 -Wall Street- , Bolsa 2 - Tokyo - , ... , Bolsa 12 - Buenos Aires - ).

## Ejemplo: Bolsa de cotizaciones

PL2

### Detalle de los AD

### Matriz

Detalle de los Agregados de Datos de Ejemplo:

#### AD Valores

Valor
1,02
1,09
1,68
...
...
0,98
1,56
1,88
...
...
2,67

EOF

- 1,02 ← Moneda 1/Bolsa 1
- 1,09 ← Moneda 1/Bolsa 2
- 1,68 ← Moneda 1/Bolsa 3
- ...
- ...
- 0,98 ← Moneda 1/Bolsa 12
- 1,56 ← Moneda 2/Bolsa 1
- 1,88 ← Moneda 2/Bolsa 2
- ...
- ...

2,67 ← Moneda 10/Bolsa 12

#### AD Modificaciones

Moneda	Bolsa	Porcentaje	Sentido
1	1	0,50 %	A
5	3	1,06 %	B
3	7	0,02 %	B
8	4	0,09 %	B
10	9	1,01 %	A
...	...	...	...
...	...	...	...
5	3	2,00 %	B

EOF

El AD contiene las diferentes "Modificaciones", indicándose el N° de **Moneda**, el N° de **Bolsa**, el **Porcentaje** en que se modifica la cotización y el **sentido** de la modificación (en Alza o en Baja).

Detalle de los AD &gt; Matriz

	B.1	B.2	B.3	B.4	B.5	B.6	B.7	B.8	B.9	B.10	B.11	B.12
M.1	XXX	XXX	XXX									
M.2	XXX	XXX	XXX									
M.3	XXX	XXX	XXX									
M.4	XXX	XXX	XXX									
M.5	XXX	XXX	XXX									
M.6	XXX	XXX	XXX									
M.7	XXX	XXX	XXX									
M.8	XXX	XXX	XXX									
M.9	XXX	XXX	XXX									
M.10	XXX	XXX	XXX									

(\*) Sustituyendo el indicativo XXX por el valor de la cotización correspondiente a la Moneda/Bolsa de cada posición.

Este Agregado de Datos deberá contener exactamente **120 registros** (**10 monedas x 12 Bolsas**). Si tiene más registros que los esperados significará que algún dato está repetido o redundante por error, pero no sabremos cuál, por lo que no podremos confiar en ninguno de los registros del Agregado de Datos.

Si, por el contrario, el Agregado de Datos tiene menos de 120 registros significará que está incompleto o truncado, por lo que tampoco deberemos confiar en la información contenida en él, ni siquiera parcialmente.

```
# Declaración de Variables Globales  
Matriz Numérica Cotizaciones [10] [12]
```

```
Registro Valor
```

```
    Variable Numérica Valor  
Fin Registro
```

```
Agregado de Datos Valores Tipo Registro Valor
```

```
Registro Modificaciones
```

```
    Variable Numérica Moneda  
    Variable Numérica Bolsa  
    Va Programación_bolsa de cotizaciones  
    Variable Alfanumérica Sentido  
Fin Registro
```

Video "Programación: Bolsa de cotizaciones" | Elaboración Propia. DEPROE - Colegio Universitario IES  
<https://www.youtube.com/watch?v=Jzo3vpydHdw>

De esta manera y a través de este ejemplo, hemos observado cómo los distintos tipos de estructuras estudiados, hasta el momento, pueden interactuar entre sí con el fin de resolver cualquier situación que se le plantee como programador.

Si bien le será de utilidad en innumerables situaciones, en el caso de nuestra Situación profesional no es aplicable esta herramienta.



¿Estás listo para un desafío?

**1. Indique la opción correcta**

Para recorrer una matriz se necesita de:

- Dos estructuras repetitivas.
- Una estructura repetitiva.
- Dos estructuras repetitivas y un índice.

**2. Indique la opción correcta**

A un agregado datos se accede:

- Por medio de un índice.
- Por medio de una clave.
- De manera secuencial.

**3. Indique la opción correcta**

Vectores, matrices y agregado de datos, son estructuras homogéneas.

- Verdadero
- Falso

**4. Indique la opción correcta**

Las matrices requieren de dos índices para ser recorridas.

- Verdadero
- Falso

**5. Indique la opción correcta**

Los agregados de datos necesitan tantos índices como subconjuntos posean.

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Para recorrer una matriz se necesita de:

- X Dos estructuras repetitivas.
- o Una estructura repetitiva.
- o Dos estructuras repetitivas y un índice.

## 2. Indique la opción correcta

A un agregado datos se accede:

- o Por medio de un índice.
- o Por medio de una clave.
- X De manera secuencial.

## 3. Indique la opción correcta

Vectores, matrices y agregado de datos, son estructuras homogéneas.

X Verdadero

o Falso

## 4. Indique la opción correcta

Las matrices requieren de dos índices para ser recorridas.

X Verdadero

o Falso

## 5. Indique la opción correcta

Los agregados de datos necesitan tantos índices como subconjuntos posean.

o Verdadero

X Falso

## SP3 / Ejercicio resuelto

El problema planteado en la Situación profesional 3: Biblioteca CÓRDOBA requiere la utilización de dos agregados de datos, que en el programa deberán definirse como datos globales:

# Definición de Datos Globales

# Definición del Registro

**Registro Carrera**

Variable Alfanumérica Código

Variable Alfanumérica Nombre

**Fin Registro**

# Definición del Agregado de Datos

**Agregado de Datos Carreras Tipo Registro Carrera**

# Definición del Registro

**Registro Préstamo**

Variable Alfanumérica Carrera

Variable Fecha Fecha

Variable Numérica Legajo

Variable Alfanumérica Título

Variable Lógica Devuelto

**Fin Registro**

# Definición del Agregado de Datos

**Agregado de Datos Préstamos Tipo Registro Préstamo**

A continuación, se resuelve cada procedimiento por separado:

**Libros Sin Devolver**

El siguiente es el modelo del listado que se quiere imprimir:

### Listado de libros sin devolver

Carrera	Fecha	Legajo	Título
	25/08/15	2235	La Biblia del C++
Management	Fecha	Legajo	Título
	03/07/15	4453	Administración Financiera
	07/09/15	4453	Estructuras de Costos
Publicidad	Fecha	Legajo	Título
	14/08/15	3547	El Modelo Comunicacional Actual
Recursos Humanos	Fecha	Legajo	Título
	16/08/15	2474	Psicología Evolutiva

Listado de todos los libros "Sin Devolver", clasificados por carrera, indicando la fecha en que cada uno de ellos fue prestado, el legajo del alumno que lo tiene y el título del libro en cuestión.

```
Procedimiento ListaSinDevolver ( )
  Imprimir "Listado de Libros sin Devolver", SL, SL
  Abrir AD Carreras
  Leer AD Carreras
  Abrir AD Préstamos
  Leer AD Préstamos
  Mientras ( NOT Carrera.EOF )
    Imprimir Carrera.Nombre
    Imprimir "Fecha      Legajo      Título", SL
    Mientras ( NOT Préstamo.EOF AND Carrera.Código = Prestamo.Carrera )
      Si ( Préstamo.Devuelto = F )
        Imprimir Préstamo.Fecha
        Imprimir Préstamo.Legajo
        Imprimir Préstamo.Título, SL
      Fin Si
      Leer AD Préstamos
    Fin Mientras
    Imprimir SL, SL
    Leer AD Carreras
  Fin Mientras
  Cerrar AD Préstamos
  Cerrar AD Carreras
Fin Procedimiento
```

### Libros "Sin devolver" por carrera

El siguiente es el modelo del listado que se quiere imprimir:

### Cantidad de libros sin devolver por carrera

Carrera	Cantidad
Informática	1
Management	2
Marketing	0
Publicidad	1
Recursos Humanos	1

Cantidad de libros "Sin Devolver" por Carrera, indicando el nombre de la carrera.

```
Procedimiento SinDevolverPorCarrera ( )
  Variable Numérica ContCarrera
  Imprimir "Libros Sin Devolver Por Carrera", SL, SL
  Imprimir "Carrera" Cantidad", SL
  Abrir AD Carreras
  Leer AD Carreras
  Abrir AD Préstamos
  Leer AD Préstamos
  Mientras ( NOT Carreras.EOF )
    # Contar los libros "no devueltos" de la carrera actual
    ContCarrera = 0
    Mientras ( NOT Préstamo.EOF AND Préstamo.Carrera = Carrera.Código )
      # Mientras no cambie la carrera actual en el AD Préstamos
      Si ( Préstamo.Devuelto = F )
        ContCarrera = ContCarrera + 1
      Fin Si
      Leer AD Préstamos
    Fin Mientras
    Imprimir Carrera.Nombre, ContCarrera,SL, SL
    Leer AD Carreras
  Fin Mientras
  Cerrar AD Préstamos
  Cerrar AD Carreras
Fin Procedimiento
```

### Observación

```

Procedimiento SinDevolverPorCarrera ( )
  Variable Numérica ContCarrera
  Imprimir "Libros Sin Devolver Por Carrera", SL, SL
  Imprimir "Carrera" Cantidad" SI
  Abrir Observe que la Repetitiva Interna permite procesar los registros del AD Préstamo en los que el
  Leer campo Carrera tenga el mismo valor que el campo Código del registro activo de la tabla Carrera
  Abrir posibilitando, de esta manera, procesar todos los libros de una carrera antes de salir de ese
  Leer bucle. Una vez que sale imprime los datos y lee la siguiente carrera en el AD Carreras. Luego,
  Mient vuelve a entrar con lo que comienza a procesar todos los libros de la nueva carrera y así
  # sucesivamente. Esto permite procesar grupos de libros (en el mientras interno) discriminados
  Co por carrera.

  Mientras ( NOT Préstamo.EOF AND Préstamo.Carrera = Carrera.Código )
    # Mientras no cambie la carrera actual en el AD Préstamos
    Si ( Préstamo.Devuelto = F )
      ContCarrera = ContCarrera + 1
    Fin Si
    Leer AD Préstamos
  Fin Mientras
  Imprimir Carrera.Nombre, ContCarrera,SL, SL
  Leer AD Carreras
Fin Mientras
Cerrar AD Préstamos
Cerrar AD Carreras
Fin Procedimiento

```

**Recuerde:** Para que este y cualquier otro corte de control funcione, los agregados de datos relacionados tienen que estar ordenados por el mismo campo. Este ejemplo funciona sólo si los registros de ambos ADD están ordenados por carrera.

