FACULTAD DE TECNOLOGÍA INFORMÁTICA

UNIVERSIDAD ABIERTA
INTERAMERICANA

Programación Estructurada

Ing. Rafael Antonio Brizuela

Dania Calderón 1º Año

Universidad Abierta Interamericana

Facultad de Tecnología Informática

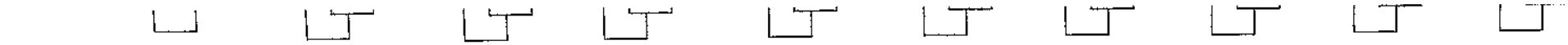
Carrera: Ingeniería en Sistemas Informáticos

Programación Estructurada

Ing. Rafael Antonio Brizuela
(Profesor Titular)

Autorización: T1 0001/2003
Séptima Versión - Año 2010

ÍNDICE



• *Fases en la resolución de problemas*

- **Las fases**
- **Análisis del problema**
- **Diseño del algoritmo**
- **Herramientas de programación**
- **Compilación y ejecución de un programa**
- **Verificación y ejecución de un programa**
- **Documentación y mantenimiento**
- **Tipos de representación de la programación**
- **Ciclo de vida del software**

El proceso de resolución de un problema con una computadora conduce a la escritura de un programa y a su ejecución en la misma. Aunque el proceso de diseñar programas es "esencialmente", un proceso creativo, se puede considerar una serie de fases o pasos comunes, que generalmente deben seguir todos los programadores.

Las fases de resolución de un problema con computadora son:

- ***Análisis del problema:*** El problema se analiza teniendo presente la especificación de los requisitos dados por el cliente de la empresa o por otra persona que encarga el programa.

- ***Diseño del algoritmo:*** una vez analizado el problema, se diseña una solución que conducirá a un *algoritmo* que resuelva el problema.

- ***Codificación (implementación):*** la solución se escribe en la sintaxis del lenguaje de alto nivel (por ejemplo, C) y se obtiene un programa.

- ***Ejecución, verificación y depuración:*** el programa se ejecuta, se comprueba rigurosamente y se elimina todos los errores (denominados "*bugs*", en inglés) que puedan aparecer.

- ***Mantenimiento:*** El programa se actualiza y modifica, cada vez que sea necesario, de modo que se cumplan todas las necesidades de cambio de sus usuarios.

- ***Documentación:*** Escritura de las diferentes fases del ciclo de vida del software, esencialmente el análisis, diseño y codificación, unidos a manuales de usuario y referencia, así como normas para el mantenimiento.

Las dos primeras fases conducen a un diseño detallado escrito en forma de algoritmo. Durante la tercera etapa (*codificación*) se implementa el algoritmo en un código escrito en un lenguaje de programación, reflejando las ideas desarrolladas en las fases de análisis y diseño.

La fase de *ejecución* y *compilación* traduce y ejecuta el programa. En las fases de *verificación* y *depuración* el programador busca errores de las etapas anteriores y los elimina. Comprobará que mientras más tiempo se gaste en la fase de análisis y diseño, menos se gastara en la depuración del programa. Por último, se debe realizar la *documentación del programa.*

Un ***algoritmo*** es un método para resolver un problema mediante una serie de pasos precisos, definidos y finitos.

- Las Fases

Un algoritmo debe producir un resultado en un tiempo finito. Los métodos que utilizan algoritmos se denominan *métodos algorítmicos*, en oposición a los métodos que implican algún juicio o interpretación que se denominan *métodos heurísticos*.

Los métodos algorítmicos se pueden *implementar* en computadoras; sin embargo, los procesos heurísticos no han sido convertidos fácilmente en las computadoras. En los últimos años las técnicas de inteligencia artificial han hecho posible la implementación del proceso heurístico en computadoras.

Ejemplos de algoritmos son: instrucciones para montar en una bicicleta, hacer una receta de cocina, obtener el máximo divisor de dos números, etc.

Los algoritmos se pueden expresar por *formulas*, *diagramas de flujo* y *pseudocódigos*. Esta última representación es la más utilizada en lenguajes estructurados como C.

Características de un algoritmo

- *preciso* (indicar el orden de realización en cada paso),
- *definido* (sí se sigue dos veces, obtiene el mismo resultado cada vez),
- *finito* (tiene fin; un número determinado de pasos).

- Análisis del problema

La primera fase de la resolución de un problema es el *análisis del problema*. Esta fase requiere una clara definición, donde se contemple exactamente lo que debe hacer el programa y el resultado o solución deseada.

Para poder definir bien un problema es conveniente responder a las siguientes preguntas:

- ¿Qué entradas se requieren? (tipo y cantidad)
- ¿Cuál es la salida? (tipo y cantidad)
- ¿Qué método produce la salida deseada?

- Diseño del algoritmo

En la etapa de análisis del proceso de programación se determina *qué* hace el programa. En la etapa de diseño se determina *cómo* hace el programa la tarea solicitada.

Los métodos más eficaces para el proceso de diseño se basan en el conocido por *divide y vencerás*. Es decir, la resolución de un problema complejo se realiza dividiendo el problema

en subproblemas y a continuación dividir estos subproblemas en otros de nivel más bajo, hasta que pueda ser *implementada* una solución en la computadora.

Este método se conoce técnicamente como **diseño descendente** (top-down) o **modular.**

El proceso de romper el problema en cada etapa y expresar cada paso en forma más detallada se denomina *refinamiento sucesivo.*

Cada subprograma es resuelto mediante un **módulo** (*subprograma*) que tiene un solo punto de entrada y un solo punto de salida.

Cualquier programa bien diseñado consta de un *programa principal* (el módulo de nivel más alto) que llama a subprograma (módulo de nivel bajo) que a su vez pueden llamar a otros subprogramas. Los programas estructurados de esta forma se dice que tienen un *diseño modular* y el método de romper el programa en módulos más pequeños se llama *programación modular.*

Los módulos pueden ser planeados, codificados, comprobados y depurados independientemente (incluso por diferente programadores) y a continuación combinarlos entre sí. El proceso implica la ejecución de los siguientes pasos hasta que el programa se termina:

1. **Programar un módulo.**
2. **Comprobar el módulo.**
3. **Si es necesario, depurar el módulo.**
4. **Combinar el módulo con los anteriores.**

El proceso que convierte los resultados del análisis del problema en un diseño modular con refinamiento sucesivo que permitan una posterior traducción a un lenguaje se denomina **diseño del algoritmo.**

El diseño del algoritmo es independiente al lenguaje de programación en el que se vaya a codificar posteriormente.

- Herramientas de programación.

Las dos herramientas más utilizadas comúnmente para diseñar algoritmos son: *diagramas de flujo y pseudocódigos*

Diagramas de flujo

Un **diagrama de flujo** es una representación gráfica de un algoritmo.

Pseudocódigo

El **pseudocódigo** es una herramienta de programación en la que las instrucciones se escriben en palabras similares al inglés o español, que facilitan tanto la escritura como la lectura de programas. En esencia, el pseudocódigo se puede definir como *un lenguaje de especificaciones de algoritmos.*

Codificación de un programa

Codificación es la escritura en un lenguaje de programación de la representación del algoritmo desarrollado en las etapas precedentes. Dado que el diseño de un algoritmo es independiente del lenguaje de programación utilizado para su implementación, el código puede ser escrito con igual facilidad en un lenguaje o en otro.

Compilación y ejecución de un programa

Una vez que el algoritmo se ha convertido en un programa fuente, es preciso introducirlo en memoria mediante el teclado y almacenarlo posteriormente en un disco. Esta operación se realiza con un programa editor, posteriormente el programa fuente se convierte en un *archivo de programa* que se guarda en disco.

El **programa fuente** debe ser traducido a lenguaje máquina, este proceso se realiza con el compilador y el sistema operativo que se encarga prácticamente de la compilación.

Si tras la compilación se presentan errores (*errores de compilación*) en el programa fuente, es preciso volver a editar el programa, corregir los errores y compilar de nuevo. Este proceso se repite hasta que no se producen errores, obteniéndose el **programa objeto** que todavía no es ejecutable directamente. Suponiendo que no existen errores en el programa fuente, se debe instruir al sistema operativo para que realice la fase de **montaje** o **enlace** (*link*), carga, del programa objeto con las librerías del programa el compilador. El proceso de montaje produce un **programa ejecutable.**

Cuando el programa ejecutable se ha creado, se puede ya ejecutar (correr o rodar) desde el sistema operativo con sólo teclear su nombre (en el caso de DOS). Suponiendo que no existen errores durante la ejecución (llamados **errores en tiempo de ejecución**), se obtendrá la salida de resultados del programa.

- Verificación y ejecución de un programa

La verificación o compilación de un programa es el proceso de ejecución del programa con una amplia variedad de datos de entrada, llamados datos de test o prueba, que determinaran si el programa tiene errores (bugs). Para realizar la verificación se debe desarrollar una amplia gama de datos de test: Valores normales de entrada, valores extremos de entrada que comprueben los limites del programa y valores de entrada que comprueben aspectos especiales del programa.

La *depuración* es el proceso de encontrar los errores del programa y corregir o eliminar dichos errores.

Cuando se ejecuta un programa se puede producir tres tipos de errores:

1. *Errores de compilación.* Se producen normalmente por uso incorrecto del lenguaje de programación y suelen ser *errores de sintaxis.* Si existe un error de sintaxis la computadora no puede comprender la instrucción, no se obtendrá el programa objeto y el compilador imprimirá una lista de todos los errores encontrados durante la compilación.

2. *Errores de ejecución.* Estos errores se producen por instrucciones que la computadora puede comprender pero no ejecutar. Ejemplos típicos son: división por ceros y raíces cuadradas de números negativos. En estos casos se detiene la ejecución del programa y se imprime un mensaje de error.

3. *Errores lógicos.* Se producen en la lógica del programa y la fuente de error suele ser el diseño del algoritmo. Estos errores son los más difíciles de detectar, ya que el programa puede funcionar y no producir errores de compilación ni de ejecución, y solo puede advertir el error por la obtención de resultados incorrectos. En este caso se debe volver a la fase de diseño del algoritmo, modificar el algoritmo, cambiar el programa fuente y compilar y ejecutar una vez más.

- Documentación y mantenimiento

La documentación consta de la descripción de los pasos a dar en el proceso de resolución de un problema. La importancia de la documentación debe ser destacada por su decisiva influencia en el producto final. Programas pobremente documentos son difíciles de leer, más difíciles de depurar y casi imposible de mantener y modificar.

La documentación de un programa puede ser *interna y externa. La documentación interna* es la contenida en líneas de comentarios. La documentación externa incluye análisis, diagrama de flujo y/o pseudocodigos, manuales de usuarios con instrucciones para ejecutar el programa y para interpretar los resultados.

La documentación es vital cuando se desea corregir posibles errores futuros o bien cambiar el programa. Tales cambios se denominan *mantenimiento del programa.* Después de cada cambio, la documentación debe ser actualizada para facilitar cambios posteriormente.

Tipos de representación de la programación

Programación modular

La programación modular es uno de los métodos de diseño más flexible y potente para mejorar la productividad de un programa. En programación modular el programa se divide en módulos (partes independientes), cada una de las cuales ejecuta una única actividad o tarea y

se codifica independientemente de otros módulos. Cada modulo se analiza, codifica y ponen a punto por separado.

Cada programa contiene un modulo denominado programa principal que controla todo lo que sucede; se transfiere el control a submódulos (que se denominaran subprogramas).

Cada submódulo devuelve el control a submódulos, que se denominaran subprogramas, de modo que puedan ejecutar sus funciones. Cada submódulo devuelve el control al módulo principal cuando se haya completado su tarea. Si la tarea asignada a cada submódulo es demasiado compleja, este deberá romperse en otros módulos más pequeños. El proceso sucesivo de subdivisión en módulos continua hasta que cada módulo tenga solamente una tarea especifica que ejecutar. Esta tarea puede ser *entrada, salida, manipulación de datos, control de otros módulos o alguna combinación de estos.* Un módulo puede transferir temporalmente (bifurcar) el control a otro módulo; sin embargo, cada módulo debe eventualmente devolver el control al módulo del cual se recibe originalmente el control.

Los módulos son independientes en el sentido en que ningún módulo puede tener acceso directo a cualquier otro módulo excepto el módulo al que llama y sus propios submódulos.

Dado que son independientes, diferentes programadores pueden trabajar simultáneamente en diferentes partes del mismo programa. Reducirá el tiempo del diseño del algoritmo y posterior codificación del programa.

La descomposición de un programa en módulos se conoce también como el método de divide y vencerás (divide and conquer). Se diseña un módulo con independencia de los demás, y siguiendo un método ascendente o descendente se llagará hasta la descomposición final del problema en módulos en forma jerárquica.