#### Libro prestados a un estudiante en particular

El siguiente es el modelo del listado que se quiere imprimir:

Libros Prestados al Alumno: 4453	
Fecha	Devuelto
03/07/15	No
07/09/15	No

Listado de libros prestados a un estudiante en particular (especificando el legajo del mismo), indicando Fecha del préstamo y Estado de devolución (devuelto o no).

```

Procedimiento PrestamosAlumno ( Variable Numérica Leg )
  Imprimir "Libros Prestados al Alumno Legajo:" & Leg, SL, SL
  Imprimir "Fecha _____ Devuelto", SL
  Abrir AD Préstamos
  Leer AD Préstamos
  # Alternativa que valida si el legajo buscado (Leg) es igual al legajo
  # que se encuentra en el archivo (Préstamos.Fecha)
  Mientras ( NOT Préstamos.EOF )
    Si ( Préstamo.Legajo = Leg )
      Imprimir Préstamo.Fecha
      # Valida el contenido del campo Préstamo.Devuelto para colocar el
      # mensaje correspondiente
      Si ( Préstamo.Devuelto = V )
        Imprimir "Sí"
      Si No
        Imprimir "No"
      Fin Si
      Imprimir Salto de Línea
    Fin Si
    Leer AD Préstamos
  Fin Mientras
  Cerrar AD Préstamos
Fin Procedimiento

```

### Informe estadístico de préstamos por carrera

El siguiente es el modelo del listado que se quiere imprimir:

<b>Informe estadístico de préstamos</b>			
<b>Carrera</b>	<b>Prestados</b>	<b>Devueltos</b>	<b>Índice Devolución</b>
Informática	132	98	74.25 %
Marketing	226	143	63.27 %
Management	103	45	43.69 %
Publicidad	56	20	35.71 %
Recursos Humanos	72	71	98.61 %
<b>TOTALES</b>	<b>589</b>	<b>377</b>	<b>64.01 %</b>

Informe estadístico de préstamos por modelo de salida del reporte:

```

Procedimiento EstadísticaPorCarrera ( )
  Variable Numérica ContCarrera    #Cantidad Prestada a la carrera actual
  Variable Numérica ContDevueltos #Cantidad Devueltos por la carrera actual
  Variable Numérica TotPrestados   #Acumulador Total Prestados
  Variable Numérica TotDevueltos   #Acumulador Total Devueltos
  Variable Numérica Porcentaje     #Auxiliar para cálculo de Porcentaje
  Imprimir "Libros Sin Devolver Por Carrera", SL, SL
  Imprimir "Carrera      Prestados      Devueltos      Índice Devolución", SL
  Abrir AD Carreras
  Leer AD Carreras
  Abrir AD Préstamos
  Leer AD Préstamos
  Mientras ( NOT Carreras.EOF )
    # Contar los libros de la carrera actual
    ContCarrera = 0
    ContDevueltos = 0
    Mientras ( NOT Préstamos.EOF AND Préstamo.Carrera = Carrera.Código )
      ContCarrera = ContCarrera + 1
      Si ( Préstamo.Devuelto = V )
        ContDevueltos = ContDevueltos + 1
      Fin Si
      Leer AD Préstamos
    Fin Mientras
    Si ContCarrera >0
      Porcentaje = (ContDevueltos*100) / ContCarrera
    Si No
      Porcentaje = 100
    Fin Si
    Imprimir Carrera.Nombre, ContCarrera, ContDevueltos, Porcentaje, SL
    TotPrestados= TotPrestados + ContCarrera
    TotDevueltos= TotDevueltos + ContDevueltos
    Leer AD Carreras
  Fin Mientras
  Cerrar AD Préstamos
  Cerrar AD Carreras
  Imprimir " ", SL
  Si TotPrestados > 0
    Porcentaje = (TotDevueltos*100) / TotPrestados
  Si No
    Porcentaje = 100
  Fin Si
  Imprimir "TOTALES", TotPrestados, TotDevueltos, Porcentaje
Fin Procedimiento

```

De este modo, podemos ver paso a paso el desarrollo del proceso, para la obtención de los datos solicitados.

## SP3 / Ejercicio por resolver

El APROSS \*1.1, cuenta con un sistema donde están registrados los datos de todos los afiliados.

En este momento, los afiliados están agrupados en redes, que determinan los centros de atención a los que se puede acceder. Además cada afiliado puede obtener diferentes tipos de prestaciones de salud, para ellos requiere una orden:

- **Orden de Consulta:** para hacer una consulta o control médico.
- **Orden de Práctica Bioquímica:** para realizar cualquier tipo de análisis, por ejemplo: de sangre.
- **Orden de Prácticas Complementarias:** para estudios más específicos, como por ejemplo un electrocardiograma.
- **Recetarios:** para adquirir medicamentos en una farmacia autorizada.

Se le solicita que:

1. Desarrolle el pseudocódigo completo de un programa que permita procesar los siguientes agregados de datos de la obra social APROSS.

### Agregados de datos Apross

PL2

[Ver todas las Referencias](#)

**Ordenamiento:** El AD se encuentra ordenado por código de afiliado.

#### 1: AD Afiliado

Código	Nombre	Red	Integrantes
10	Guzmán Claudio	Naranja	1
15	Pereyra Victor	Azul	2
21	García Guillermo	Verde	5
47	Lopez Claudio	Naranja	2
56	Pérez Ignacio	Magenta	1
84	Ramirez Gustavo	Naranja	3
93	Soldano Sergio	Verde	6

EOF

#### 2: AD Ordenes

Código	Fecha	Tipo	Importe
10	01/12/2015	C	70,00
10	02/12/2015	B	450,00
10	02/12/2015	P	512,30
10	02/12/2015	R	185,90
10	06/12/2015	C	70,00
10	08/12/2015	C	70,00
15	01/12/2015	R	396,00
15	02/12/2015	C	70,00
15	02/12/2015	R	496,10
15	05/12/2015	B	150,00
21	02/12/2015	C	70,00

EOF

### **Referencia AD Afiliado**

**Código:** Número identificador del afiliado titular.

**Nombre:** Apellido y nombre del titular.

**Red:** red a la que pertenece el titular. Son cinco: Naranja, Magenta, Verde, Violeta y Azul.

**Integrantes:** cantidad de integrantes del grupo familiar.

### **Referencia AD Ordenes**

**Código:** Número identificador de cada afiliado titular.

**Fecha:** Día, mes y año en que se realizó una prestación de salud.

**Tipo:** Tipo de orden utilizada. Están codificadas:

C: Orden de Consulta.

B: Orden para práctica bioquímica.

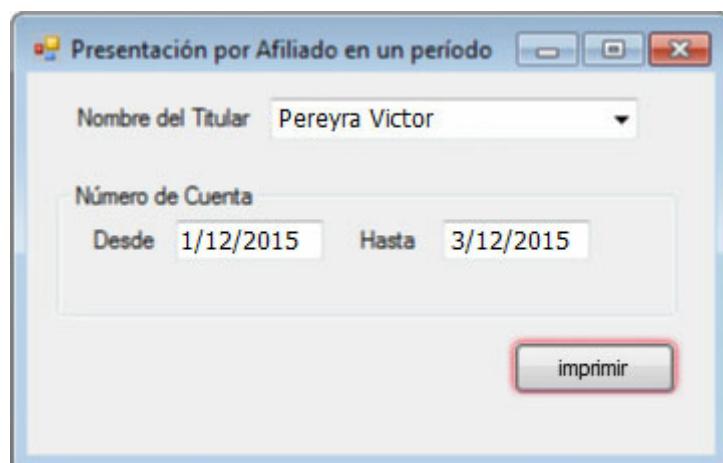
P: Orden de práctica complementaria.

R: Recetario de medicamentos.

**Importe:** Importe facturado a IPAM en la orden.

El AD se encuentra ordenado por código de afiliado (Primer Campo de la Clave) Fecha (Segundo Campo de la Clave) Observe, por ejemplo, que los registros correspondientes al día 02/12/2015 no están ubicados en forma consecutiva.

El programa deberá permitirle al usuario emitir un reporte que requerirá de la siguiente interfaz gráfica, en la que el usuario especificará la información necesaria para producirlo:



El reporte impreso respetará el siguiente modelo:

**Reporte de Prestaciones por Afiliado:**

Desde: 01/12/2015 Hasta: 03/12/2015

Titular: Pereyra Victor

Red: Azul

Fecha		Importe
<i>Tipo Orden</i>		
01/12/15	Recetario	\$ 396,00
02/12/15	Consulta	\$ 70,00
02/12/15	Recetario	\$ 496,10

Cantidad de Prestaciones: 3  
Importe Total: \$ 962,10

# REFERENCIAS 1

## 1.1 : APROSS

Administración provincial del Seguro de Salud (Córdoba).

Es la obra social de la provincia.

---



¡Vamos a comprobar cuánto aprendiste!

**1. Indique la opción correcta**

Clave Única es una clave compuesta por campos cuyos valores jamás se repiten para dos registros diferentes, es decir, que identifica de manera biunívoca a cada registro de la Tabla o Agregado de Datos.

- Verdadero
- Falso

**2. Indique la opción correcta**

Si se trabaja con conjuntos de archivos que están relacionados entre sí se utiliza una técnica que se denomina apareo de archivos.

- Verdadero
- Falso

**3. Indique la opción correcta**

Para recorrer una matriz se requiere de dos estructuras repetitivas.

- Verdadero
- Falso

**4. Indique la opción correcta**

Los agregados de datos se acceden secuencialmente.

- Verdadero
- Falso

**5. Indique la opción correcta**

Para procesar agregados de datos se debe tratar cada posición de un archivo con el equivalente de otro archivo.

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Clave Única es una clave compuesta por campos cuyos valores jamás se repiten para dos registros diferentes, es decir, que identifica de manera biunívoca a cada registro de la Tabla o Agregado de Datos.

- Verdadero
- Falso

## 2. Indique la opción correcta

Si se trabaja con conjuntos de archivos que están relacionados entre sí se utiliza una técnica que se denomina apareo de archivos.

- Verdadero
- Falso

## 3. Indique la opción correcta

Para recorrer una matriz se requiere de dos estructuras repetitivas.

- Verdadero
- Falso

## 4. Indique la opción correcta

Los agregados de datos se acceden secuencialmente.

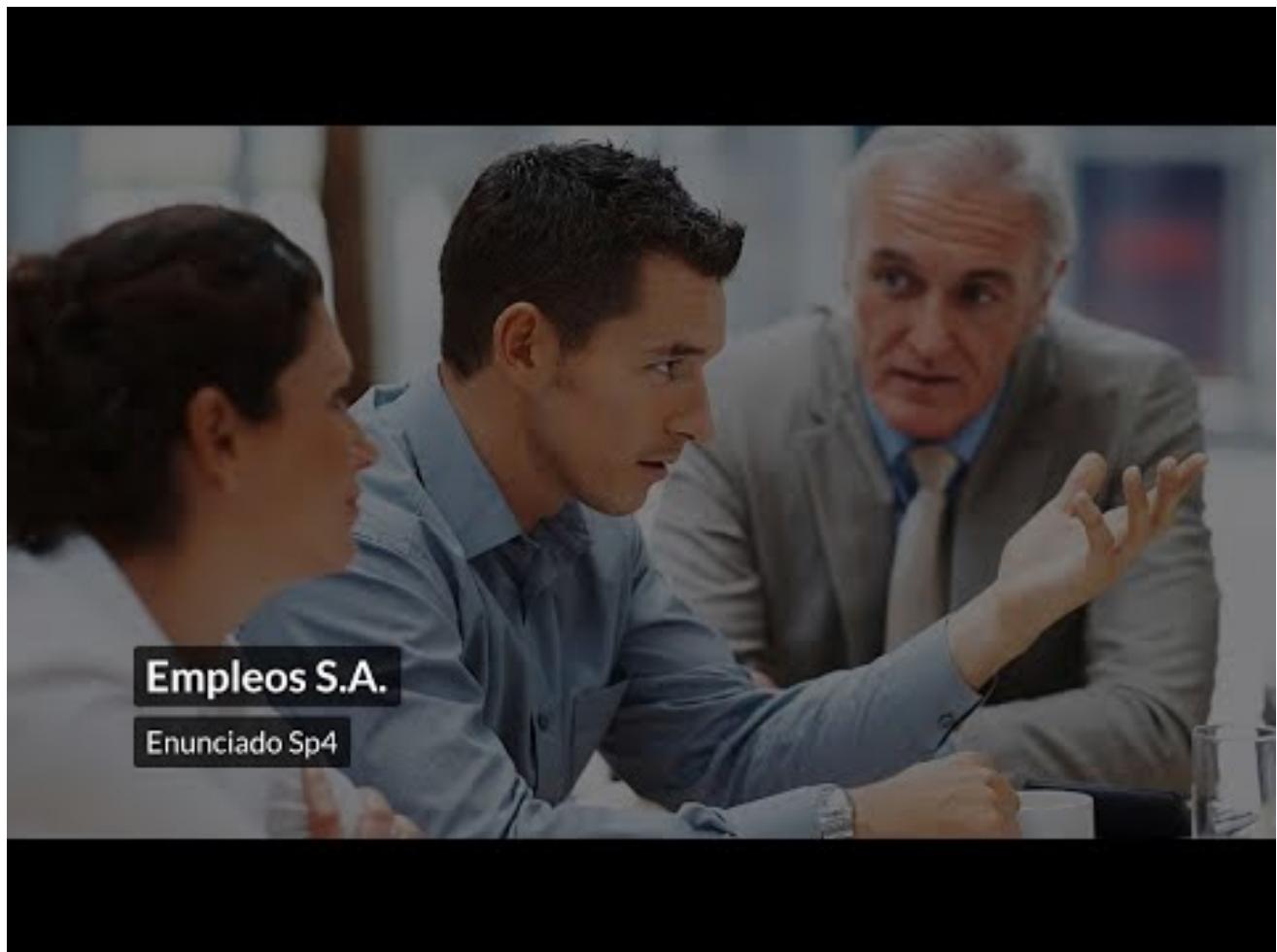
- Verdadero
- Falso

## 5. Indique la opción correcta

Para procesar agregados de datos se debe tratar cada posición de un archivo con el equivalente de otro archivo.

- Verdadero
- Falso

## Situación profesional 4: EMPLEOS S.A.



Video "Enunciado SP4" | Elaboración Propia. DEPROE - Colegio Universitario IES

"Empleos S.A." es una empresa de nuestra ciudad cuenta con un AD con los datos de sus empleados. En este AD se registra el Legajo, Apellido y Nombre, CUIT, Fecha de Ingreso y Sueldo Básico de cada empleado.

La empresa solicita una interfaz que le permita el ingreso de nuevos empleados, la modificación de datos ya existentes y la baja de legajos.

Como asistente de sistemas se le encarga el desarrollo de los procedimientos necesarios para el mantenimiento del AD Secuencial Empleados:

# SP4 / H1: Actualización de archivos

Ya vimos en la SP1 / H1: Registros y Agregados de Datos las operaciones básicas que se pueden realizar con un agregado de datos secuencial (AD): apertura, lectura y cierre de los mismos. También vimos que un **AD** se recorre de manera secuencial, y que su recorrido culmina cuando encontramos el final de archivo conocido como **EOF**.

Agregaremos a estas operaciones ya vistas, las referidas a la actualización del contenido de un agregado de dato, también conocidas como operaciones de mantenimiento de los mismos. Estas son:

- Alta de nuevos registros.
- Baja de registros.
- Modificación de registros.

*Para buscar, eliminar, insertar o modificar un registro, obligatoriamente debe abrir el AD.*

En cuanto a las operaciones con AD, usaremos para estas acciones, la acción de escribir. Al usar esta operación escribiremos en la posición actual en la que estamos posicionados en el AD.

Presentaremos tres modos para actualizar los archivos:

- a) La actualización de archivos secuencial.
- b) La actualización de archivos mediante Acceso Directo.
- c) La actualización de archivos mediante Indexación.

## a) Actualización de archivos secuencial

### 1. Alta de nuevos registros

Una operación de alta o inserción consiste en la inserción de un nuevo registro al AD. Considerando que el AD está ordenado por un campo clave, insertaremos el nuevo registro al final del mismo luego del último registro contenido.

Por ejemplo, si necesitamos dar de alta un nuevo cliente en el AD Clientes. Ingresaremos los datos del nuevo cliente en la interfaz gráfica e iniciaremos el saldo en cero. Para la inserción recorreremos el AD hasta encontrar el final de archivo (EOF) y ahí agregamos el nuevo registro.

```

Programa Alta_de_Cliente
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica      Codigo
        Variable Alfanumérica Nombre
        Variable Numérica      Saldo
    Fin Registro
    # Definición de la Interfaz
    Formulario frmAltaCliente
        Marco Datos
            Caja de texto txtCodigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdAlta
    Fin formulario
    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
    # Desarrollo de los Procedimientos de Evento
    Procedimiento cmdAlta:Click
        Registro Nuevo Tipo Registro Cliente
            # Cargo los valores de la interfaz
            Nuevo.Codigo = txtCodigo
            Nuevo.Nombre = txtNombre
            Nuevo.Saldo = 0
            # Apertura y lectura del Agregado de Datos
            Abrir AD Clientes
            Leer AD Clientes
            # Recorremos el AD hasta encontrar el final de archivo
            Mientras ( NOT Clientes.EOF )
                Leer AD Clientes
            Fin mientras
            # Grabamos el nuevo registro luego del último registro del AD
            Escribir AD Nuevo
            # Cerramos el AD Clientes
            Cerrar AD Clientes
        Fin Procedimiento
    Fin Programa

```

Al igual que en este punto, ***en nuestra Situación profesional*** nos será de gran utilidad esta herramienta puesto que contamos con un agregado de datos secuencial de empleados por lo que debemos ofrecer la posibilidad de que, a través de nuestra interfaz, se pueda dar de ***alta a nuevos empleados***. Para ello deberemos, en primer lugar, crear una estructura de registro con los datos que guardaremos de los empleados y, luego, adaptar este algoritmo para que cumpla con el objetivo de dar de alta a nuevos empleados.

El procedimiento correspondiente al botón de alta quedaría de la siguiente manera:

```
Procedimiento cmdAlta:Click
  Registro NvoEmpleado Tipo Registro Empleado
  # Cargo los valores de la interfaz
  NvoEmpleado.legajo = txtLegajo
  NvoEmpleado.nombre = txtNombre
  NvoEmpleado.CUIT = txtCUIT
  NvoEmpleado.ingreso = txtIngreso
  NvoEmpleado.sueldo = txtSueldo
  # Apertura y lectura del Agregado de Datos
  Abrir AD Empleados
  Leer AD Empleados
  # Recorremos el AD hasta encontrar el final de archivo
  Mientras ( NOT Empleados.EOF )
    Leer AD Empleados
  Fin mientras
  # Grabamos el nuevo registro luego del último registro del AD
  Escribir AD NvoEmpleado
  # Cerramos el AD Empleados
  Cerrar AD Empleados
Fin procedimiento
```

## 2. Baja de registros

Baja es la acción de eliminar un registro de un AD. La baja del registro puede ser lógica o física. Una baja lógica no es un borrado del registro, sino que se determina la baja por medio de un campo del registro. La baja física implica el borrado y desaparición del registro. A continuación vemos cómo se hace:

```

Programa Baja_de_Cliente
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica      Código
        Variable Alfanumérica  Nombre
        Variable Numérica      Saldo
    Fin Registro
    # Definición de la Interfaz
    Formulario frmBajaCliente
        Marco Datos
            Caja de texto txtCódigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de Comando cmdBaja
    Fin formulario
    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
    # Desarrollo de los Procedimientos de Evento
    Procedimiento cmdBaja:Click
        # Apertura y lectura del Agregado de Datos
        Abrir AD Clientes
        Leer AD Clientes
        # Recorremos el AD hasta tanto encontremos el registro que deseamos dar
        # de baja o hasta encontrar el final de archivo
        Mientras ( NOT Clientes.EOF AND txtCódigo <> Clientes.Código)
            Leer AD Clientes
        Fin mientras
        # Corroboramos que realmente encontramos el registro a eliminar
        Si txtCódigo = Clientes.Código
            # Eliminamos el registro
            Eliminar Registro AD Clientes
        Sino
            Imprimir "El cliente que desea eliminar no existe"
        Fin Si
        # Cerramos el AD Clientes
        Cerrar AD Clientes
    Fin Procedimiento
Fin programa

```

Al igual que con la herramienta anterior, **para resolver nuestra Situación profesional**, modificaremos este algoritmo para adaptarlo al caso de baja de empleados que deberemos tener en nuestra interfaz. El procedimiento resuelto de la baja de un empleado sería el siguiente:

```

Procedimiento cmdModificar:Click
    # Apertura y lectura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados
    # Recorremos el AD hasta tanto encontremos el registro que
        deseamos modificar o hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
        Leer AD Empleados
    Fin mientras
    # Corroboramos que realmente encontramos el registro a eliminar
    Si txtLegajo = Empleados.Legajo
        # Eliminamos el registro
        Eliminar Registro AD Empleados
    Sino
        Imprimir "El empleado que desea eliminar no existe"
    Fin Si
    # Cerramos el AD Clientes
    Cerrar AD Empleados
Fin procedimiento

```

### 3. Modificación de registros

Una modificación en un AD consiste en la operación de cambiar todo o parte del contenido de un registro se denomina modificación de un AD. Un ejemplo claro sería que, si contamos con AD de Clientes y necesitamos cambiar la dirección o el número de teléfono de uno de los clientes (por ende de uno de los registros):

```

Programa Modificación_de_Cliente
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica Código
        Variable Alfanumérica Nombre
        Variable Numérica Saldo
    Fin Registro
    # Definición de la Interfaz
    Formulario frmAltaCliente
        Marco Datos
            Caja de texto txtCódigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdBaja
    Fin formulario
    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
    # Desarrollo de los Procedimientos de Evento
    Procedimiento cmdModificación:Click
        # Apertura y lectura del Agregado de Datos
        Abrir AD Clientes
        Leer AD Clientes
        # Recorremos el AD hasta tanto encontremos el registro que
        # deseamos modificar o hasta encontrar el final de archivo
        Mientras ( NOT Clientes.EOF AND txtCódigo <> Clientes.Código)
            Leer AD Clientes
        Fin mientras
        # Corroboramos que realmente encontramos el registro a modificar
        Si txtCódigo = Clientes.Código
            # Modificamos el registro
            Clientes.Código = txtCódigo
            Clientes.Nombre = txtNombre
            Clientes.Saldo = txtSaldo
            Actualizar Registro AD Clientes
        Sino
            Imprimir "El cliente que desea modificar no existe"
        Fin Si
        # Cerramos el AD Clientes
        Cerrar AD Clientes
    Fin Procedimiento
Fin programa

```

*En nuestra Situación profesional* esta será también una de las herramientas que utilizaremos. Modificaremos la estructura a la de empleados y adaptaremos este algoritmo **para poder modificar los datos de un empleado** ya guardado en nuestro AD.