Un módulo se puede modificar radicalmente sin afectar a otros módulos, incluso sin alterar su función principal.

Programación estructurada

La programación estructurada significa escribir un programa de acuerdo a las siguientes reglas:

- El programa tiene un diseño modular.
- Los módulos son diseñados de modo descendente.
- Cada modulo se codifica utilizando las tres estructuras de control básicas: secuencia, selección y repetición.

Programación estructurada significa también programación sin GOTO (C no requiere el uso de la sentencia GOTO)

Esta utiliza un numero limitado de estructuras de control que minimiza la complejidad de los programas y, por consiguiente, reducen los errores; hace los programas más fáciles de escribir, verificar, leer y mantener. Los programas deben estar dotados de una estructura.

La programación estructurada es el conjunto de técnicas que incorporan:

- Recursos abstractos,
- Diseño descendente (top-down),
- Estructuras básicas.

Recursos abstractos

La programación estructurada se auxilia de los recursos abstractos en lugar de los recursos concretos de que dispone un determinado lenguaje de programación.

Diseño descendente

Conocido también como top-down es el proceso por el cual un problema se descompone en una serie de niveles o pasos sucesivos de refinamiento.

Es decir se descomponen en etapas jerárquicas relacionadas por medio de las entradas y salidas de información.

A cada estructura se la puede considerar desde dos puntos de vista, que hace y como lo hace.

Estructuras de control

Son métodos para especificar el orden en que las instrucciones de un algoritmo se ejecutaran. Dicho orden determina el flujo de control.

La programación estructurada hace mas fácil el escribir, verificar, leer y mantener un programa, utilizando un número limitado de estructuras que minimizan la complejidad de los programas.

- **Teorema de la programación estructurada**

➢ Toda estructura posee un solo punto de entrada y uno solo de salida de la misma.
➢ Existen varios caminos desde la entrada hasta la salida.
➢ Todas las instrucciones pueden ser ejecutables en algún momento y no existen bucles infinitos o sin fin.

Representación grafica de los algoritmos

Existen muchísimos métodos y formas dependiendo del centro de cómputos donde se este trabajando, Chaplin, Jackson, diagrama de flujo estructurado, de UTN,etc.

Ciclo de vida del software

El ciclo de vida de un sistema comienza cuando se detecta su necesidad y finaliza cuando se volvió obsoleto, ya sea por el tiempo, cambio de negocio u cualquier otra causa que nos indique que ya no es necesario.
Tenemos dentro de él varias etapas

1. Análisis
2. Diseño
3. Implementación
4. Depuración
5. Mantenimiento

- Análisis

Determina la definición del problema y especifica los requisitos que ha de tener por necesidad del usuario, quien deberá tener una participación activa en esta etapa a fin de modificar o corregir cualquier mala interpretación realizada por el analista.

- Diseño

En esta etapa se definirá como el sistema lo hará para lograr las especificaciones solicitadas.

- Implementación

Es la etapa en la cual se codifica a un lenguaje de programación los diseños efectuados anteriormente.

Es conveniente para un mas fácil mantenimiento y lectura del programa la utilización de sangrías y comentarios de los distinto procedimientos que se codifican.

- Depuración

En esta etapa se corrigen los distintos errores de codificación, se realizan pruebas y se integra, siendo la fase de prueba algo muy difícil pues a lo sumo con los lotes de prueba que nosotros creamos, detectaremos que el programa no cometa un error, o sea que detectamos la presencia de un error y no su ausencia.

Los lotes de prueba deben ser tan amplios, que alberguen dentro de sus valores, todas las posibles combinaciones a fin de detectar errores.

Existen varios métodos, uno de ellos es la corrida en paralelo, del sistema para ver si da diferencia entre el nuevo y el anterior.

Recordemos que un sistema no tiene porque haber existido anteriormente en modo computacional.

- Mantenimiento

Es la tarea mas ardua y quizás mas costosa, debemos hacerlo cuando se modifica el entorno de hardware o si cambian las necesidades del usuario, por ejemplo.

Factores en la calidad del software

- Eficiencia
- Transportabilidad
- Verificabilidad
- Integridad
- Fácil de usar
- Robustez
- Extensibilidad
- Reutilización
- Compatibilidad

•El lenguaje C

- Estructura general de un programa en C
- Tipos de datos en C
- Operadores
- Tipos de estructuras
- Estructura secuencial
- Estructuras de selección
- Estructuras de iteración
- Funciones
- Pasaje de parámetros a una función
- Paso de parámetros por valor
- Paso de parámetros por referencia
- Clases de mantenimiento
- Concepto y uso de funciones de biblioteca
- Arrays
- Arrays unidimensionales (vectores)
- Arrays bidimensionales (matriz)

- Estructura general de un programa en C

Un programa en C se compone de una o más funciones. Una de las funciones debe ser obligatoriamente main. Una función en C es un grupo de instrucciones que realizan una o más acciones. Asimismo, un programa contendrá una serie de directivas #include que permitirán incluir en el mismo archivos de cabecera que a su vez constaran de funciones y datos predefinidos en ellos.

```
#include <stdio.h>          <-------  archivo de cabecera stdio.h
int main ()                 <-------  cabecera de función
{                           <-------  nombre de la función
    ...                     <-------  sentencias
}
```

#include	Directivas del preprocesador
#define	Macros del procesador

Declaraciones globales

- **Prototipos de funciones**
- **Variables**

Función principal main

```
main()
{
  declaraciones locales
  sentencias
}
```

Definiciones de otras funciones

```
tipo1 func1(...)
{
 ...
}
```

De un modo mas explicito un programa en C puede incluir:

- Directivas de preprocesador;
- Declaraciones globales;
- La función main();
- Funciones definidas por el usuario;
- Comentarios del programa (utilizados en su totalidad)

ADVERTENCIA

El programa mas corto de C es el «programa vacío»que no hace nada.

- Directivas del preprocesador

El procesador en un programa C se puede considerar como un editor de texto inteligente que consta de directivas (instrucciones al compilador antes de que se compile el programa principal). Las dos directivas más usuales son #include y #define.

Todas las directivas del preprocesador comienzan con el signo de libro o *«almohadilla»* (#), que indica al compilador que lea las directivas antes de compilar la parte (función) principal del programa.

Las *directivas* son instrucciones al compilador. Las directivas no son generalmente sentencias -obsérvese que su línea no termina en punto y coma-, sino instrucciones que se dan al compilador antes de que el programa se compile. Aunque las directivas pueden definir macros, nombres de constantes, archivos fuente adicionales, etc., su uso más frecuente en C es la inclusión de archivos de cabecera.

Existen archivos de cabecera estándar que se utilizan ampliamente, tales como STDIO.H, STDLIB.H, MATH.H, STRING.H y se utilizaran otros archivos de cabecera definidos por el usuario para diseño estructurado.

La mayoría de los programadores C sitúan las directivas del procesador al principio del programa, aunque esta posición no es obligatoria.

- Declaraciones globales

Las declaraciones globales indican al compilador que las funciones definidas por el usuario o variables así declaradas son comunes a todas las funciones de su programa. Estas, se sitúan antes de la función main(). Si se declara global una variable Grado_clase del tipo

```
Int Grado_clase;
```

Cualquier función de su programa, incluyendo main(), puede acceder a la variable Grado_clase.

La zona de declaraciones globales de un programa puede incluir declaraciones de variables además de declaraciones de función. Las declaraciones de función se denominan prototipos

Int media (int a, int b)

Función main()

Cada programa C tiene una función main() que es el punto de entrada al programa. Su estructura es:

```
main()
{
  ...          <-------------------  bloque de sentencia
}
```

Las sentencias incluidas entre la llaves {... } se denominan bloque. Un programa debe tener solo una función main(). Si se intenta hacer dos funciones main() se produce un error.

Además de la función main(), un programa C consta de una colección de funciones.

Una función es un subprograma que devuelve un único valor, un conjunto de valores o realiza alguna tarea especifica tal como E/S.

Las variable y constantes globales se declaran y definen fuera de la definición de las funciones, generalmente en la cabecera del programa, antes de main(), mientras que las variables y constantes locales se declaran y definen en la cabecera del cuerpo o bloque de la función principal, o en la cabecera de cualquier bloque. Las sentencias situadas en el interior del cuerpo de la función main(), o cualquier otra función, deben terminar en punto y coma.

- Funciones definidas por el usuario

Un programa C es una colección de funciones. Todos los programas se construyen a partir de una o más funciones que se integran para crear una aplicación. Todas las funciones contienen una o más sentencias C y se crean generalmente para realizar una única tarea, como imprimir la pantalla, escribir un archivo cambiar el color de la pantalla. Se puede declarar y ejecutar un numero de funciones casi ilimitado en un programa C.

Todas las funciones tienen nombre y una lista de valores que reciben. Se puede asignar cualquier nombre a su función, pero normalmente se procura que dicho nombre describa el propósito de la función. En C las funciones requieren una declaración o prototipo en el programa:

void trazarcurva();

Comentarios

Un comentario es cualquier información que se añade a su archivo fuente para proporcionar documentación de cualquier tipo. El compilador ignora los comentarios, no realiza ninguna tarea concreta. El uso de comentarios es totalmente opcional, aunque dicho uso es muy recomendable.

Se considera buena practica de programación comentar su archivo fuente tanto como sea posible, al objeto de que usted mismo y otros programadores puedan leer fácilmente el programa con el paso del tiempo. Es buena practica de programación comentar su programa en la parte superior de cada archivo fuente. La información que se suele incluir es el nombre del programador, una breve descripción, la fecha en que se creo la versión y la información de la revisión.

Los comentarios en C estándar comienzan con la secuencia */.

Todo el texto situado entre las dos secuencias es un comentario ignorado por el compilador.

```
/* PRUEBA1.C —Primer programa C */
```

si se necesitan varias líneas de programa se puede hacer lo siguiente:

```
/*
  Programa          : PRUEBA1.C
  Programador       : Pepe Mortimer
  Descripción       : Primer programa C
  Fecha creación    : septiembre 2002
  Revisión          : Ninguna
*/
```

También se puede situar comentarios de la siguiente forma:

```
scanf ("%&", &X);      /* sentencia de entradas de un valor entero */
```

Los archivos de cabecera en C tienen normalmente una extensión.*h* y los archivos fuente, la extensión.*c*.

- Tipos de datos en C

C no soporta un gran numero de tipos de datos predefinidos, pero tiene la capacidad para crear sus propios tipos de datos. Todos los tipos de datos simples o básicos de C son, esencialmente, números. Los tres tipos de datos básicos son:

- enteros;
- números de comas (reales);
- caracteres.

Tipo	Ejemplo	Tamaño en bytes	Rango Mínimo..Máximo
char	'C'	1	0...2
short	-15	2	-128...127
int	1024	2	-32768...32767
unsigned int	42325	2	0...65535
long	262144	4	-2147483648...2147483637
float	10.5	4	$3.4*(10^{-38})...3.4*(10^{308})$
double	0.00045	8	$1.7*(10^{-308})...1.7*(10^{308})$
long double	1e-8	8	*igual que* double

Los tipos de datos fundamentales en C son:

- **enteros:** (números completos y sus negativos), de tipo int.
- **Variantes de enteros:** tipos short, long, int y unsigned.
- **reales:** números decimales, tipos float, double o long double.
- **caracteres:** Letras, dígitos, símbolos y signos de puntuación, tipo char.

Char, int, float y double son palabras reservadas, o más especificas, especificadores de tipos. Cada tipo de dato tiene su propia lista de atributos que definen las características del tipo y pueden variar de una maquina a otra. Los tipos char, int y double tienen variaciones o modificadores de tipos de datos, tales como short, long, signed y unsigned, para permitir un uso más eficiente de los tipos de datos.
Existe el tipo adicional enum (constante de enumeración)

Enteros (int)

Probablemente el tipo de dato más familiar es el entero, o tipo int. Los enteros son adecuados para aplicaciones que trabajen con datos numéricos. Los tipos enteros se almacenan internamente en 2 bytes (o 16 bits) de memoria. La tabla siguiente resume los tres tipos de enteros básicos, junto con el rango de valores y el tamaño en bytes usual, dependiente de cada maquina.

Tipo C	Rango de valores	Uso recomendado
int	-32.768.. +32.767	Aritmética de enteros, bucles for, conteo.
unsigned	0.. 65.535	Conteo, bucles for, indices.
short int	-128.. +127	Aritmética de enteros, bucles for, conteo.

- Declaración de variables

La forma más simple de una variable en C es poner primero el tipo de dato y a continuación el nombre de la variable, éste se pone a continuación.