El pseudocódigo del procedimiento de modificación será:

```

Procedimiento cmdModificar:Click
    # Apertura y lectura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados

    # Recorremos el AD hasta tanto encontremos el registro que
    # deseamos modificar o hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
        Leer AD Empleados
    Fin mientras

    # Corroboramos que realmente encontramos el registro a modificar
    Si txtLegajo = Empleados.Legajo
        # Modificamos el registro
        Empleados.Legajo = txtLegajo
        Empleados.Nombre = txtNombre
        Empleados.CUIT = txtCUIT
        Empleados.Ingreso = txtIngreso
        Empleados.Sueldo = txtSueldo
        Actualizar Registro AD Empleados
    Sino
        Imprimir "El empleado que desea modificar no existe"
    Fin Si

    # Cerramos el AD Clientes
    Cerrar AD Empleados

Fin procedimiento

```

## b) Actualización de archivos mediante Acceso Directo

### 1. Acceso directo a registros

Se dice que un archivo es de acceso directo cuando se puede acceder a cualquier registro de manera directa especificando un índice del registro en base al origen del AD. La lectura/escritura de un registro es rápida, ya que se accede directamente al registro y no se necesitan recorrer los anteriores.

Hasta el momento hemos realizado acciones con AD de manera secuencial, *ahora veremos AD con organización de manera directa*. Un AD está organizado en modo directo cuando los registros están ordenados en un orden lógico que no se corresponde con el orden físico de los mismos. Los registros se sitúan en el AD y se accede a ellos de manera directa mediante su posición, es decir, el lugar relativo que ocupan.

La **ventaja** de este modo de organización es que los AD se pueden leer y escribir en cualquier orden y posición, lo que da una mayor velocidad de acceso a los datos.

Como **desventaja** de este modo señalamos la necesidad de programar la relación que existe entre el contenido de un registro y la posición que ocupa. El acceso directo a registros da la posibilidad de que queden huecos libres entre registros.

La correspondencia entre la clave de un registro y la dirección del mismo debe ser programada y la relación entre el registro y su posición física se obtiene mediante un algoritmo de Tratamiento por Transformación de

clave, que veremos con detalle en esta unidad.

**Para que un AD pueda organizarse de manera directa debe cumplir las siguientes condiciones:**

- Almacenar en un soporte direccionable (como un disco magnético).
- Todos los registros deben contener un campo clave que identifica a cada registro de modo único.
- Debe existir una correspondencia entre los valores posibles de la clave y las direcciones disponibles para almacenar los registros.

En un soporte direccionable, cada posición se localiza con su dirección absoluta. En la práctica, el programador no gestiona directamente las direcciones absolutas, sino que gestiona las direcciones relativas respecto al principio del archivo.

El programador crea una relación perfectamente dirigida entre la clave de cada registro y la posición física dentro del dispositivo. Para esto genera un algoritmo de direccionamiento que hace uso de una función de conversión de claves o función hash. Suponiendo que N es el número de posiciones disponibles para el AD, el algoritmo de direccionamiento convierte cada valor de la clave en una dirección relativa dr, comprendida entre 1 y N. El algoritmo de conversión de claves debe eliminar o reducir al máximo el número de colisiones.

Una colisión se presenta cuando dos registros de claves distintas producen la misma dirección física. Debido a que se debe situar al registro en un lugar diferente al obtenido por el algoritmo de conversión, el acceso al mismo será mucho más lento. Estas colisiones son difíciles de evitar, pero si se puede dar un tratamiento adecuado en las operaciones de lectura y de escritura que se realizan sobre el AD.

Para representar la función de transformación de claves hash, utilizaremos una notación matemática. Por lo tanto, si K es una clave, f(K) es la correspondiente dirección; f es la llamada función de conversión.

**Para entender estos nuevos conceptos, analicemos el siguiente ejemplo:**

Una empresa que cuenta con un plantel de vendedores, y un archivo en el que cada registro corresponde a un vendedor. Considerando que existen 100 vendedores, cada uno de ellos con un legajo de 5 dígitos.

Para el diseño de un archivo que contenga estos registros, crearemos 125 registros (25% más de los que necesitamos) y que distribuiremos de la siguiente manera:

- De la posición 0 a la 99 la consideramos área principal y en ella se almacenan los registros de los vendedores.
- Y las posiciones que van de la posición 100 a la 124 forman el área de desbordamiento, si  $K(1) <> K(2)$ , pero  $f(K(1)) = f(K(2))$ , y el registro con clave K(1) ya está almacenado en el área principal, entonces el registro K(2) lo almacenamos en el área de desbordamiento.

*La función f se define como:*

$f(K) = \text{resto cuando } K \text{ se divide por } 99$ , esto es, el módulo de 99; 99 ha sido elegido por ser el mayor número que puede tener una posición en el área principal.

Para establecer el archivo se borrarán primero las 125 posiciones.

Luego, para cada registro de vendedor se calcula  $p = f(K)$ . Si la posición "p" está vacía, se almacena el registro en ella. En caso de que este ocupado se busca secuencialmente a través de las posiciones 100, 101, ..., 125 (área de desborde) para el registro con la clave deseada.

## **2. Tratamiento por transformación de clave (HASHING)**

El método de transformación de clave consiste en transformar un número de orden (clave) en direcciones de almacenamiento por medio de un algoritmo de conversión.

Cuando las altas se realizan por el método de transformación de clave, la dirección donde introducir un determinado registro se conseguirá por la aplicación a la clave del algoritmo de conversión (HASH). Si encontráramos que dicha dirección ya está ocupada, el nuevo registro deberá ir a la zona de sinónimos o excedentes.

### 3. Operaciones con AD directos

Veremos solo algunas operaciones que se realizan con AD de acceso directo, y son las siguientes:

- Alta por transformación de claves.

La operación de alta en un AD directo consiste en ir introduciendo los sucesivos registros en una determinada posición, especificada a través de un índice. Mediante el índice nos posicionaremos directamente sobre la ubicación del AD que se encuentra en la posición  $(índice - 1) * tamaño\_de\_registro$  y escribimos ahí nuestro registro.

Siguiendo con el ejemplo de los clientes. Supongamos que deseamos dar de alta un registro en un archivo creado para almacenar 100 clientes. Lo que debemos hacer es:

```

Programa Alta_de_Cliente_TransClave
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica     Código
        Variable Alfanumérica Nombre
        Variable Numérica     Saldo
        Variable Alfanumérica Ocupado
    Fin Registro
    Variable numérica finDatos = 99
    Variable numérica tope = 125
    # Definición de la Interfaz
    Formulario frmAltaCliente
        Marco Datos
            Caja de texto txtCódigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdAlta
    Fin formulario
    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente Modo Acceso Directo
    # Desarrollo de los Procedimientos de Evento
    Procedimiento cmdAlta:Click
        Variable Alta Tipo Registro Cliente
        Variable Lógica hayLugar
        Variable numérica posición
        # Cargo los valores de la interfaz
        Alta.Código = txtCódigo
        Alta.Nombre = txtNombre
        Alta.Saldo = txtSaldo
        Alta.Ocupado = ""
        # Apertura el Agregado de Datos
        Abrir AD Clientes
        # Obtenemos la posición en donde escribiremos usando la función HASH
        posición = HASH (Alta.Código)
        # Leemos la posición obtenida para verificar si está disponible. La
        # sentencia estará compuesta por en primer lugar por el nombre del AD y
        # separado por una "," pondremos la posición que guardamos en la variable
        # para tal fin.
        Leer AD Clientes, posición
        Si Clientes.Ocupado = "*"
            hayLugar = FALSO
            posición = finDatos

```

```

# Recorremos el área de excedentes hasta encontrar un lugar donde
# grabar
Mientras (posicion < tope AND NOT hayLugar)
    posicion = posicion +1
    Leer AD Clientes, posicion
    Si Clientes.Ocupado = "*"
        hayLugar = TRUE
    Fin Si
Fin mientras
Si no
    hayLugar = TRUE
fin Si
# Preguntamos si tenemos lugar en donde grabar
Si hayLugar
    # Marcamos el campo como ocupado
    Alta.Ocupado = "*"
    # Grabamos el nuevo registro
    Escribir AD Alta, posicion
Si no
    Imprimir "No hay lugar para este nuevo registro"
Fin Si
# Cerramos el AD Clientes
Cerrar AD Clientes
Fin Procedimiento
# Desarrollamos la función HASH
Función HASH (Codigo)
    Variable numérica posí
    # Como se explico anteriormente se calcula el módulo con respecto al
    # mayor número posible.
    posí = Codigo MOD finDatos
    retornar posí
Fin Función
Fin programa

```

- **Consulta por transformación de claves.**

Puede ocurrir que la clave o código por el que deseamos acceder a un determinado registro no coincida con la posición de dicho registro en el archivo, aunque guarden entre sí una cierta relación, pues al escribir los registros en el archivo la posición se obtuvo aplicando a la clave un algoritmo de conversión.

En este caso es imprescindible el almacenamiento de la clave en uno de los campos del registro y las operaciones a realizar para llevar a cabo una consulta sería:

- Definir la clave del registro buscado.
- Aplicar algoritmo de conversión de clave a dirección.
- Lectura del registro ubicado en la dirección obtenida.
- Comparación de las claves de los registros leídos y buscados y, si son distintos, exploración secuencial del área de excedentes.