El formato de la declaración es:

<tipo de dato> <nombre de variable> = <valor inicial>

se pueden declarar también múltiples variables en la misma línea:

<tipo_de_dato> <nom_var1>, <nom_var2>... <nom-varn>

Así por ejemplo:

```
Int longitud; int. valor = 99;
Int valor1, valor2;
Int num_parte = 1141, num_ítems = 45;
```

Si se desea forzar al compilador para tratar sus constantes como long, añada la letra L (o bien 1) a su constante. Por ejemplo,

```
Long numeros_grandes = 40000L;
```

Tipo C	Rango de valores
Long	-2147483648.. 2147483647
Insigne long	0.. +4294967295

- Tipos de coma flotante (float / double)

Los tipos de datos de coma (punto) flotante representan número reales que contienen una coma (un punto) decimal, tal como 3.14159, o números muy grandes, tales como $1.85*10^{15}$. La declaración de las variables de coma flotante es igual que la de variables enteras. Así, un ejemplo es el siguiente:

```
Float valor;          /* declara una variable real */
Float valor1, valor2; /* declara varias variables de coma flotante */
Float valor = 99.99;  /* asigna el valor 99.99 a la variable valor */
```

C soporta tres formatos de coma flotante como muestra el cuadro siguiente. El tipo float requiere 4 bytes de memoria, double requiere 8 bytes y long double requiere 10 bytes (Borland C)

Tipo C	Rango de valores	Precisión	BYTES
Float	3.4×10^{-38}.... 3.4×10^{38}	7 dígitos	4
Double	1.7×10^{-308}.... 1.7×10^{308}	15 dígitos	8
Long double	3.4×10^{-4932}.... 1.1×10^{4932}	19 dígitos	10

Caracteres (char)

Un carácter es cualquier elemento de un conjunto de caracteres predefinidos o alfabeto. C procesa datos carácter (tales como texto) utilizando el tipo de dato char. En unión con la estructura *array*, que se verá posteriormente, se puede utilizar para almacenar cadenas de caracteres (grupo de caracteres). Se puede definir una variable carácter escribiendo:

```
char dato_car;
char letra = 'A';
char respuesta = 'S';
```

Internamente, los caracteres se almacenan como números. La letra A, por ejemplo se almacena internamente como el número 65, la letra B es 66, la letra C es 67, etc. El tipo char representa valores en el rango –128 a + 127 y se asocian con el código ASCII.

Dado que el tipo char almacena valores en el rango de –128 a + 127, C proporciona el tipo unsigned char para representar valores de 0 a 255 y así representar todos los caracteres ASCII. Puesto que los caracteres se almacenan internamente como números, se pueden realizar operaciones aritméticas con datos tipo char. Por ejemplo, se puede convertir una letra minúscula a una letra mayúscula A, restando 32 del código ASCII. Las sentencias para realizar la conversión:

```
Char car_uno  = 'a';
.....
car_uno  =  car_uno + 32;
```

Esto convierte a (código ASCII 97) a A (código ASCII 65). De modo similar, añadiendo 32 convierte el carácter de letra mayúscula a minúscula:

```
Car_uno  =  car_uno + 32;
```

Como los tipos char son subconjuntos de los tipos enteros, se puede asignar un tipo char a un entero. Por ejemplo:

```
Int suma  =  0;
Char valor;
.....
scanf ("%c", &valor);          /* función estándar de entrada*/
suma  =  suma + valor;         /* operador...*/
```

Existen caracteres que tienen un propósito especial y no se pueden escribir utilizando el método normal. C proporciona secuencias de escape. Por ejemplo, literal carácter de un apostrofe se puede escribir como

'\''

y el carácter nueva línea

'\n'

Secuencia de escape	Significado
\a	Alarma
\b	Retroceso de espacio
\f	Avance de página
\n	Retorno de carro y avance de línea
\r	Retorno de carro
\t	Tabulación
\v	Tabulación vertical
\\	Barra inclinada
\?	Signo de interrogación
\"	Dobles comillas
\000	Número octal
\xhh	Número decimal
\0	Cero, nulo (ASCII 0)

- El tipo de dato Lógico

Los compiladores de C que siguen la forma ANSI no incorporan el tipo de dato **lógico** cuyos valores son "verdadero" (*true*) y " falso" (*false*). El lenguaje C simula este tipo de dato tan importante en la estructura de control (if, while...). Para ello utiliza el tipo de dato int. C interpreta todo valor distinto de cero como "verdadero" y el valor 0 "falso". De esta forma se pueden escribir expresiones lógicas de igual forma que en otros lenguajes de programación se utiliza *true* y *false*. Una expresión lógica que se evalúa a "0" se considera falsa; y una expresión lógica que se evalúa a 1 (o valor entero distinto de 1) se considera verdadera.

En C, se puede definir un tipo que asocia valores enteros constantes con identificadores, es el tipo enumerado. Para representar los datos lógicos en C, el sistema usual es definir un tipo enumerado Boolean con dos identificadores *false* (valor 0) y *true* (valor 1) de la forma siguiente:

enum Boolean { FALSE, TRUE };

Esta declaración hace a Boolean un tipo definido por el usuario con literales o identificadores (valores constantes) TRUE y FALSE.

- Escritura de valores lógicos

La mayoría de las expresiones lógicas aparecen en estructuras de control que sirven para determinar la secuencia en que se ejecutan las sentencias C. Raramente se tiene la necesidad de leer valores lógicos como dato de entrada o de visualizar valores lógicos como resultado de programas. Si es necesario, se puede visualizar el valor de la variable lógica utilizando la función para salida printf(). Así, si encontrado es false, la sentencia

```
Printf (" el valor de encontrado es %d\n", encontrado);
```

Visualizara

```
El valor de encontrado es 0
```

- Constantes

En C existen cuatro tipos de constantes:

- constantes literales,
- constantes definidas,
- constantes enumeradas,
- constantes declaradas.

Las constantes literales son las más usuales; toman valores como 45.32564,222 o bien "introduzca sus datos" que se escriben directamente en el texto del programa. Las constantes definidas son identificadores que se asocian con valores literales constantes y que toman determinados nombres. Las constantes declaradas son como variables: sus valores se almacenan en memoria, pero no con una secuencia de otros nombres, tales como Azul, Verde, Rojo y Amarillo.

- Constantes literales

Las constantes literales o constantes, en general, se clasifican también en cuatro grupos, cada uno de los cuales puede ser de cualquiera de los tipos:

- Constantes enteras,
- Constantes caracteres,
- Constantes de coma flotante,
- Constantes de cadena.

- Constantes enteras

La escritura de constantes enteras requiere seguir unas determinadas reglas:

- No utilizar nunca comas ni otros signos de puntuación en números enteros o completos.

  123456 *en lugar de* 123.456

- Para forzar un valor al tipo long, termina con una letra L o 1. por ejemplo,

  1024 *es un tipo entero* 1024L *es un tipo largo* (long)

- Para forzar un valor al tipo unsigned, terminarlo con una letra mayúscula U. Por ejemplo, 4352U.

- Para representar un entero en octal (base 8), éste debe de estar precedido de 0.

  Formato decimal 123
  Formato octal 0777 (están precedidas de la cifra 0)

- Para representar un entero en hexadecimal (base 16), este debe de estar precedido de 0x.

  Formato hexadecimal 0XFF3A (están precedidas de "Ox" o bien "OX")

Se pueden combinar sufijos L(1), que significa *long* (largo), o bien U (u), que significa *unsigned* (sin signo).

3455UL

- Constantes reales

Una constante flotante representa un número real; siempre tiene signo y representa aproximaciones en lugar de valores exactos.

82.347 .63 83. 47e-4 1.25E7 61.e+4

La notación científica se representa con un exponente positivo o negativo.

2.5E4 *equivale a* 25000
5.435E-3 *equivale a* 0.005435

Existen tres tipos de constantes:

Float	4 bytes
Double	8 bytes
Long double	10 bytes

- Constantes carácter

Una constante carácter (char) es un carácter del código ASCII encerrado entre apóstrofe.

'A' 'B' 'C'

Además de los caracteres ASCII estándar, una constante carácter soporta caracteres que no se pueden representar utilizando su teclado, como, los códigos ASCII altos y las secuencias de escape.

Así, el carácter sigma (Σ –código ASCII 228, hex E4–se representa mediante el prefijo \x y el numero hexadecimal del código ASCII, por ejemplo,

Char sigma = ' \xE4';

Este método se utiliza para almacenar o imprimir cualquier carácter de la tabla ASCII por su numero hexadecimal. En el ejemplo anterior, la variable sigma no contiene cuatro caracteres sino únicamente el símbolo sigma.

código de escape	significado	código ASCII Dec	Hec
'\ n'	nueve línea	13 10	0D 0A
'\ r'	retorno de carro	13	0D
'\ t'	tabulación	9	09
'\ v'	tabulación vertical	11	0B
'\ a'	alerta (pitido sonoro)	7	07
'\ b'	retroceso de espacio	8	08
'\ f'	avance de pagina	12	0C
'\ \'	barra inclinada inversa	92	5C
'\ ' '	comilla simple	39	27
'\ "'	doble comilla	34	22
'\ ?'	signo de interrogación	63	3F
'\ 000'	numero octal	*Todos*	*Todos*
'\ xhh'	numero hexadecimal	*Todos*	*Todos*

Un carácter que se lee utilizando una barra oblicua (\) se llama *secuencia* o *código de escape.*

```
/ * Programa : Pruebas código de escape * /
# include <stdio.h>

int main ( )
{
  char alarma = '\ a';          / * alarma * /
  char bs = '\ b';              / * retroceso de espacio * /
  printf ( "%c", alarma,bs);
  return 0;
}
```

- Aritmética con caracteres C

Dada la correspondencia entre un carácter y su código ASCII, es posible realizar operaciones aritméticas sobre datos de carácter. Observe el siguiente segmento de código.

```
char c;
C = 'T' + 5;      / * suma 5 al carácter ASCII * /
```

Realmente lo que sucede es almacenar Y en C. El valor ASCII de la letra T es 84, y al sumarle 5 produce 89, que es el código de la letra Y. A la inversa, se puede almacenar constantes de carácter en variables enteras. Así,

Int j = 'p'

Donde asigna el valor 80 a j.

- Variables

Es una posición de memoria con nombre donde se almacenara un dato según el tipo previamente definido.

El nombre de la variable puede comenzar con letras o guión bajo, no con números.

código	Tipo de variable
%d	Entero decimal
%o	Octal
%x	Hexadecimal
%u	Entero sin signo
%c	Carácter
%e	Float con notación científica
%f	Float con notación decimal
%g	Float con notación científica o decimal, la mas corta
%s	Cadena de caracteres
%lf	Double

Para las entradas y salidas de datos puede ser necesario secuencias de escape para una mayor legibilidad de las pantallas

código de escape	Significado
\a	Alarma
\b	Retroceso de espacio
\f	Avance de pagina
\n	Retorno de carro y avance de línea
\r	Retorno de carro
\t	Tabulación
\v	Tabulación vertical
\\	Barra inclinada
\?	Signo de interrogación
\"	Dobles comillas
\000	Numero octal
\xhh	Numero hexadecimal
\0	Cero, valor nulo

Operadores

- Operadores de asignación

Símbolo	uso	Descripción
=	A = b	Asigna a A el valor de b
* =	A * = b	Multiplica A por b y guarda en A
/ =	a / = b	Divide a por b y guarda en a
% =	a % = b	Guarda en a el resto de a / b
+ =	a + = b	Suma a + b y lo guarda en a
- =	a - = b	Resta a – b y lo guarda en a

- Operadores aritméticos

operador	Tipo enteros	Tipo reales
+	Suma	Suma
-	Resta	Resta
*	Producto	Producto
/	Cociente	Cociente
%	Resto	

Existen dos operadores especiales y con los que hay que manejarse con cuidado

a = a + 1; es lo mismo que a + +
a = a –1; es lo mismo que a - -

Ahora prueben que resultados dan las siguientes operaciones siendo a = 1 y b = 3

a = + + b; a = b + +;
a = - - b; a = b - -;

que valor toma a y b en cada caso luego de realizar la operación?

- Operadores relacionales

Operador	Significado	Ejemplo
= =	Igual a	A = = b
! =	Distinto a	A ! = b
>	Mayor que	A > b
<	Menor que	a < b
> =	Mayor e igual que	a >= b
< =	Menor e igual que	a <= b

- Operadores lógicos

Operador	Significado	
!	Negación	not
&&	Y	and
||	O	or

- Operador condicional

Este operador reemplaza al if else de una estructura condicional normal

a > b ? 5 : 10; si a es mayor que b devuelve 5 de lo contrario 10

- Operador coma

permite escribir mas de una sentencia en una sola línea

i = 12, j = 13; seria lo mismo que i = 12;
j = 13;

- Operador sizeof

devuelve el tamaño en bits de una variable, array, etc.

Sizeof (variable);

- Operación de conversión explicita

C puede convertir el contenido de las variables a otros tipos de datos

Se debe escribir el tipo deseado y luego la variable sobre la que se debe aplicar

(int) a;
(float) b;

Estructuras

Estructuras secuenciales: conjunto de instrucciones con un ordenamiento lógico para lograr un fin.

- Estructuras de selección

La sentencia if tiene dos alternativas o formatos posibles

If (condición) acción

If (condición) acción_1 else acción_2

Donde condición es una expresión lógica, accion1 es el conjunto de sentencias a ejecutarse en caso de ser afirmativa la condición y accion2 es el conjunto de sentencias a ejecutarse en caso de ser falsa dicha condición.

Las sentencias van incluidas dentro de { } en caso de ser de mas de una línea de código.

Puede darse el caso de if anidados, donde dentro del else de un if nazca otro if, deberá en este caso ser cuidadoso de donde comienza y termina cada una de las estructuras del if.

Así mismo, puede en la misma expresión de condición conectar varias condiciones con operadores lógicos.

Cuidado, el orden de los operadores pude ser critico en algunas situaciones

En caso de tener varios caminos a tomar según el valor que puede tomar una variable, es conveniente el uso del operador switch.
El valor de la variable o expresión que involucra el switch debe ser int o char.

```
Switch ( condición )
{
        case etiqueta:  sentencias;
                                        break;
        case etiqueta:  sentencias;
                                        break;
        case etiqueta:  sentencias;
                                        break;
        case etiqueta:  sentencias;
                                        break;
        default:                        sentencias;
}
```

El break es obligatorio a fin de evitar que luego de ejecutar un as sentencias de un case, no siga con las que se encuentran debajo, sino que corte la instrucción switch.
La etiqueta default no es obligatoria pero si conveniente y sirve para indicar que hacer si la expresión no toma ningún valor preestablecido en los case.

Por supuesto las sentencias es conveniente que se escriban entre {}.

Puede darse la siguiente posibilidad de switch

```
Switch (c)
{
case 0: case 1 : case 3 :       sentencias;
                                                    break;
case 2: case 4 : case 5 :       sentencias;
                                                    break;
default:                        sentencias;
                                                    break;
}
```

o sea que ante distintos valores que pueda tomar c haga lo mismo.

- Estructuras de iteración

Sentencia while

```
While ( condición )
{
sentencias;
}
```

en esta estructura las sentencias se ejecutan mientras la condición es verdadera y termina cuando pasa a falsa y se ejecuta la próxima sentencia fuera del ciclo while.
La variable que representa la condición del bucle se denomina variable de control del bucle y debe ser

- Inicializada
- Comprobada
- Actualizada dentro de las sentencias del bucle

También puede darse el caso de los ciclos conocidos como do - while donde el control del ciclo se hace debajo de las sentencias.
Como característica fundamental de este tipo de ciclos es que al menos una vez se van a ejecutar las sentencias definidas en el bucle.

```
Do
{
        sentencias;
}
while (condición);
```

Dentro de los ciclos también existe la sentencia *break* para realizar una terminación anormal del bucle.

Sentencia for

Esta sentencia se utiliza para la realización de un numero finito y conocido de ciclos de las sentencias definidas dentro de el.

```
For (valor inicial; condición final; incremento)
{
sentencias;
}
```

Hay que tener cuidado que ninguna operación interna del bucle modifique el valor de la variable de control, haciendo a esta incontrolable.

El siguiente ejemplo

```
For (;;)
{
sentencia;
}
```

Dará un resultado infinito de ejecuciones de las sentencias, debiendo finalizarlo el operador por teclado (CTRL+C)

en cambio

```
for (i=1;i<=10;++);
{
sentencia;
}
```

No se ejecutara nunca por el ";" luego del for

Por supuesto y como parte de todo programa dentro de una estructura de iteración puede existir una, ninguna o varias estructuras de iteración, por lo que debemos tener los mismos cuidados que con los if anidados.

Funciones

Una función es un conjunto de sentencias que pueden ser llamadas desde cualquier parte de un programa.
Las funciones en C no pueden anidarse o sea declarase dentro de otra función.

La estructura de una función es la siguiente

Tipo de resultado nombre de la función (lista de parámetros)

```
{
        declaración de las variables
        sentencias
        valor devuelto
}
```

analicemos los aspectos mas sobresalientes de una función

Tipo de resultado :	tipo de dato que devuelve la función
Nombre de la función :	nombre arbitrario de la función
Lista de parámetros:	parámetros tipificados
Cuerpo de la función :	encerrado entre { } sin ";" final
Paso de parámetros:	generalmente por valor
Declaración local de variables :	solo se activan en esta función
Valor devuelto por la función:	mediante ***return*** devuelve el valor

Una llamada a una función produce su ejecución y luego retorna a donde fue llamada con un valor resultante mediante **return**.

La longitud de la definición de una función no debería sobrepasar, en lo posible, el largo de una pantalla

Nombre de una función

Comienza con una letra o subrayado (_), distinguiendo entre mayúsculas y minúsculas

Ej
```
Int   Calculo_suma ( int a, int b);
Int   _suma (int a, int b);
```

Tipo de dato de retorno

Si la función no devuelve un valor int se debe especificar que tipo de valor devolverá.

- double
- float
- char
- punteros

si la función **NO** devuelve ningún resultado se colocara **Void**

Resultados de una función

Una función puede devolver un único resultado con una sentencia return.

Llamada a una función

Las funciones para poder ser ejecutadas deben ser llamadas o invocadas, normalmente se lo hace desde la función main( ).

La función que llama a otra se denomina **llamadora** y la función controlada se denomina **llamada.**

Prototipo de una función

Un prototipo declara una función y proporciona información suficiente al compilador para verificar que la función esta siendo llamada correctamente con respecto al número y tipo de parámetros y el tipo devuelto por la función.
Normalmente se sitúan al principio de un programa, antes de la función main( ).

```
Float  Calculo_area ( float a, float b);
```

Cuando una función se declara, se esta proporcionando el nombre y se listan las características de la misma.
Cuando una función se define, se indica el nombre y se reserva el espacio de memoria para esa entidad.

Un formato especial de prototipo es aquella con un numero no especificado de parámetros

```
Int  muestra ( int a, int b,....);
```

Se representa por medio de puntos suspensivos y es necesario utilizar # include < stdarg.h >.

Tipos de pasaje de parámetros a una función

Paso de parámetros por valor

En este método se realiza una copia de los valores que se vienen manejando en el programa. Estos dentro de la función podrán ser modificados por medio de cálculos o asignaciones, pero cuando la función devuelve el control a la función llamadora, recuperan los valores originales que tenían antes de ser modificados por la función.

Paso de parámetros por referencia

Este método se utiliza cuando una función debe modificar el valor de un parámetro recibido y devolverlo modificado a la función llamadora.
El compilador pasa la dirección donde se encuentra el valor en cuestión y no una copia del mismo, como lo hacia el pasaje por valor.
La notación en la sintaxis varia para pasar una variable por referencia debe preceder al nombre de la misma el símbolo & y el parámetro correspondiente a la función debe declararse como puntero *.

```
Float x;
Int y;
Entrada (& x,& y);
.....
void ( float *  x, float *  y );
```

Parámetros *const* de una función

Para añadir seguridad adicional a las funciones, se puede agregar el especificador *const*, que indica al compilador que solo es de lectura dentro del interior de la función.
Si se intentase escribir en este parámetro dentro de la función dará error.

Resumen del comportamiento de los diferentes tipos de parámetros

Parámetro especificado como	Item pasado por	Cambia item dentro de la función	Modifica parámetros al exterior
Int item	Valor	Si	No
Const int item	Valor	No	No
Int* item	Por dirección	Si	Si
Const int* item	Por dirección	No su contenido	No

Ámbito o alcance de una variable

Si una función reconoce a una variable se dice que esa variable es visible a esa función. Existen cuatro tipos de ámbitos

Programa
Archivo fuente
función
Bloque

Las variables que tienen ámbito de programa son conocidas como globales, se la define antes de la función main ( ) y fuera de cualquier función.
Esta función será visible para todo el programa y para cualquier función.

Las que tienen ámbito en el archivo fuente van precedidas en su declaración por la palabra static y se escriben fuera de cualquier función, por ejemplo static int x.
Las que se declaran con ámbito dentro de una función se dicen que son locales a ellas y solo son visibles dentro de ellas; **no** se las puede utilizar fuera de ellas.
Estas variables solo existen en memoria cuando la función esta activada, por lo tanto sus nombres pueden no ser únicos.

Las variables declaradas dentro de un bloque solo son visibles en el, por ejemplo dentro del bloque de instrucciones de las ramas de una estructura condicional encerrado por las { }.

Clases de mantenimiento

Variables automáticas

Son las que se declaran dentro de una función. Opcionalmente se puede anteponer a su declaración la palabra reservada auto

Auto int i; es lo mismo que int i;

Variables externas

Se presenta este caso cuando una función necesita utilizar una variable que otra función inicializa.
Como las variables locales son temporales, lo que hay que hacer es utilizar una variable definida en otro archivo.
Entonces a la variable local se la declarara *extern* así el compilador sabe que el espacio de la variable esta definido en otro lugar.

Variables registro

Estas variables se almacenan en uno de los registros hardware del microprocesador, será local a una función, nunca global.
El uso de este tipo de variable no garantiza que un valor se guarde, solo lo hará si existe un lugar para ello disponible.
Por ejemplo register int i;

Variables estáticas

Son opuestas a las automáticas, su valor no se borra cuando la función finaliza.

```
{
Static int i =25;
}
```

Concepto y Uso de Funciones de Biblioteca

Todas las versiones de C ofrecen una biblioteca estándar de funciones en tiempo de ejecución que proporcionan soporte a las operaciones mas frecuentes y con solo llamarlas permiten realizar operaciones sin necesidad de escribir su código.

Estas funciones estandares o predifinidas se dividen en grupos y se escriben encerrado entre corchetes su nombre
Ej:

< assert.h >	< ctype. H >	< errno. h >	< float. h >
< limits.h >	< math. H>	< setjmp. h >	< signal. h >
< stdarg. h >	< stdef. h >	< stdio. h >	< string. h >
< time. h >	<conio.h>		

En los módulos de programa se pueden incluir líneas # include con el archivo de cabecera correspondiente
Se pueden incluir tantos archivos de cabecera como sean necesarios.

Funciones de carácter

El archivo de cabecera < type.h> define un grupo de funciones y macros de manipulación de caracteres
Devuelve 0 si es falso u otro valor si es verdadero

Función	Prueba (test) de c
int isalpha (int c)	Devuelve verdadero si c es una letra
int isdigit (int c)	Devuelve verdadero si c es un digito entre 00 y 9
int issupper (int c)	Devuelve verdadero si c es letra mayúscula
int islower (int c)	Devuelve verdadero si c es letra minúscula
int iscntrl(int c)	Devuelve verdadero si c es un carácter alfabético o un digito entre 0 y 9
int isxdigit (int c)	devuelve verdadero si c es un carácter de control (códigos ASCII de 0 a 31)
int isprint (int c)	Devuelve verdadero si c es un digito hexadecimal
int isgraph (int c)	Devuelve verdadero si c es un código imprimible incluyendo espacio (código ASCII 32 a 127)
int isspace (int c)	Devuelve verdadero si c es un código imprimible excepto espacio
int ispunct (int c)	Devuelve verdadero si c es espacio, nueva línea, retorno de carro, tabulación
int toupper (int c)	Devuelve verdadero si c es un carácter de puntuación
int tolower (int c)	Convierte el carácter c a mayúscula
	Convierte el carácter c a minúscula

Funciones numéricas

El archivo de cabecera < math.h> define un grupo de funciones para el manejo numérico.

Funciones matemáticas

Función	Operación que realiza
ceil(x)	Redondea al entero mas cercano
fabs(x)	Devuelve el valor absoluto de x (valor positivo)
floor(x)	Redondea por defecto al entero mas próximo
fmod(x,y)	Calcula el resto en coma flotante de x/y
pow(x,y)	Calcula x elevado a la y donde y es entero
pow10(x)	Calcula 10 a la x donde x debe ser entero
sqrt(x)	Devuelve la raíz cuadrada de x

Funciones trigonométricas

Función	Operación que realiza
acos(x)	Calcula el arco coseno de x, -1< x <1
asin(x)	Calcula el arco seno de x, -1< x < 1
atan(x)	Calcula el arco tangente
atan2(x,y)	Calcula el arco tangente de x dividido por y
cos(x)	Calcula el coseno de x, se expresa en radianes
sin(x)	Calcula el seno de x, se expresa en radianes
tan(x)	Calcula la tangente de x, se expresa en radianes

Funciones logarítmicas y exponenciales

Función	Operación que realiza
exp(x)	Devuelve ex
expl(x)	Devuelve ex utilizando un valor long double
log(x)	Devuelve el logaritmo natural de x
logl(x)	Devuelve el logaritmo natural de x utilizando un valor long double
log10(x)	Devuelve el logaritmo decimal de x
log100l(x)	Devuelve el logaritmo decimal de x utilizando un valor long double

Funciones aleatorias

El archivo de cabecera < stdlib.h > agrupa a estas funciones.