Si tampoco se encuentra el registro en el área de excedentes entonces podemos afirmar que el registro buscado no existe.

```

Programa Consulta_de_Cliente_TransClave
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica     Código
        Variable Alfanumérica Nombre
        Variable Numérica     Saldo
        Variable Alfanumérica Ocupado
    Fin Registro

    Variable numérica finDatos = 100
    Variable numérica tope = 125

    # Definición de la Interfaz
    Formulario frmConsultaCliente
        Marco Datos
            Caja de texto txtCódigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdConsulta
    Fin formulario

    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
        # Desarrollo de los Procedimientos de Evento
        Procedimiento cmdConsulta:Click
            Variable Lógica     encontrado
            Variable numérica   posición

            # Apertura y lectura del Agregado de Datos
            Abrir AD Clientes
            Leer AD Clientes

            # Calculamos la posición a consultar
            posición = HASH (txtCódigo)
            # Leemos la posición obtenida
            Leer AD Clientes, posición

```

```

# Verificamos si encontramos el registro
Si Clientes.Ocupado <> "*" OR Clientes.Codigo <> txtCodigo
    encontrado = FALSO
    posición = finDatos
    # Si no encontramos en la posición calculada buscamos
    # secuencialmente en el área de excedentes
    Mientras posicion < tope AND NOT encontrado
        posicion = posicion +1
        Leer AD Clientes, posicion
        Si Clientes.Ocupado = "*" AND Clientes.Codigo = txtCodigo
            encontrado = TRUE
        Fin Si
    Fin mientras
Si no
    encontrado = TRUE
Fin Si
# Verificamos si encontramos el registro y lo mostramos.
Si encontrado
    Imprimir Clientes.Codigo & " " Clientes.nombre & " " & Clientes.Saldo"
Si no
    Imprimir "El registro que está buscando no existe."
Fin Si
# Cerramos el AD Clientes
Cerrar AD Clientes
Fin Procedimiento
# Desarrollamos la función HASH
Función HASH (Codigo)
    Variable numérica posi
    # Como se explico anteriormente se calcula el módulo con respecto al
    # mayor número posible.
    posi = Codigo MOD finDatos
    retornar posi
Fin Función
Fin programa

```

- **Baja por transformación de claves.**

En el proceso de bajas se considera el contenido de un campo indicador, por ejemplo,

persona.ocupado, que, cuando existe información válida en el registro está marcado con un "\*".

Para dar de baja al registro, es decir, considerar su información como no válida, eliminaremos dicho \*. Este tipo de baja es una baja lógica.

Desarrollaremos a continuación un algoritmo que realice bajas lógicas y acceda a los registros a los que se desea dar de baja por el método de transformación de clave.

```

Programa Baja_de_Cliente_TransClave
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica Codigo
        Variable Alfanumérica Nombre
        Variable Numérica Saldo
        Variable Alfanumérica Ocupado
    Fin Registro

    Variable numérica finDatos = 100
    Variable numérica tope = 125

    # Definición de la Interfaz
    Formulario frmBajaCliente
        Marco Datos
            Caja de texto txtCodigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdBaja
    Fin formulario

    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
        # Desarrollo de los Procedimientos de Evento
        Procedimiento cmdBaja:Click
            Variable Lógica encontrado
            Variable numérica posicion

```

```

# Apertura y lectura del Agregado de Datos
Abrir AD Clientes
# calculamos la posición del registro a dar de baja
posicion = HASH(txtCodigo)
Leer AD Clientes, posicion
# Verificamos si encontramos el registro en la posición calculada
Si Clientes.ocupado <> "*" OR Clientes.Codigo <> txtCodigo
    encontrado = FALSO
    posicion = finDatos
    # Si no encontramos en la posición calculada buscamos
    # secuencialmente en el área de excedentes
    Mientras (posicion < tope AND NOT encontrado)
        posicion = posicion + 1
        Leer AD Clientes, posicion
        Si Clientes.Ocupado = "*" AND Clientes.Codigo = txtCodigo
            encontrado = TRUE
        Fin Si
    Fin Mientras
Si no
    encontrado = TRUE
Fin Si
# Verificamos si encontramos el registro y lo mostramos
Si encontrado
    # Marcamos el campo como disponible
    Clientes.ocupado = ""
    # Escribimos el registro con la modificación
    Actualizar Registro AD Clientes, posicion
Si no
    Imprimir "El cliente que desea eliminar no se encuentra en el AD"
Fin Si
# Cerramos el AD Clientes
Cerrar AD Clientes
Fin Procedimiento

# Desarrollamos la función HASH
Función HASH (Codigo)
    Variable numérica posi
    # Como se explico anteriormente se calcula el módulo con respecto al
    # mayor número posible.
    posi = Codigo MOD finDatos
    retornar posi
Fin Función
Fin programa

```

- **Modificación por transformación de claves.**

En un archivo de acceso directo se localiza el registro que se desea modificar mediante la especificación del índice o aplicando el algoritmo de conversión de clave a dirección y, en caso necesario, la búsqueda en zona de colisiones se modifica el contenido y se reescribe.

```

Programa Modificacion_de_Cliente_TransClave
  # Definición de Datos Globales
  Registro Cliente
    Variable Numérica      Codigo
    Variable Alfanumérica  Nombre
    Variable Numérica      Saldo
    Variable Alfanumérica  Ocupado
  Fin Registro

  Variable numérica  finDatos = 100
  Variable numérica  tope = 125

  # Definición de la Interfaz
  Formulario frmModificarCliente
    Marco Datos
      Caja de texto txtCodigo
      Caja de texto txtNombre
      Caja de texto txtSaldo
    Fin marco
    Botón de comando cmdModificar
  Fin formulario

  # Definición del Agregado de Datos
  Agregado de Datos Clientes Tipo Registro Cliente

  # Desarrollo de los Procedimientos de Evento
  Procedimiento cmdModificar:Click
    Variable Lógica  encontrado
    Variable numérica  posicion

    # Apertura y lectura del Agregado de Datos
    Abrir AD Clientes
    # Calculamos la posición a modificar y la leemos
    posicion = HASH(txtCodigo)
    Leer AD Clientes, posición
    # Verificamos si encontramos el registro en la posición calculada
    Si Clientes.Ocupado <> "*" OR Clientes.Codigo <> txtCodigo
      encontrado = FALSO
      posicion = finDatos

```

```

# Si no encontramos en la posición calculada buscamos
# secuencialmente en el área de excedentes
Mientras (posicion < tope AND NOT encontrado)
    posicion = posicion + 1
    Leer AD Clientes, posicion
    Si Clientes.Ocupado = "*" AND Clientes.Codigo = txtCodigo
        encontrado = TRUE
    Fin Si
Fin Mientras
Si no
    encontrado = TRUE
Fin Si
# Verificamos si encontramos el registro y lo mostramos
Si encontrado
    # Grabamos las modificaciones
    Clientes.Codigo = txtCodigo
    Clientes.Nombre = txtNombre
    Clientes.Saldo = txtSaldo
    Clientes..Ocupado = "*"
    Actualizar Registro AD Clientes, posicion
Si no
    Imprimir "El cliente que desea modificar no se encuentra en el AD"
Fin Si

# Cerramos el AD Clientes
Cerrar AD Clientes

Fin Procedimiento

# Desarrollamos la función HASH
Función HASH (Codigo)
    Variable numérica posi
    # Como se explico anteriormente se calcula el módulo con respecto al
    # mayor número posible.
    posi = Codigo MOD finDatos
    retornar posi
Fin Función
Fin programa

```

En nuestra Situación profesional, no utilizaremos las herramientas de actualización mediante transformación de claves, ya que contamos con un AD de acceso secuencial.

## c) Actualización de archivos mediante Indexación (Índices)

### 1. Clave-Dirección

Con respecto a las transformaciones clave-dirección deberemos realizar aún algunas consideraciones.

En un soporte direccionable, cada posición se localiza por su dirección absoluta. Los AD directos manipulan direcciones relativas en lugar de absolutas, lo que hará al programa independientemente de la posición absoluta del archivo en el soporte. Los algoritmos de conversión de clave transformaran las claves en direcciones relativas. Suponiendo que existen N posiciones disponibles para el archivo, los algoritmos de

conversión de clave producirán una dirección relativa en el rango 1 a N por cada valor de la clave.

Existen varias técnicas para obtener direcciones relativas. En el caso en que dos registros distintos produzcan la misma dirección, se dice que se produce una colisión o sinónimo.

## 2. Tratamiento de las colisiones

Las colisiones son inevitables y, como se ha comentado, se originan cuando dos registros claves diferentes producen la misma dirección relativa. En estos casos las colisiones se pueden tratar de dos formas diferentes.

Supongamos que un registro **r1** produce una dirección **d1** que ya está ocupado ¿Dónde colocamos el nuevo registro?

Existen dos métodos básicos:

- Considerar una zona de excedentes y asignar el registro a la primera posición libre en dicha zona.  
Fue el método aplicado en los algoritmos anteriores.
- Buscar una nueva dirección libre en la zona de datos del AD.

## 3. Acceso a AD directos mediante indexación

La indexación es una técnica para el acceso a los registros de un AD. En esta técnica el AD principal de registros esta suplementado por uno o más índices. Los índices pueden ser archivos independientes o array (vectores) que se cargan al comenzar en la memoria del ordenador. En ambos casos estarán formados por registros con los campos código o clave y posición o número de registros.

El almacenamiento de los índices en memoria permite encontrar los registros más rápidamente que cuando se trabaja en disco.

Cuando utilizamos un AD indexado localizaremos los registros en el índice a través del campo clave y este devolverá la posición del registro en el AD principal, directo.

Las operaciones básicas a realizar con AD indexado son:

- Crear las zonas de índice y datos como archivos vacíos originales.
- Carga el archivo índice en memoria antes de utilizarlo.
- Reescribir el archivo índice desde memoria después de utilizarlo.
- Añadir registros al archivo de datos y al de índices.
- Borrar registros al archivo de datos.
- Actualizar registros en el archivo de datos.