Función	Operación que realiza
rand(void)	Genera un numero aleatorio
Randomize(void)	Inicializa al generador de rand ( ) llamando a la función time por lo cual hay que incluir: <time.h>
srand (semilla)	Inicializa al generador de rand ( ) desde el valor semilla
Random (num)	Genera un numero aleatorio entre 0 y num

Funciones de fecha y hora

El archivo de cabecera < time.h > agrupa a estas funciones, la fecha se guarda con el formato del calendario gregoriano

Función	Operación que realiza
Clock(void)	Determina el tiempo transcurrido desde el principio de la ejecución del programa
time(hora)	Determina la hora actual
Localtime(hora)	Convierte la fecha y hora en una estructura de tipo tm
Mktime((t)	Convierte la fecha en formato de calendario

Funciones de utilidad

El archivo de cabecera <stdlib.h> agrupa a estas funciones.

Función	Operación que realiza
abs(n)	Devuelve el valor absoluto de n
labs(n)	Devuelve el valor absoluto de n en long
div(num,denom)	Calcula el cociente y el resto de num/denom, dejándolos en las variables quot y en rem
ldiv(num,denom)	Idem anterior pero con variables tipo long

Biblioteca string.h

FUNCION	Cabecera de la función y prototipo
Mencpy()	Void* mencpy(void*s1,const void*s2,size_t n) Reemplaza los primeros n bytes de *s1 por los de *s2
strcat	Char strcat*(char **destino, const char** fuente) Concatena, agrega fuente al final de destino
Strchr()	Char*strchr(char* s1,int ch) Devuelve un puntero a la primera ocurrencia de ch en s1, devuelve null si no esta
Strcmp()	Int strcmp(const char *s1, const char *s2) Compara alfabéticamente ambas cadenas Devuelve 0 si s1 =s2 ; <0 si s1 < s2 ; >0 si s1 > s2
Strcmpi()	Int strcmpi(const char *s1,const char *s2) Igual que strcmp pero no distingue entre mayúsculas y minúsculas
Strcpy()	Char *strcpy(char *destino, char *origen) Copia origen en destino
Strcspn()	Size_t strcspn(const char* s1, const char* s2) Devuelve la longitud de la subcadena mas larga s1 que comienza con el caracter s1[0] y no contiene ningun carater de la cadena s2
Strlen()	Size_tstrlen(const char* s1) **Devuelve la longitud de la cadena s1**
Strncat()	Char * strncat(char * s1, const char *s2,size_t n) Añade los primeros n caracteres de s2 a s1
Strncmp()	Int strncmp(const char* **s1,const char*** s2, size_t n) Compara s1 en la subcadena de n caracteres de s2 Devuelve 0 si s1 =s2 ; <0 si s1 < s2 ; >0 si s1 > s2
Strnset()	Char *strnset (char *s, int ch, size_t n) Copia n veces el caracter ch en la cadena s a partir de la posicion inicial s[0]
Strpbrk()	Char *strpbrk(const char *s1,const char *s2) Devuelve la dirección de la primera ocurrencia en s1 de cualquiera de los caracteres de s2, null si no existen
Strrchr()	Char* strrchr(const char *s, int c) Devuelve un puntero a la ultima ocurrencia de c en s, null si no esta. La búsqueda la hace de atrás hacia adelante
Strspn()	Size_t strspn(const char * s1, const char *s2) Devuelve la longitud izquierda de la subcadena mas larga de s1 que contenga únicamente caracteres de s2
Strstr()	Char * strstr(const char* **s1, const char*** s2) Busca la cadena s2 en s1 y devuelve un puntero a los caracteres donde se encuentra s2
Strtok()	Char * strtok(char *s1,const char *s2) Analiza la cadena s1 en tokens, esta limitado por caracteres de la cadena s2. la llamada inicial devuelve la dirección del primer token y de allí sucesivamente. Esta función modifica la cadena s1 colocando null en cada separador encontrado.

Const

Se utiliza para definir valores que no van a ser modificados durante el programa, son en general datos de entrada a una función.

Formas de lectura de una cadena

Para la lectura o ingreso de una cadena es conveniente lo siguiente

```
# include<string.h>
#include<conio.h>
#include<stdio.h>

void main ()
{
char cadena[30] ;
int  i;

for ( i=1;i<=3;i++ )
        {
         printf (" ingrese el nombre del empleado");
        fflush(stdin);
        gets ( cadena);
        printf (" el nombre ingresado es  %s ", cadena);
        }
}
```

donde fflush limpia el buffer de entrada
gets reemplaza a scanf que veníamos utilizando para las variables numéricas

EJERCICIO:
Ingrese un nombre compuesto (Ana María) y léalo en un programa desde la pantalla con gets y con scanf, habiendo declarado como char (%s)
¿qué ocurre? ¿qué diferencias encuentra?.

si deseásemos leer carácter por carácter de una variable char de entrada deberíamos utilizar la función getchar( )

Visibilidad de una función

El ámbito de un elemento es su visibilidad desde otras partes del programa y la duración de un elemento es su tiempo de vida, lo que implica no solo cuanto tiempo existe la variable, sino cuando se crea y cuando se hace disponible.

Recursividad

Una función recursiva es aquella que se llama a si misma. **Directa** es cuando lo hace desde el propio cuerpo de la función, e **indirecta** cuando implica a más de una función.
Toda función recursiva debe tener una condición de terminación, porque sino seria su continuidad indefinidamente.
Una función típica de recursividad es el calculo del factorial.

Arrays

Un array es una secuencia de datos del mismo tipo, os datos se llaman elementos de un array y se enumeran consecutivamente 0, 1, 2,... etc. el tipo del dato almacenado en un array pude ser de cualquier tipo de dato de C.

Otros nombres que reciben los arrays unidimensionales son tablas o vectores.

La numeración de los elementos se denomina índice o subíndice de un array y solo determina la posición del elemento dentro del array y no su valor o contenido.

Entonces dado el array denominado pp de 5 elementos

15	45	56	78	89
0	1	2	3	4

Tendríamos que pp [1] = 45

O sea que el array llamado pp en la posición 1 tiene un elemento cuyo valor es 45

Declaración de un array

Un array se declara de la misma forma que cualquier otra variable, solo que debemos indicarle al compilador su tamaño.

Tipo nombre del array [tamaño]

```
Int pp[5];
```

El subíndice puede ser referenciado de distintas maneras para cargar o recuperar el valor almacenado en esa posición, por ejemplo

```
Edad[4];
Ventas[mes+1];
Bonos[mes];
Salario[mes[i]+1];
```

C no comprueba que los índices del array estén dentro del rango definido, o sea si deseo leer la posición 8 de pp, no dará error de lectura pero si luego indicara una falla en el programa.

El operador sizeof devuelve el numero de bytes necesarios para contener su argumento, es así que podemos usarlo para conocer el tamaño de un array.

Si tenemos int a[100] y le aplicamos la función sizeof[a] devolverá un valor de 200.

Formas de inicializar un array

Si se deben asignar valores iniciales a un array podemos emplear distintos métodos

```
pp[0] = 15;
pp[1] = 45;
pp[2] = 56;
pp[3] = 78;
pp[4] = 89;
```

También podríamos escribir lo siguiente, determinando automáticamente el tamaño del array

```
int pp[5]  = { 15, 45, 56, 78, 89 };
char ww[ ] = { 'a', 's', 'I' };
int pp[] = { 15, 45, 56, 78, 89 };
```

En **C** los arrays de caracteres, las cadenas, se caracterizan por tener un elemento nulo final, que indica el fin de la cadena.

Otra manera de inicializar, escribir y leer arrays de gran tamaño, es mediante un ciclo *for*.

También se pueden asignar variables simbólicas

```
# define ene 31
# define feb 28
# define mar 31

int meses[12] = { ene, feb, mar,.....};
```

Arrays de caracteres y cadenas

Una cadena es un conjunto de caracteres, tal como << abcdef.>> las cadenas contienen un elemento final nulo que indican fin de la cadena.

char cadena[ ] = " abcdef ";

Las cadenas se deben almacenar en un array de caracteres, pero no todos los arrays de caracteres contiene cadenas.

Para copiar una cadena a otra variable cadena se debe utilizar la función strcpy( ), donde si quiero copiar "hola" a la variable saludos seria:

Strcpy(saludos,"hola");

Para evitar un error debemos tener en cuenta que el tamaño del receptor sea de igual o mayor tamaño que la cadena que vamos a mover.

Arrays multidimensionales

Estos arrays contienen mas de una dimensión, los mas conocidos son llamados matrices.
Por lo tanto estos arrays tiene mas de un subíndice.
Su declaración será pues:

tipo nombre [numero de filas] [numero de columnas];
int matriz [5] [6];

En realidad un array de dos dimensiones es un array de arrays que para representarlo en una forma mas cómoda se lo hace por medio de una matriz.

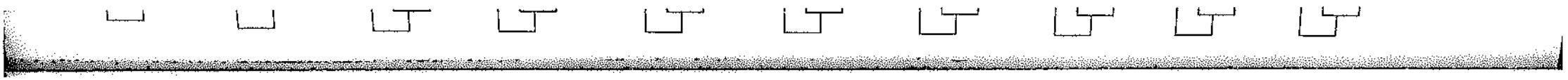
La inicialización de las matrices puede ser de la misma forma que los vectores, teniendo en cuenta por ejemplo

```
Int matriz [3] [4] = { {10,20,30,110},
                       {40,50,60,101},
                       {70,80,90,100}  }
```

El acceso a las matrices para su escritura o lectura será similar a los vectores, teniendo en cuenta la necesidad de al menos dos ciclos *for*, uno para las filas y otro para las columnas en el caso de una matriz bidimensional.

Utilización de arrays como parámetros

En **C** todos los arrays se pasan por referencia, o sea que se pasa la dirección del mismo y es posible de modificar su contenido en la función llamada y que al regresar a la función llamadora mantenga este valor modificado.



•*Programación lógica*

- **Consideraciones básicas**
- **Diagramación en jackson**
- **Estructuras secuenciales**
- **Estructuras de selección o condicionales**
- **If, switch**
- **Estructuras de iteración**
- **While,do,for**
- **Máximos y mínimos**
- **Tipos de datos estructurados**
- **Vectores**
- **Matrices**
- **Ejercicios y ejemplos**

Consideraciones básicas

A lo largo del tiempo se ha entendido al tema de programación como el hecho de conocer un lenguaje y a travez de él lograr la solución a determinados problemas.

Este concepto hoy se ha modificado, entendiendose como programación a la posibilidad de la resolución del problema, independientemente del lenguaje a utilizar.

La solución de problemas por medio de la programación es una tarea que requiere meditación, planificación, lógica y perseverancia.por lo cuál es una actividad que tiene características de mucha creatividad y desafío personal.

Para la resolución de un problema pues, es necesario primero entenderlo, luego crear el **algorítmo** necesario y por último volcarlo a un lenguaje determinado.es aquí donde se ve que no basta conocer sólo un lenguaje sino que es necesario conocer una técnica de programación.

Algoritmo, es un conjunto finito de pasos, procesados en un tiempo finito despues del cuál se obtiene la solución deseada.
O sea que todo algoritmo debe tener un principio y un fin determinado.

Este curso buscará enseñar a resolver problemas mediante técnicas de programación, buscará enseñar a **pensar** esas resoluciones, lo intentará a travez de un algoritmo.

Como dijimos antes existen etapas para la solución del problema, las cuales podriamos definir como la estrategia a seguir

- **Determinar datos**
- **Determinar resultados**
- **Determinar los procesos que debemos aplicarle a esos datos para obtener esos resultados**

Los **datos** serán todos aquellos elementos que provengan del exterior de nuestro programa, y que su procesamiento en una forma determinada nos permitirá obtener los resultados requeridos que es lo que llamamos **información.**

Representación de los datos y de los resultados en un programa

Estos serán representados por medio de lo que conoceremos como variables, estas llevaran nombres claros y representativos de lo que contienen y podrán ser de distintos tipos
por ej

Dato o resultado	**Variable**	**Tipo**
Nombre empleado	Nomemp	Texto
Hs trabajadas	Hstrab	Numérica
Valor hora	Valhs	Numérica
Sueldo	Sdo	Numérica
Fecha de nacimiento	Fnac	Fecha y hora
Comentarios	Curric	Memo

Los tipos de datos estarán ligados al lenguaje a utilizar pero en general todos tienen los mismos tipos o muy parecidos.

Dentro de las variables nosotros podremos guardar datos, valores constantes y a otras variables que vayan surgiendo durante la ejecución de nuestro algoritmo.

A continuación los datos se verán afectados por procedimientos para obtener un resultado, es conveniente dividir estos procedimientos en procesos lo más pequeños posible a fin de tener mayor control sobre los mismos, y que una posterior modificación en una parte de él no repercuta en todo el resto negativamente, "divide y triunfaras ".

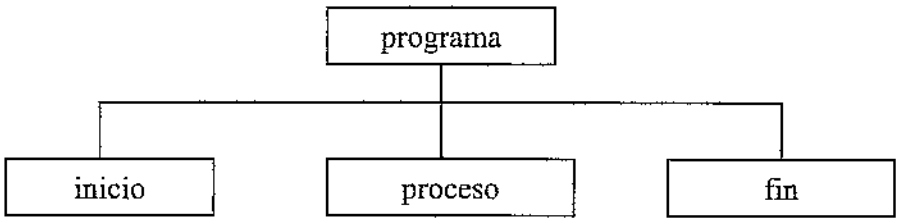
Lo que restaría es armar el algoritmo que no es más que colocar estos pequeños procesos en una forma lógicamente ordenada para poder obtener los resultados deseados.

DIAGRAMACION JACKSON

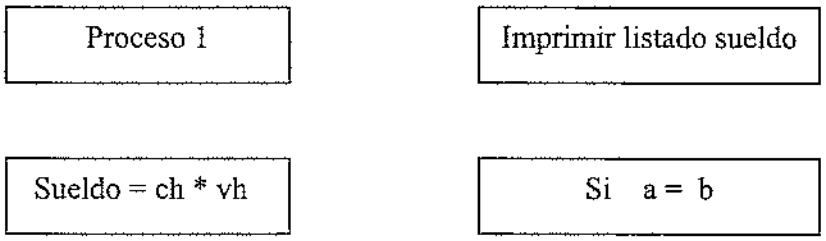
Esta es una forma más de representar la lógica de programación.

Un proceso es una secuencia de instrucciones que ocupan una cantidad de recursos del computador y permiten la solución de un problema, ya sea en función de un solo proceso o de varios subprocesos que al combinarse, correcta y lógicamente, generan la solución deseada.

La diagramación Jackson consiste en dibujar a todos sus elementos como rectángulos, que se encuentran ordenados en forma secuencial de izquierda a derecha y poseen distintos niveles entre sí.



En todos los casos dentro del rectángulo se escribe la instrucción, el procedimiento que se llama, etc.



- Tipos de Estructuras

Dentro de la programación estructurada reconocemos tres estructuras básicas

Estructuras secuenciales
Estructuras condicionales
Estructuras iterativas o de repetición

Forma de dibujar las distintas operaciones

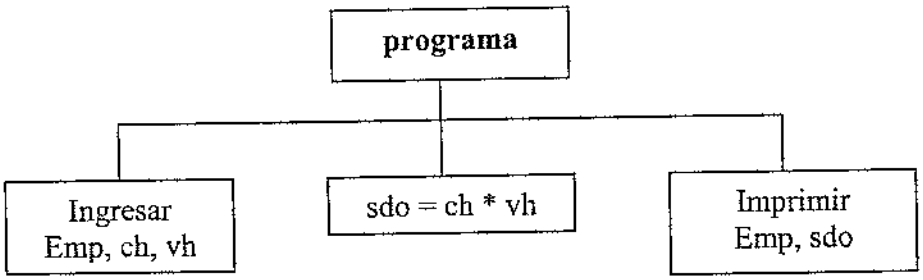
Secuenciales

Dado el valor de la hora y la cantidad de horas trabajadas por un empleado, calcular su sueldo

Datos : valor de la hora, vh
cantidad de horas trabajadas, ch

resultado : sueldo, sdo

proceso : sdo = ch* vh



En seudo código sería:

```
Comienzo
        Ingresar "el nro de empleado"
        Leer, emp
        Ingresar "la cantidad de horas"
        Leer,ch
        Ingresar "el valor de la hora"
        Leer vh
        Sdo = ch * vh
        Imprimir "el empleado, emp, cobra,  sdo,pesos"
fin
```

En lenguaje C sería

```
#include <stdio.h>
#include <conio.h>
void main()
{
int emp,ch,vh,sdo;
clrscr();
printf("ingrese el nro de empleado  ");
scanf("%d",& emp);
printf("ingrese la cantidad de horas  ");
scanf("%d",&ch);
printf("ingrese el valor de la hora  ");
scanf("%d",&vh);
sdo=ch*vh;
printf("el empleado %d, cobra %d pesos",emp,sdo);
getch();
}
```

```
//*****************************************************************
CON FUNCIONES
//*****************************************************************

#include <stdio.h>
#include <conio.h>

//////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(int*, int*, int*);
int calculo (int, int);
void informe (int, int);
void FIN (void);

void main()
{
 int emp,ch,vh,sdo;
 clrscr();
 //////////////////// LLAMADAS A FUNCIONES
 carga_datos(&emp,&ch,&vh);
 sdo=calculo(ch,vh);
 informe(emp,sdo);    // TAMBIEN SE PUEDE REEMPLAZAR LAS DOS ULTIMAS
 FIN();               // LLAMADAS POR ------> informe (emp, calculo(ch,vh));
}
```

```
//////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(int *x, int *y, int *z)
{
 printf("ingrese el nro de empleado  ");
 scanf("%d",x);
 printf("ingrese la cantidad de horas  ");
 scanf("%d",y);
 printf("ingrese el valor de la hora  ");
 scanf("%d",z);
}

////////////////////////////////////////////
int calculo (int x, int y)
{
 return (x *y);
}

////////////////////////////////////////////
void informe (int x, int y)
{
 printf("el empleado %d, cobra %d pesos",x,y);
}
////////////////////////////////////////////
void FIN(void)
{
 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}
```

EJERCICIOS DE VARIABLES Y ASIGNACIONES

1. Ingresar dos valores enteros y sumarlos
2. Ingresar tres valores, imprimir la suma total, sólo sabe sumar de a dos.
3. Ingresar tres valores y sumarlos, se puede sumar de a varios operandos.
4. Ingresar los lados de un triángulo calcular su perímetro
5. Ingresar dos lados de un triángulo rectángulo y calcular, la hipotenusa, el perímetro, la superficie.

6. Ingresar los lados de un rectángulo y calcular su diagonal principal, superficie y perímetro.

7. Ingresar dos valores, calcular su suma, su producto y la resta del 1ro menos el 2do.

8. Ingresar el valor de la hora y el tiempo trabajado por un operario, calcular su sueldo.

9. Ingresar el tiempo trabajado por un operario y si el valor de la hora es de 10 pesos, calcular su sueldo

10. Una concesionaria de autos paga a cada vendedor $ 500 por mes más un plus del 10 % del precio sobre cada vehículo vendido y un valor constante de 50 pesos por cada uno de ellos, sólo vende un tipo de vehículo, calcular su sueldo

Condicionales

La condición que se desea comprobar va adentro del rectángulo

Puede darse tres casos

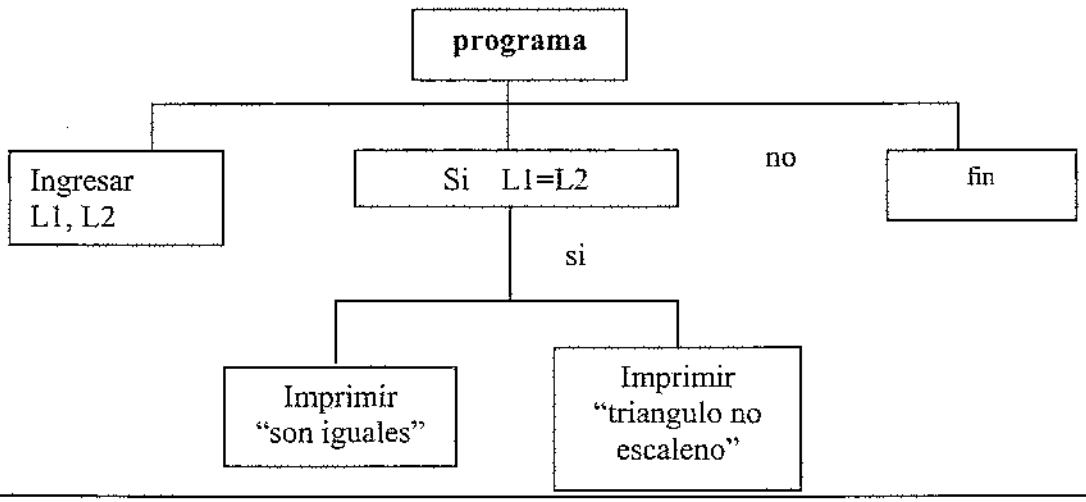
1.condicionales con salida por el verdadero de la condición especificada

Ingrese dos lados de un triangulo, indique si son iguales y por lo tanto que el triangulo no puede ser escaleno

Datos : lado 1 , L1
Lado 2 ; L2

Resultado : imprimir son iguales

Proceso : comparar L1 si es = a L2



En seudo código sería :

```
Comienzo
        Ingresar"ingrese el primer lado "
        Ingresar L1
        Ingresar "ingresar el segundo lado"
        Ingresar L2
        Si  L1 = L2 entonces
                Imprimir "son iguales"
                Imprimir "triangulo no escaleno"
        Fin si
fin
```

En lenguaje *C* sería

```
#include <stdio.h>
#include <conio.h>
void main()
{
int l1,l2;
clrscr();
printf("ingrese el primer lado  ");
scanf("%d",&l1);
printf("ingrese el segundo lado  ");
scanf("%d", &l2);
if(l1==l2)
        {
        printf("son iguales \n");
        printf("el triangulo no es escaleno");
        getch();
        }
}

//*********************************************************************
 CON FUNCIONES
//*********************************************************************
#include <stdio.h>
#include <conio.h>

//////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(int*, int*);
void informe(void);
```

```
void FIN (void);

void main()
{
int l1,l2;
clrscr();

carga_datos( &l1,&l2);  // LLAMADA A FUNCION
if(l1==l2)
        {
        informe();      // LLAMADAS A FUNCIONES
        FIN();
        }
}


//////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(int *x,int *y)
{
 printf("ingrese el primer lado  ");
 scanf("%d",x);
 printf("ingrese el segundo lado  ");
 scanf("%d", y);
}

////////////////////////////////////////////
void informe(void)
{
  printf("son iguales \n");
  printf("el triangulo no es escaleno");
}

////////////////////////////////////////////
void FIN(void)
{
  printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
  getch();
}
```

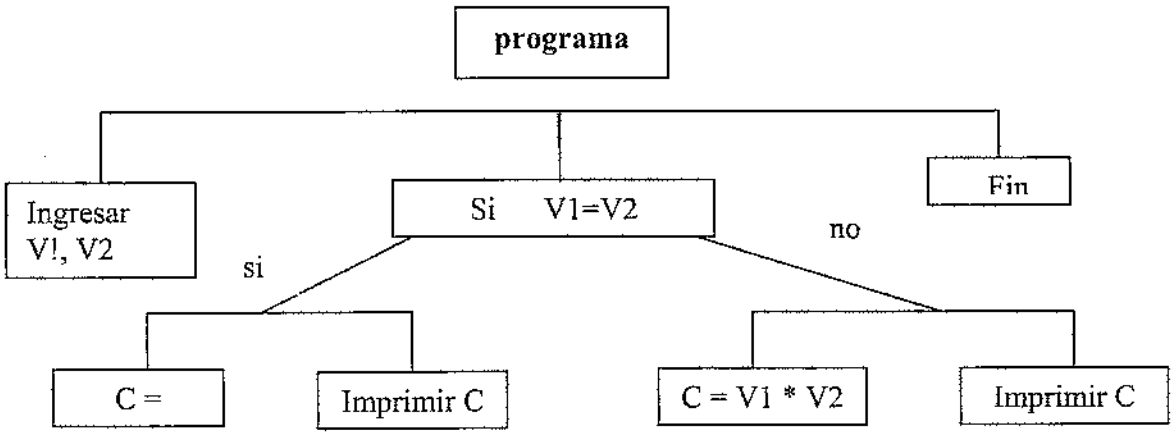
2. Condicionales con salida por el verdadero y por el falso de la condición especificada

Ingresar dos valores, sumarlos si son iguales y multiplicarlos si son distintos

Datos: valor 1, V1
Valor 2, V2

Resultado : realizar el producto si son distintos
Realizar la suma si son iguales

Proceso : C = V1 + V2
C = V1 * V2



En seudo código seria :

```
Comienzo
        Ingresar"ingrese el primer valor"
        Ingresar V1
        Ingresar "ingrese el segundo valor"
        Ingresar V2
        Si V1=V2 entonces
                C = V1 + V2
                Imprimir"son iguales y la suma es C"
        De lo contrario
                C = V1 + V2
                Imprimir"son distintos y el producto es C"
        Fin si
Fin
```