Veamos, a continuación, un algoritmo de consulta de un AD mediante indexación:

```

Programa Consulta_de_Cliente_Index
    # Definición de Datos Globales
    Registro Cliente
        Variable Numérica Código
        Variable Alfanumérica Nombre
        Variable Numérica Saldo
    Fin Registro

    Registro Indice
        Variable Numérica Código
        Variable Numérica posición
    Fin Registro

    Variable numérica finDatos = 100
    Variable numérica tope = 125

    # Definición de la Interfaz
    Formulario frmConsultaCliente
        Marco Datos
            Caja de texto txtCódigo
            Caja de texto txtNombre
            Caja de texto txtSaldo
        Fin marco
        Botón de comando cmdConsulta
    Fin formulario

    # Definición del Agregado de Datos
    Agregado de Datos Clientes Tipo Registro Cliente
    Agregado de Datos Indices Tipo Registro Indice
    Variable Vector (tope)

```

```

# Desarrollo de los Procedimientos de Evento
Procedimiento cmdConsulta:Click
    Variable numérica i
    Variable numérica Largo
    Variable numérica ubic

    # Abrimos los AD con los que trabajaremos
    Abrir AD Clientes
    Abrir AD Indices
    # Calculamos el largo del AD y lo dibidimos por
    Largo = calcularLongitud(AD Clientes) / calcularLongitud(Indice)
    i = 1
    mientras i<= largo
        Vector(i) = Leer AD Indices, i
        i = I + 1
    fin mientras
    # Cerramos el AD Indices
    Cerrar AD Indices
    # Buscamos el código buscado en el vector y obtenemos la posición en la
    # variable ubic
    Busqueda_Binaria(Vector(), Largo, txtCodigo, ubic, encontrado)
    # Si encontramos el código, mostramos los datos.
    Si encontrado
        Leer AD Clientes, Vector(ubic).posicion
        Imprimir Clientes.Codigo & " " & Clientes.Nombre & "$" & Clientes.Saldo
    Sino
        Imprimir "El cliente que desea consultar no está contenido en el AD"
    Fin Si
    # Cerramos el AD Clientes
    Cerrar AD Clientes
Fin Procedimiento
Fin programa

```

#### 4. Altas en AD directos mediante indexación

Un procedimiento necesario para dar de alta a nuevos clientes en el archivo antes visto sería:

```

Procedimiento Alta_Clientes_Index(Vector(), largo)
    Variable numérica p
    Variable lógica encontrado
    Variable numérica num

    # Cargo los valores de la interfaz
    Alta.Codigo = txtCodigo
    Alta.Nombre = txtNombre
    Alta.Saldo = 0

    # Verificamos si tenemos lugar para dar de alta, recordemos que n es una
    # variable global
    Si n = tope
        Imprimir "El AD se encuentra lleno"
    sino
        encontrado = FALSO
        # Buscamos el código buscado en el vector y obtenemos la posición en la
        # variable ubic
        busqueda_binaria(Vector(), n txtCodigo, p, encontrado)
        # Verificamos si el código ya existe
        Si encontrado
            Imprimir "El código que desea dar de alta ya se encuentra en el AD"
        Sino
            # Calculamos la posición en donde se encontrará el nuevo registro
            num = calcularLongitud(AD Clientes) / calcularLongitud(Clientes)+1
            # Agregamos en el Vector el nuevo registro manteniendo la
            # ordenación del mismo.
            alta_indice(Vector, n, p, txtCodigo, num)
            n = n + 1
            # Escribimos el nuevo registro al final del AD Clientes
            Escribir AD Clientes , num, Alta
        Fin Si
        # En el programa principal crearemos un nuevo archivo de indices en
        # base a Vector()
    Fin Si
Fin Procedimiento

```

Al igual que la herramienta anterior, la actualización de AD mediante indexación no será utilizada en la resolución de nuestra Situación profesional debido a que contamos con un AD de acceso secuencial.



¿Te animás a medir cuánto aprendiste?

**1. Indique la opción correcta**

La baja de un registro puede ser:

- Lógica
- Física
- Las dos primeras son correctas

**2. Indique la opción correcta**

Una modificación en un AD consiste en:

- La operación de cambiar el nombre del AD
- La operación de cambiar todo o parte del contenido de un registro
- La operación de cambiar solo el campo clave de un registro

**3. Indique la opción correcta**

Las operaciones de mantenimiento sobre un AD son la apertura, la lectura y el cierre.

- Verdadero
- Falso

**4. Indique la opción correcta**

Un archivo es de acceso directo cuando se puede acceder a cualquier registro de manera directa especificando la clave de un registro

- Verdadero
- Falso

**5. Indique la opción correcta**

El algoritmo de direccionamiento HASH convierte cada valor de la clave en una dirección relativa

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

La baja de un registro puede ser:

- Lógica
- Física
- X Las dos primeras son correctas

## 2. Indique la opción correcta

Una modificación en un AD consiste en:

- La operación de cambiar el nombre del AD
- X La operación de cambiar todo o parte del contenido de un registro
- La operación de cambiar solo el campo clave de un registro

## 3. Indique la opción correcta

Las operaciones de mantenimiento sobre un AD son la apertura, la lectura y el cierre.

- X Verdadero
- Falso

## 4. Indique la opción correcta

Un archivo es de acceso directo cuando se puede acceder a cualquier registro de manera directa especificando la clave de un registro

- Verdadero
- X Falso

## 5. Indique la opción correcta

El algoritmo de direccionamiento HASH convierte cada valor de la clave en una dirección relativa

- X Verdadero
- Falso

# SP4 / H2: Organización de archivos

## Procesamiento de AD secuenciales indexados

Los AD de organización secuencial indexada contienen tres áreas:

- Un área de datos que agrupa a los registros.
- Un área de índice que contiene los niveles de índice.
- Una zona de desbordamiento o excedentes para el caso de actualizaciones con adición de nuevos registros.

Los registros han de ser grabados obligatoriamente en orden secuencial ascendente por el contenido del campo clave; simultáneamente a la grabación de los registros, el sistema crea los índices.

Una consideración adicional con respecto a este tipo de organización es que es posible usar más de una clave, hablaríamos así de la clave primaria y de una o más secundarias. El valor de la clave primaria es la base para la posición física de los registros en el AD y debe ser única. Las claves secundarias pueden o no ser únicas y no afectan al orden físico de los registros.

## Métodos de ordenación sobre archivos

El tratamiento de los AD secuenciales exige que estos se encuentren ordenados respecto a un campo del registro, denominado campo clave.

Supongamos un archivo del personal de una empresa, cuya estructura de registro es la siguiente:

Metodo de ordenación sobre archivos		
AD Personal		
Nombre	tipo cadena	(nombre del empleado)
Dirección	tipo cadena	(dirección)
Fecha	tipo cadena	(fecha de nacimiento)
Salario	tipo numérico	(Salario)
Categoría	tipo cadena	(Categoría laboral)
DNI	tipo cadena	(número de DNI)

La clasificación en orden ascendente o descendente se puede realizar con respecto a una clave (nombre, dirección, etc.). Sin embargo, puede ser interesante tener clasificado un fichero por categoría laboral y, a su vez, se puede tener por cada categoría laboral, los registros agrupados por nombres o direcciones. Ello nos lleva a la conclusión de que un archivo puede estar ordenado por un campo clave o una jerarquía de campos.

Se dice que un archivo con campos de registro **C1, C2, C3....Cn**, está ordenado principalmente por el campo

**C1**, en orden secundario por el campo **C2**, en orden secundario 2 por el campo **C3**, etc. en orden secundario n por el campo **Cn**. Si el archivo tiene la siguiente organización:

Los registros aparecen en los archivos según el orden de los valores del campo **C1**.

Si se considera un mismo valor **C1**, los registros aparecen en el orden de los valores del campo **C2**.

Para un mismo valor **C (c1)** los registros aparecen según el orden secundario 1 por C2, se necesita:

- Ordenar primero por **C2**.
- Ejecutar a continuación una ordenación por el campo **C1**.

La mayoría de los Sistemas Operativos actuales disponen de programas estándar que realizan la clasificación en uno o varios archivos (sort).

## Clasificación de Archivos

Los archivos están clasificados en orden ascendente o descendente cuando todos sus registros están ordenados en uno de estos sentidos en base al valor de un campo determinado, denominado clave de ordenación.

Si el archivo a ordenar cabe en la memoria central, se carga en un vector y se realiza una clasificación interna, transfiriendo a continuación el Archivo ordenado al soporte externo o copiando el resultado en el archivo original si no se desea conservar.

En el caso de que el archivo sea insuficiente para la memoria central, la clasificación se realizará sobre el archivo almacenado en un soporte externo. El inconveniente de este tipo de clasificación reside en el tiempo, que será mucho mayor debido, especialmente, a las operaciones de entrada/salida de información que requiere la clasificación externa.

Los algoritmos de clasificación son muy variados, pero muchos de ellos se basan en procedimientos mixtos consistentes en aprovechar al máximo la capacidad de la memoria central.

*Como ejemplo de algoritmos de clasificación que no utilizan la memoria central y son aplicables a archivos secuenciales, tenemos la mezcla directa y la mezcla natural.*

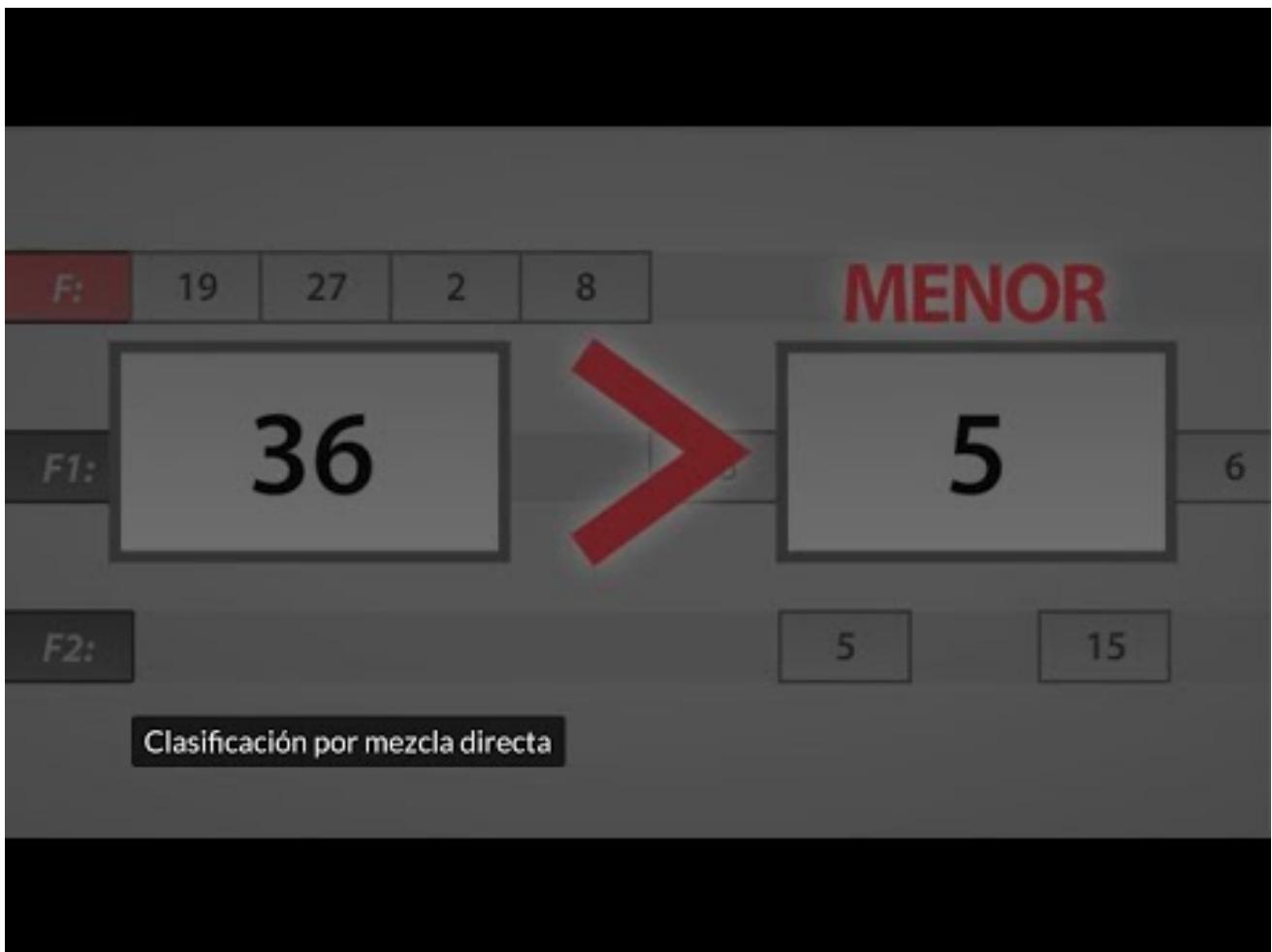
Analicemos los siguientes métodos de ordenación sobre AD:

1. Clasificación por **mezcla directa**
2. Clasificación por **muestra natural**
3. Clasificación por **mezcla de secuencias equilibradas**

### 1. Clasificación por mezcla directa

El método más fácil de comprender es el que se denomina mezcla directa. Veremos su aplicación a través de un breve ejemplo en el que se aplicará el método sobre un vector. Podemos pensar en los componentes del vector como las claves de los registros sucesivos del archivo.