En lenguaje *C* sería

```
#include <stdio.h>
#include <conio.h>
void main()
{
int V1,V2,c;
clrscr();
printf("ingrese el primer valor \n");
scanf("%d", &V1);
printf("ingrese el segundo valor \n");
scanf("%d", &V2);
if(V1==V2)
        {
        c=V1+V2;
        printf("son iguales y la suma es %d ",c);  }

        else
        {
        c=V1*V2;
        printf("son distintos y el producto es %d",c);
        }
        getch();

}
```

CON FUNCIONES

****

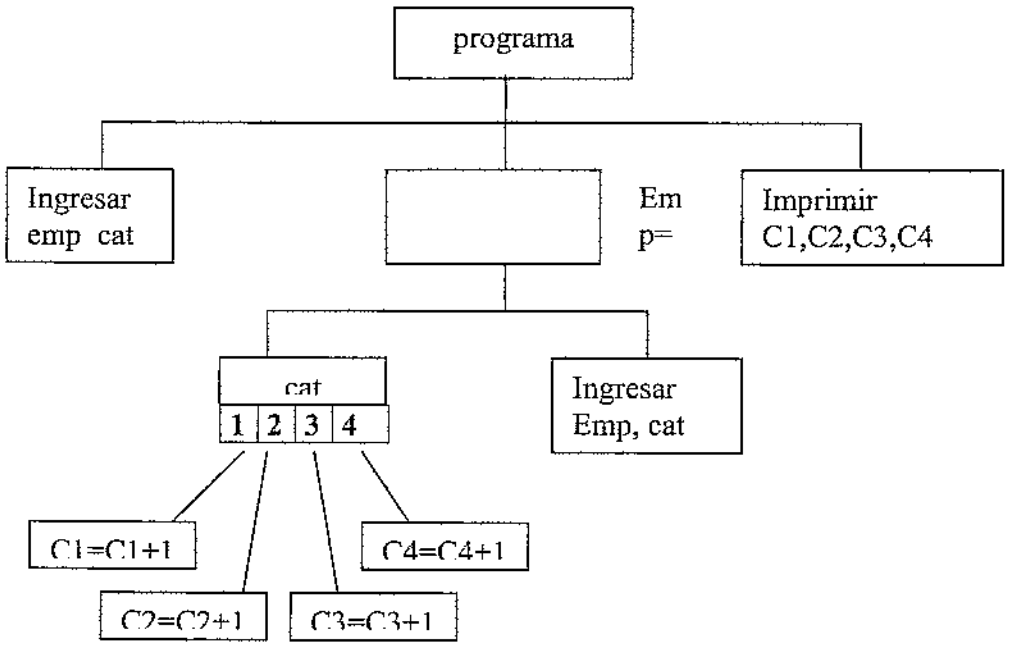
```
#include <stdio.h>
#include <conio.h>

////////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(int*, int*);
int suma(int, int);
int producto(int,int);
void FIN (void);

void main()
{
 int V1,V2,c;
 clrscr();
 carga_datos(&V1,&V2);     // LLAMADA A FUNCION
```

```
if(V1==V2)
        printf("son iguales y la suma es %d ",suma(V1,V2) );
else
        printf("son distintos y el producto es %d : ",producto(V1,V2) );
FIN();
}

//////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(int *x,int *y)
{
 printf("ingrese el primer valor : ");
 scanf("%d", x);
 printf("ingrese el segundo valor : ");
 scanf("%d", y);
}

////////////////////////////////////////////
int suma(int x, int y)
{
 return (x+y);
}

////////////////////////////////////////////
int producto(int x, int y)
{
 return (x*y);
}

////////////////////////////////////////////
void FIN(void)
{
  printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
  getch();
}
```

3. Condicional case o switch

Este tipo de condicional sólo se da cuando una variable puede tomar varios valores y por cada una de esos valores tomar distintas alternativas de acción.

Ingresar el nro. de empleado y categoría a la que pertenece (son 4),calcule cuantos empleados hay en cada una de ellas. Los datos finalizan con emp = 0



En seudo código seria

```
Comienzo
        Ingresar "ingrese empleado y categoría"
        Ingresar emp, cat
        Hacer hasta emp = 0
                Seleccionar caso cat
                        Caso 1:  C1=C1+1
                        Caso 2:  C2=C2+1
                        Caso 3:  C3=C3+1
                        Caso 4:  C4=C4+1
                Fin selección
                Ingresar "ingrese empleado y categoría"
                Ingresar emp, cat
        Repetir
        Imprimir "la cantidad de empleados de la cat 1 es C1"
        Imprimir "la cantidad de empleados de la cat 2 es C2"
        Imprimir "la cantidad de empleados de la cat 3 es C3"
        Imprimir "la cantidad de empleados de la cat 4 es C4"
Fin
```

En lenguaje *C* seria

```
#include<stdio.h>
#include <conio.h>

int emp,cat,c1,c2,c3,c4;


void main (void)
{
clrscr();
printf("ingresar el nro de empleado   ");
scanf("%d",&emp);
while (emp!=0)
{
        printf("ingrese la categoría  ");
        scanf("%d",&cat);
        switch(cat)
                {
                case 1:  c1=c1+1;
                break;
                case 2:  c2=c2+1;
                break;
                case 3:  c3=c3+1;
                break;
                case 4:  c4=c4+1;
                break;
                default: printf("error de categoría");
                }
        printf("ingresar el nro de empleado   ");
        scanf("%d",&emp);
}
printf("el total de empleados de la categoría 1 es %d \n ",c1);
printf("el total de empleados de la categoría 2 es %d \n",c2);
printf("el total de empleados de la categoría 3 es %d \n",c3);
printf("el total de empleados de la categoría 4 es %d \n",c4);
getch();
}
```

****

CON FUNCIONES

****

```
#include<stdio.h>
#include <conio.h>

////////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_empleado(int*);
void carga_categoria(int*);
void seleccion(int, int*, int*,int*,int*);
void informe(int, int, int, int );
void FIN (void);
void main (void)
{
 int emp,cat,c1,c2,c3,c4;
 c1=c2=c3=c4=0;
 clrscr();
 carga_empleado(&emp);
 while (emp)   // es igual a -------> while(emp !=0)
 {
   carga_categoria(&cat);          // LLAMADA A FUNCIONES
   seleccion(cat, &c1, &c2, &c3,&c4);
   carga_empleado(&emp);
 }
 informe(c1,c2,c3,c4);
 FIN();
}

////////////////////// DEFINICIONES DE FUNCIONES

void carga_empleado(int *x)
{
 printf("Ingresar el nro de empleado : ");
 scanf("%d",x);
}

////////////////////////////////////////////
void carga_categoria(int *x)
{
 printf("Ingresar la categoria del empleado : ");
 scanf("%d",x);
}
////////////////////////////////////////////
void seleccion(int c, int *x1, int *x2,int *x3,int *x4)
{
 switch(c)
  {
   case 1:  (*x1)++;   // es lo mismo que -----> ++*x1;
```

```
            break;
   case 2:  (*x2)++;
            break;
   case 3:  (*x3)++;
            break;
   case 4:  (*x4)++;
            break;
   default: printf("\nERROR EN LA CATEGORIA INGRESADA \n");
   }
}

////////////////////////////////////////////
void informe(int x1, int x2, int x3, int x4)
{
 printf("\nEl total de empleados de la categoría 1 es : %d ",x1);
 printf("\nEl total de empleados de la categoría 2 es : %d ",x2);
 printf("\nEl total de empleados de la categoría 3 es : %d ",x3);
 printf("\nEl total de empleados de la categoría 4 es : %d ",x4);
}

////////////////////////////////////////////
void FIN(void)
{
 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}
```

EJERCICIOS DE OPERACIONES CONDICIONALES

1. Ingresar dos valores, indicar si son iguales

2. Ingresar un valor indicar si es positivo, negativo o cero

3. Ingresar dos valores y realizar el producto, si el 1ro es mayor al 2do, si son iguales solo indicarlo

4. Ingresar dos valores y realizar la resta del mayor menos el menor

5. Ingresar los tres lados de un triángulo e indicar que tipo de triángulo es

6. Ingresar tres valores, sumarlos, calcular el promedio e indicar cúal de estos valores es mayor al promedio

7. Ingresar cuatro valores, sumar el 1ro y el 2do, el 3ro y el 4to, indicar cúal de esta sumas es mayor
8. Ingresar la edad y la altura de dos personas, indicar la estatura del de mayor edad
9. Ingresar el valor de la hora y el tiempo trabajado por un empleado, calcular su sueldo si se sabe que recibe un premio de $ 100 si trabajo más de 50 hs y si trabajo más de 150 hs le dan $ 100 adicionales.
10. Ingresar tres valores correspondientes al día, mes y año de una fecha, indicar si es válida, considerar los años bisiestos (existe una funcion que devuelve "B" en caso de bisiesto y "N" si no lo es)
11. Ingresar el sueldo, categoría y antigüedad de un empleado, calcular el sueldo final si se le da $ 50 por cada año trabajado a cada empleado de la categoria 1.
12. Sobre los datos del ejercico anterior imprimir los sueldos de los empleados con más de 5 años de antigüedad
13. Ingresar las horas trabajadas por un empleado y su categoría, calcular su sueldo si se sabe que los de la categoría 1 cobran $50, la 2 cobra $ 70 y la 3 cobra $ 80.

Iterativas

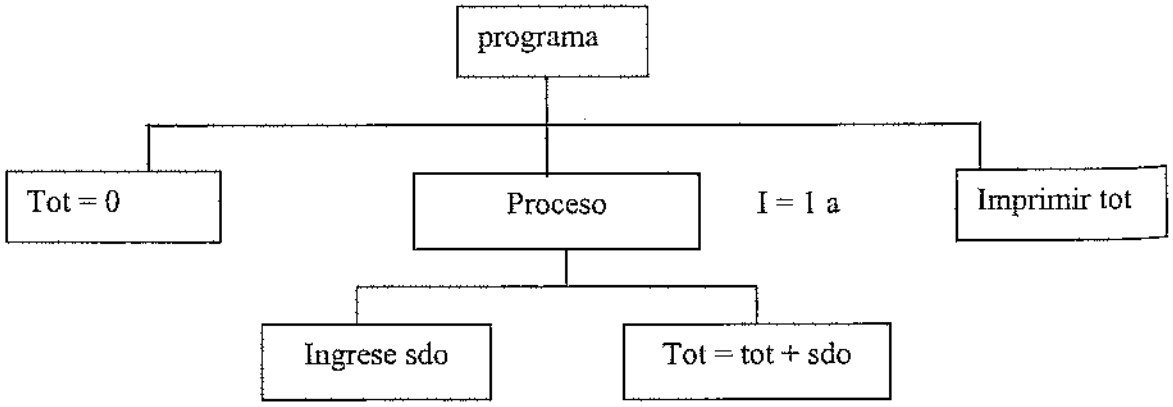
La condición que se desea que se cumpla, se escribe a la derecha del rectángulo

Pueden darse dos casos de iteración:

- **1.ciclos repetitivos exactos**

Dados los sueldos de N empleados, determinar el total a pagar

Tipo de ciclo = exacto N **datos** = N, sdo **resultado:** tot = tot + sdo



En seudo código sería:

```
Comienzo
Tot =0
Para I = 1 a 10
        Ingresar "ingrese el sdo del empleado, I"
        Ingresar sdo
        Tot = tot + sdo
Próximo
Imprimir "el total de sueldos es tot"
Fin
```

En lenguaje *C* sería

```
#include <stdio.h>
#include <conio.h>

void main()
{
int i;
float sdo,tot;
tot=0;
clrscr();
for(i=1;i<=10;i++)
{
printf("ingrese el sdo del empleado %d \n",i);
scanf("%f",&sdo);
tot =tot+sdo;
}
printf("el total de sueldos es %f",tot);
getch();
}
```