El procedimiento consiste en una partición sucesiva del archivo y una fusión que produce secuencias ordenadas. La primera partición se hace para secuencias de longitud 1 utilizando dos archivos auxiliares y la fusión producirá secuencias ordenadas de longitud 2. A cada nueva partición y fusión se duplicara la longitud de las secuencias ordenadas. El método terminara cuando la longitud de las secuencias ordenadas exceda la longitud del archivo a ordenar.



Video "[Clasificación por mezcla directa](#)" | Elaboración Propia. DEPROE - Colegio Universitario IES

Consideremos el archivo:

El archivo F se divide en dos nuevos archivos F1 y F2

Ahora se funden los archivos F1 y F2 formando pares ordenados

Se vuelve a dividir en partes iguales y en secuencia de longitud 2

La fusión de estos producirá

La nueva partición será

La nueva fusión será

Cada operación que trata por completo el conjunto de datos en su totalidad se denomina una

"fase" y el proceso de ordenación se denomina "pasada".

## 2. Clasificación por muestra natural

Es uno de los mejores métodos de ordenación de archivos secuenciales. Consiste en aprovechar la posible ordenación interna de las secuencias del archivo F, obteniendo con ellas particiones ordenadas de longitud variable sobre una serie de archivos auxiliares, en este caso dos, F1 y F2. A partir de estos ficheros auxiliares escribiremos un nuevo F mezclando los segmentos crecientes máximos de cada uno de ellos.



Video "Clasificación por muestra natural" | Elaboración Propia. DEPROE - Colegio Universitario IES

Clasificar el vector.

Se divide F en dos vectores F1 y F2, donde se ponen alternativamente los elementos F1 y F2. F esta ahora vacío.

Se selecciona el elemento más pequeño de F1 y F2, que pasa a estar en F3.

Ahora se comparan 8 y 19, se selecciona 8. De modo similar 19 y 27

En F1 se ha interrumpido la secuencia creciente y se continúa con F2 hasta que también en él se termine la secuencia creciente.

Ahora 5 y 15 son menores que 36. Finalmente se tendrá:

F1 y F2 están ahora vacíos.

Etapa 2, fase 1: Dividir F3 en dos.

Etapa 2, fase 2:

Etapa 3, fase 1:

Etapa 3, fase 2:

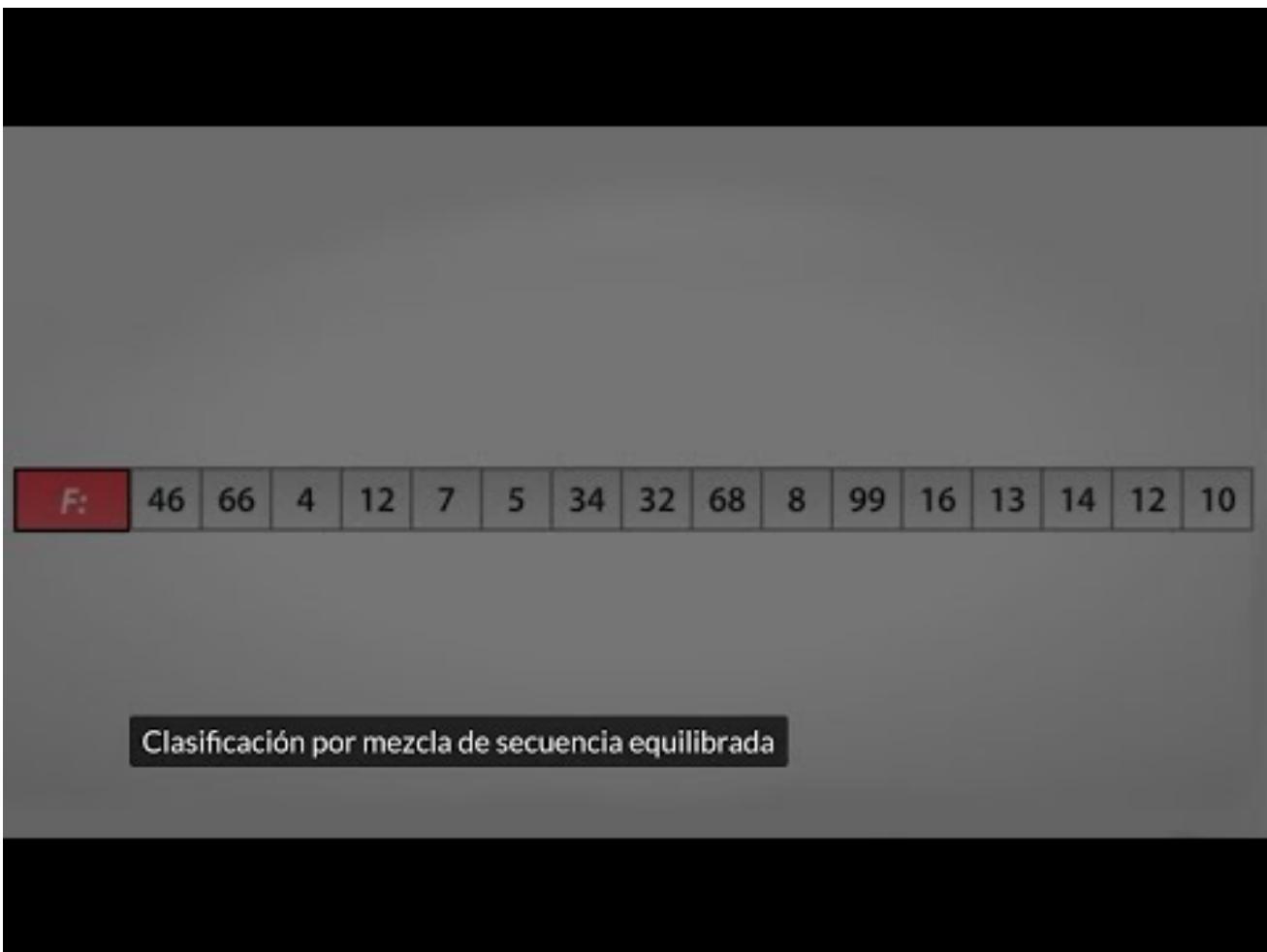
Y el archivo F3 ya está ordenado.

### 3. Clasificación por mezcla de secuencias equilibradas

Este método utiliza la memoria de la computadora para realizar clasificaciones internas y cuatro archivos secuenciales temporales para trabajar.

Supongamos un archivo de entrada F que se desea ordenar por orden creciente de las claves de sus elementos.

Se dispone de cuatro archivos secuenciales de trabajo F1, F2, F3 y F4, y que se pueden colocar m elementos en memoria central en un momento dado en una tabla T de elementos.



Video "Clasificación por mezcla de secuencias equilibradas" | Elaboración Propia. DEPROE - Colegio Universitario IES

El proceso es el siguiente:

1. Lectura de archivos de entrada por bloques de n elementos.
2. Ordenación de cada uno de estos bloques y escritura alternativa sobre F1 y F2.
3. Fusión de F1 y F2 en bloques de 2n elementos que se escriben alternativamente sobre F3 y F4.
4. Fusión de F3 y F4 y escritura alternativa en F1 y F2 de bloques con 4n elementos ordenados.
5. El proceso consiste en doblar cada vez el tamaño de los bloques y utilizando las parejas (F1, F2) y (F3 y F4).

Debido a que en nuestra Situación profesional contamos con un AD de acceso secuencial y el mismo debe ser creado ordenadamente, no utilizaremos los métodos de ordenamiento para la resolución de la misma.



¿Estás listo para un desafío?

**1. Indique la opción correcta**

Los AD de organización secuencial indexada contienen:

- Solo un área de datos.
- 2 áreas un área de índice y una zona de desbordamiento o excedentes.
- 3 áreas, un área de datos, un área de índice y una zona de desbordamiento o excedentes.

**2. Indique la opción correcta**

El método de mezcla normal, utiliza archivos temporales en donde guarda:

- Particiones desordenadas de registros.
- Particiones ordenadas de registros.
- Ninguna de las anteriores.

**3. Indique la opción correcta**

Los archivos se pueden clasificar en orden ascendente o descendente.

- Verdadero
- Falso

**4. Indique la opción correcta**

La clave de la clasificación por mezcla directa es disminuir el número de pasadas e incrementar su tamaño.

- Verdadero
- Falso

**5. Indique la opción correcta**

La ordenación de manera ascendente de un archivo se da únicamente en base a un campo clave.

- Verdadero
- Falso

**6. Indique la opción correcta**

La zona de excedente de un AD de organización secuencial, se utiliza para:

- o La adición de nuevos registros.
- o Como papelera de registros eliminados.
- o Como cola de espera de registros que no se pueden grabar.

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

Los AD de organización secuencial indexada contienen:

- Solo un área de datos.
- 2 áreas un área de índice y una zona de desbordamiento o excedentes.
- X 3 áreas, un área de datos, un área de índice y una zona de desbordamiento o excedentes.

## 2. Indique la opción correcta

El método de mezcla normal, utiliza archivos temporales en donde guarda:

- Particiones desordenadas de registros.
- X Particiones ordenadas de registros.
- Ninguna de las anteriores.

## 3. Indique la opción correcta

Los archivos se pueden clasificar en orden ascendente o descendente.

- X Verdadero
- o Falso

## 4. Indique la opción correcta

La clave de la clasificación por mezcla directa es disminuir el número de pasadas e incrementar su tamaño.

- X Verdadero
- o Falso

## 5. Indique la opción correcta

La ordenación de manera ascendente de un archivo se da únicamente en base a un campo clave.

- o Verdadero
- X Falso

## 6. Indique la opción correcta

La zona de excedente de un AD de organización secuencial, se utiliza para:

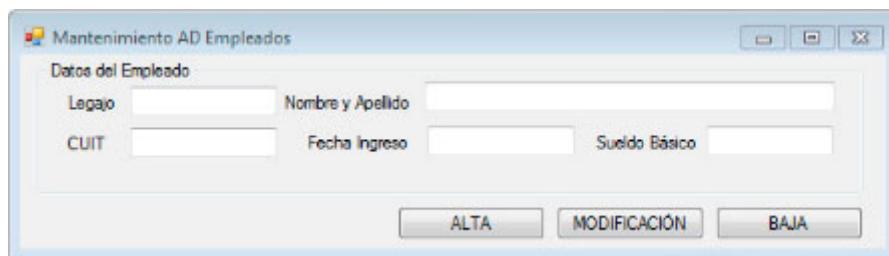
- X La adición de nuevos registros.
- o Como papelera de registros eliminados.
- o Como cola de espera de registros que no se pueden grabar.

## SP4 / Ejercicio resuelto

En nuestra Situación profesional se nos solicitó el desarrollo de los procedimientos necesarios para el mantenimiento del AD secuencial Empleados, en donde se almacena los datos de todos los empleados de la empresa.

Para resolver este pedido definiremos una estructura de registro empleado con los datos: *Legajo, Apellido y Nombre, CUIT, Fecha de ingreso y sueldo básico*. Este será utilizado para el manejo de los datos de cualquier empleado.

Veamos a continuación el desarrollo del procedimiento correspondiente a cada uno de los botones de comandos de la interfaz que representa una operación de actualización sobre el AD.



```

Programa MantenimientoAD
  # Definición de la estructura Registro Empleado
  Registro Empleado
    Variable numérica legajo
    Variable alfanumérica nombre
    Variable alfanumérica CUIT
    Variable numérica ingreso
    Variable numérica sueldo
  Fin registro
  Formulario frmMantenimientoAD
    Marco mrcEmpleado
      Caja de Texto txtLegajo
      Caja de Texto txtNombre
      Caja de Texto txtCUIT
      Caja de Texto txtIngreso
      Caja de Texto txtSueldo
    Fin Marco
    Botón de Comando cmdAlta
    Botón de Comando cmdModificar
    Botón de Comando cmdBaja
  Fin Formulario
  # Definición del Agregado de Datos
  Agregado de Datos Empleados Tipo Registro Empleado
  Procedimiento cmdAlta:Click
    Registro NvoEmpleado Tipo Registro Empleado
    # Cargo los valores de la interfaz
    NvoEmpleado.legajo = txtLegajo
    NvoEmpleado.nombre = txtNombre
    NvoEmpleado.CUIT = txtCUIT
    NvoEmpleado.ingreso = txtIngreso
    NvoEmpleado.sueldo = txtSueldo
    # Apertura y lectura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados
    # Recorremos el AD hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF )
      Leer AD Empleados
    Fin mientras
    # Grabamos el nuevo registro luego del último registro del AD
    Escribir AD NvoEmpleado
    # Cerramos el AD Empleados
    Cerrar AD Empleados
  Fin procedimiento

```

```

Procedimiento cmdModificar:Click
    # Apertura y lectura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados

    # Recorremos el AD hasta tanto encontremos el registro que
    # deseamos modificar o hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
        Leer AD Empleados
    Fin mientras

    # Corroboramos que realmente encontramos el registro a modificar
    Si txtLegajo = Empleados.Legajo
        # Modificamos el registro
        Empleados.Legajo = txtLegajo
        Empleados.Nombre = txtNombre
        Empleados.CUIT = txtCUIT
        Empleados.Ingreso = txtIngreso
        Empleados.Sueldo = txtSueldo
        Actualizar Registro AD Empleados
    Sino
        Imprimir "El empleado que desea modificar no existe"
    Fin Si

    # Cerramos el AD Clientes
    Cerrar AD Empleados

Fin procedimiento

Procedimiento cmdBaja:Click
    # Apertura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados

    # Recorremos el AD hasta tanto encontremos el registro que
    # deseamos dar de baja o hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
        Leer AD Empleados
    Fin mientras

    # Corroboramos que realmente encontramos el registro a eliminar
    Si txtLegajo = Empleados.Legajo
        # Eliminamos el registro
        Eliminar Registro AD Empleados
    Sino
        Imprimir "El empleado que desea eliminar no existe"
    Fin Si

    # Cerramos el AD Clientes
    Cerrar AD Empleados

Fin procedimiento

Fin programa

```

- Alta

```

Procedimiento cmdAlta:Click
  Registro NvoEmpleado Tipo Registro Empleado

    # Cargo los valores de la interfaz
    NvoEmpleado.legajo = txtLegajo
    NvoEmpleado.nombre = txtNombre
    NvoEmpleado.CUIT = txtCUIT
    NvoEmpleado.ingreso = txtIngreso
    NvoEmpleado.sueldo = txtSueldo

    # Apertura y lectura del Agregado de Datos
    Abrir AD Empleados
    Leer AD Empleados

    # Recorremos el AD hasta encontrar el final de archivo
    Mientras ( NOT Empleados.EOF )
      Leer AD Empleados
    Fin mientras

    # Grabamos el nuevo registro luego del último registro del AD
    Escribir AD NvoEmpleado

    # Cerramos el AD Empleados
    Cerrar AD Empleados

Fin procedimiento

```

- Modificación

**Procedimiento cmdModificar:Click**

```
# Apertura y lectura del Agregado de Datos
Abrir AD Empleados
Leer AD Empleados

# Recorremos el AD hasta tanto encontremos el registro que
# deseamos modificar o hasta encontrar el final de archivo
Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
    Leer AD Empleados
Fin mientras

# Corroboramos que realmente encontramos el registro a modificar
Si txtLegajo = Empleados. Legajo
    # Modificamos el registro
    Empleados.Legajo = txtLegajo
    Empleados.Nombre = txtNombre
    Empleados.CUIT = txtCUIT
    Empleados.Ingreso = txtIngreso
    Empleados.Sueldo = txtSueldo
    Actualizar Registro AD Empleados
Sino
    Imprimir "El empleado que desea modificar no existe"
Fin Si

# Cerramos el AD Clientes
Cerrar AD Empleados

Fin procedimiento
```

- Baja

**Procedimiento cmdBaja:Click**

```
# Apertura del Agregado de Datos
Abrir AD Empleados
Leer AD Empleados

# Recorremos el AD hasta tanto encontremos el registro que
# deseamos dar de baja o hasta encontrar el final de archivo
Mientras ( NOT Empleados.EOF AND txtLegajo <> Empleados.Legajo)
    Leer AD Empleados
    Fin mientras

# Corroboramos que realmente encontramos el registro a eliminar
Si txtLegajo = Empleados.Legajo
    # Eliminamos el registro
    Eliminar Registro AD Empleados
Sino
    Imprimir "El empleado que desea eliminar no existe"
Fin Si

# Cerramos el AD Clientes
Cerrar AD Empleados

Fin procedimiento
```

Con este programa, hemos cumplido con el requerimiento que se nos encargó. A partir de ahora la empresa podrá dar de alta nuevos empleados, dar de baja los empleados que se van de la empresa y modificar los que se encuentran en el AD Empleados.

## SP4 / Ejercicio por resolver

Una librería almacena en un AD secuencial la siguiente información de sus libros: código, título, autor y precio.

El AD está ordenado ascendentemente por los códigos de los libros.

Se solicita que realice los procedimientos de alta, baja y consulta, y un procedimiento de ordenamiento que organice el AD.

## SP4 / Evaluación de paso



¡Vamos a comprobar cuánto aprendiste!

**1. Indique la opción correcta**

El alta de un nuevo registro en un AD se realiza agregando el mismo en el primer lugar del AD.

- Verdadero
- Falso

**2. Indique la opción correcta**

Una colisión se presenta cuando a dos registros de claves distintas, producen la misma dirección física.

- Verdadero
- Falso

**3. Indique la opción correcta**

Los AD de organización secuencial indexada contienen 3 áreas: una área de inicio, un área media y una zona de desbordamiento.

- Verdadero
- Falso

**4. Indique la opción correcta**

La clasificación por mezcla directa no permite la partición ni la fusión de un archivo.

- Verdadero
- Falso

**5. Indique la opción correcta**

Se puede realizar un mantenimiento adecuado de un AD mediante las operaciones de Alta, Baja y Modificación de registros.

- Verdadero
- Falso

# Respuestas de la Autoevaluación

## 1. Indique la opción correcta

El alta de un nuevo registro en un AD se realiza agregando el mismo en el primer lugar del AD.

Verdadero

Falso

## 2. Indique la opción correcta

Una colisión se presenta cuando a dos registros de claves distintas, producen la misma dirección física.

Verdadero

Falso

## 3. Indique la opción correcta

Los AD de organización secuencial indexada contienen 3 áreas: una área de inicio, un área media y una zona de desbordamiento.

Verdadero

Falso

## 4. Indique la opción correcta

La clasificación por mezcla directa no permite la partición ni la fusión de un archivo.

Verdadero

Falso

## 5. Indique la opción correcta

Se puede realizar un mantenimiento adecuado de un AD mediante las operaciones de Alta, Baja y Modificación de registros.

Verdadero

Falso

## Cierre

Nuestro objetivo está cumplido. Ha dado un paso más hacia su gran meta de ser un Profesional.

Con los conocimientos adquiridos está en condiciones de enfrentarse a cualquier lenguaje de programación, teniendo solo que adquirir conocimiento extra de las sentencias propias de cada lenguaje. Pero lo más importante será que siempre estará con usted el poder de pensar con lógica informática.

Es hora de seguir caminando y seguir descubriendo las diversas herramientas que el mundo de la informática le da día a día y que, a diferencia de otras especialidades, en la informática siempre hay algo nuevo y maravilloso que descubrir y aprender.

Es un placer haberle servido como guía y es mi deseo que tenga éxito en el resto de esta hermosa y apasionante carrera en la que hemos coincidido.

**El autor**

## Bibliografía

- LÓPEZ ROMAN. "Programación Estructurada y Orientada a Objetos", Editorial Alfaomega Grupo Editor, 2011.
- FARRELL JOYCE. "Introducción a la Programación: Lógica y Diseño", Editorial Cengage Learning Argentina, 2009.
- JOYANES AGUILAR, L. "Fundamentos de Programación". S.A. McGraw-Hill / Interamericana de España, 2008
- BONGIOVANNI ÉRICA, FARNETI ALEJANDRA, OBREGÓN DIEGO. "Programación Lógica". Editorial IES Siglo 21 2007