CON FUNCIONES

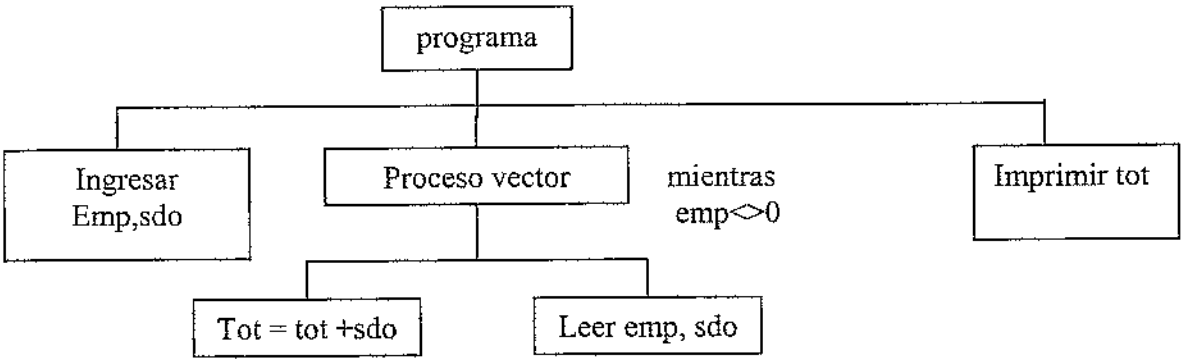
```
//*****************************************************************************
#include <stdio.h>
#include <conio.h>

//////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
float carga_datos(void);
float suma(float, float *);
void informe(float);
void FIN (void);
```

```
void main()
{
 float tot;
 clrscr();                // LLAMADAS A FUNCIONES
 tot = carga_datos();      // es lo mismo  utilizar  una funcion que
 informe(tot);            // llame a otra ------>informe(carga_datos());
 FIN();
}

//////////////////// DEFINICIONES DE FUNCIONES

float carga_datos(void)
{
  int i;
  float sdo, tot =0;
  for(i=1;i<=10;i++)
    {
     printf("\nIngrese el sueldo  del empleado %d : ",i);
     scanf("%f",&sdo);
     suma(sdo,&tot);
    }
  return tot;
}

////////////////////////////////////////////
float suma(float x, float *y)
{ return (*y+=x);  // es lo mismo que -------> *y = *y +x;
}

////////////////////////////////////////////
void informe(float x)
{
 printf("El total de sueldos es  : %f",x);
}

////////////////////////////////////////////
void FIN(void)
{
  printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
  getch();
}
```

- **2.Ciclos repetitivos inexactos**

Ingresar los sueldos de los empleados de una empresa hasta que el empleado sea igual a 0, calcular el total de sueldos a pagar

Tipo de ciclo = inexacto "hasta" **datos** = Emp **resultado**: tot = tot + sdo



En seudo código seria:

```
Comienzo
        Ingresar "ingrese el empleado y su sueldo"
        Ingresar emp,sdo
        Hacer mientras  emp <> 0
                Tot = tot + sdo
                Ingresar "ingrese el empleado y su sueldo"
                Ingresar emp,sdo
        Repetir
        Imprimir "el monto total a pagar es tot"
fin
```

En lenguaje *C* sería

```
#include <stdio.h>
#include <conio.h>

void main()
{
int emp;
sloat  sdo,tot;
clrscr();
tot=0;
printf("ingrese el nro de empleado ");
scanf("%d",&emp);
```

```
while(emp!=0)
{
printf("ingrese el sueldo ");
scanf("%f",&sdo);

tot =tot+sdo;
printf("ingrese el nro de empleado ");
scanf("%d",&emp);

}
printf("el total de sueldos es %f",tot);
getch();
}


//*********************************************************************
 CON FUNCIONES
//*********************************************************************


#include <stdio.h>
#include <conio.h>

//////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
float carga_datos(void);
float suma(float, float *);
void informe(float);
void FIN (void);
void main()
{
 clrscr();              // LLAMADAS A FUNCIONES
 informe(carga_datos());
 FIN();
}

//////////////////// DEFINICIONES DE FUNCIONES

float carga_datos(void)
{
  int emp;
  float sdo, tot =0;

  printf("\nIngrese el nro de empleado : ");
  scanf("%d",&emp);
```

```
 while(emp)
 {
  printf("\nIngrese el sueldo : ");
  scanf("%f",&sdo);
  suma(sdo,&tot);
  printf("\nIngrese el nro de empleado :");
  scanf("%d",&emp);
 }
 return tot;
}

////////////////////////////////////////////
float suma(float x, float *y)
{ return (*y+=x);  // es lo mismo que ------> *y = *y +x;
}

////////////////////////////////////////////
void informe(float x)
{
 printf("El total de sueldos es  : %8.2f",x);
}

////////////////////////////////////////////
void FIN(void)
{
 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}
```

EJERCICIOS SOBRE CICLOS, CONTADORES y ACUMULADORES

1. Ingresar 25 números, calcular su promedio

2. Ingresar 20 notas y nombres de alumnos, indicar los aplazados (menos de 4) y el nombre a quien pertenece esa nota

3. Ingresar N sueldos e indicar su suma y su promedio

4. Ingresar facturas hasta nro de factura = 0, sumar sus importes, indicar el total gastado y cúales y cuantas superan los $1000.

5. Sobre el ejercicio anterior indicar cuántas superan los $ 10000.-

6. Sobre el ejercicio anterior indicar cuántas estan entre $ 400 y $ 700 inclusive.

7. Ingresar 10 valores, indicar cuántos son positivos, cuántos negativos y cuántos ceros

8. Ingresar valores hasta uno = 0, indicar la cantidad de números ingresados y su promedio

9. Ingresar nombres y notas de alumnos teniendo en cuenta que la carga finaliza con nota = 11, calcular el promedio, imprimir los aprobados, cuántos estan entre 4 y 6..

10. Ingresar la patente y monto de la multa de 50 autos, indicar cuántos montos superan los $ 40 y del total cobrado que porcentaje representa la suma de estos últimos

11. Ingresar N valores y calcular promedio de positivos, de negativos y cantidad de ceros

12. Ingresar los datos de facturación de una empresa.
    Número de factura
    Número de artículo
    Cantidad vendida
    Precio unitario

    Los datos finalizan con numero de factura = 0, cada factura sólo tiene un número de artículo, existen tres artículos

    Se desea saber :
    Valor de cada factura
    Facturación total
    Cuánto se vendio del artículo 1 en cantidad
    Cuántas facturas mayores de $ 3000 se hicieron
    Qué porcentaje representa el monto vendido por cada artículo sobre el total

EJERCICIOS SOBRE MAXIMOS y MINIMOS

1. Ingresar N temperaturas e indicar la máxima y mínima
2. Ingresar temperaturas hasta una temperatura igual a 1000, indicar la mayor y menor
3. Ingresar los sueldos y nombres de 30 empleados, indicar el sueldo mayor y a quién pertenece
4. Ingresar las edades y estaturas de los alumnos, calcular la edad promedio, la edad mayor y la estatura menor, los datos finalizan con edad = 0
5. En una carrera de autos se ingresan el número de auto y su tiempo, indicar cuál ganó y cúal fue el último
6. Ingresar el precio de N artículos, indicar el más caro, el más barato, el precio promedio y la suma de todos los precios

EJERCICIOS COMBINADOS

1. En una empresa los empleados cobran un sueldo según la categoria, son 50 empleados y 3 categorías

   Categoría 1 = $ 1500
   Categoría 2 = $ 2000
   Categoría 3 = $ 2500

   Al sueldo se le suman $ 100 por cada año trabajado.
   Si se ingresa el nombre, categoría y antigüedad de cada empleado, calcular

   A. Cuántos empleados hay por categoría
   B. Total de sueldos pagados por categoría
   C. Sueldo promedio general
   D. Sueldo máximo y a quién pertenece
   E. Qué porcentuel sobre el total de sueldos representa cada total de sueldos de las categorías

2. Una empresa conoce el nombre, sueldo y categoría de sus empleados, son 4 categorías, y la cantidad de empleados es variable N.
   Se desea saber

A. Cantidad de empleados por categoría

B. Cantidad de empleados que cobran mas de $ 2000
C. Cantidad de empleados de la categoría 1 con sueldo mayor a $ 1000
D. Sueldo máximo y a qué empleado pertenece
E. Sueldo mínimo y a que empleado pertenece
F. Categoría con más empleados
G. Porcentual en cantidad de empleados de cada categoría sobre el total de la empresa

3. Una empresa desea procesar las ventas que efectúa conociendo los siguientes datos:
   número de factura
   código de articulo
   cantidad vendida
   precio unitario

   en cada factura sólo se vende un tipo de artículo, los artículos son 6 y los datos finalizan con número de factura = 0
   Se pide :

   A. cantidad de facturas emitidas
   B. monto de cada factura
   C. suma de las facturas (caja diaria)
   D. cantidad total de artículos vendidos
   E. cantidad vendida de cada artículo
   F. cantidad de facturas emitidas para cada artículo
   G. promedio de importe de las facturas
   H. porcentual que representa el monto total de cada artículo sobre el total

4. Una compañía aérea vende boletos en 3 aeropuertos mediante 6 empleados, cada vez que realiza una venta tiene los siguientes datos:

   número de aeropuerto
   número de empleado
   valor del pasaje
   cantidad de pasajes

   los datos finalizan con número de empleado = 99, Se desea saber

   A. cantidad de pasajes vendidos por aeropuerto
   B. cantidad de pasajes vendidos por cada empleado
   C. monto total vendido por empleado
   D. boleto de mayor valor
   E. porcentual que representa lo vendido por cada aeropuerto en dinero sobre el total
   F. cantidad de ventas que excedieron los $ 1000
   G. si cada boleto ya tiene incorporado el 10,5% de iva sobre el valor del boleto, cuánto hay que pagarle al fisco
   H. si sobre el total hay que pagar el 3% de ingresos brutos, cuál es ese monto

5. Un restaurant tiene 6 mozos y 12 mesas, cada vez que cobran una mesa se anotan los siguientes datos, que terminan con número de factura = 0

número de factura
número de mozo
número de mesa
cantidad de personas que comieron
monto de la cuenta

Se desea saber lo siguiente

A. cantidad de personas atendidas por cada mozo
B. cantidad de facturas realizadas
C. caja diaria realizada
D. monto facturado por cada mesa en total
E. porcentual facturado por cada mozo sobre el total
F. si a cada mozo se le da el 5% de su venta cuánto le corresponde a cada uno
G. cuál mozo atendió más personas en total
H. cuántas facturas superaron $ 45
I. mesa a la que se facturó más veces, o sea que más recambio de clientes tuvo

6. Un supermercado realiza un estudio sobre sus ventas, para ello cuenta con seis secciones y 10 cajas. cada vez que realiza una cuenta tiene los siguientes datos

número de caja
número de seccion
importe de la venta
tiempo en caja
cantidad de productos comprados
nro de ticket

los datos finalizan con número de caja = 0, si la compra supera los $50, se le realiza un 2% de descuento, Se desea saber

A. cantidad de tickets emitidos
B. cantidad de tickets emitidos por caja
C. ticket de mayor tiempo en caja
D. monto recaudado por sección
E. que porcentaje representa el monto recaudado por caja sobre el total
F. cantidad de comprobantes con monto mayor a $ 100
G. monto total descontado
H. porcentual que representa el monto total descontado sobre el total vendido
I. si cada cajero gana el 20 % de su recaudación cuánto cobra cada uno

7. Una cartelera de cine tiene 5 sucursales y vende a un precio de 7 pesos las entradas para sus salas en los 3 turnos, si compran mas de 8 asientos en un solo ticket se le hará un 10% de descuento.

   En este momento se estan exhibiendo las siguientes películas

   1. Belleza americana
   2. El informante
   3. La hija del general
   4. Al filo de la muerte

   Cada vez que compran, le dan un ticket donde figuran

   - Número de sucursal
   - Número de película
   - Cantidad de asientos comprados
   - Turno

   Se desea saber

   a. Cantidad de asientos vendidos en cada turno
   b. Pelicula de mayor recaudación
   c. Sucursal que vendió menos en total en pesos
   d. Si cada sucursal recibe un 20% de lo recaudado, cuánto ganó c/u
   e. Porcentual que representa lo recaudado por pelicula sobre el total
   f. Ticket de menor valor
   g. Cantidad de asientos vendidos para la pelicula 2 en la sucursal 3 y turno Tarde
   h. Cuál fue el promedio de asientos solicitados por cada sucursal

8. Un noticiero de TV tiene 10 comentaristas para cubrir 5 móviles, al término de cada transmisión presentan un informe con los siguientes datos

   código de comentarista
   número de movil
   categoría
   cantidad de horas
   localidad

   las categorias son 3

   1. policiales
   2. políticas
   3. economía

las localidades son 4

1. Capital Federal
2. Gran Bs. As.
3. interior
4. exterior

Se desea saber

a. cantidad de hs trabajadas por cada comentarista
b. localidad desde donde se transmitió más veces
c. porcentual que representa la totalidad en hs de cada categoría sobre el total
d. promedio de hs trabajadas por cada comentarista
e. cantidad de veces que se transmitió política desde el exterior
f. que categoría fue la menos transmitida en total
g. que transmisión fue la de mayor duracion en hs
h. el comentarista 1 en que localidad trabajó más

9. Una empresa de micros vende boletos a 3 destinos

Mar del Plata
Necochea
Bahía Blanca

tiene 2 clases

Turista	$ 25.-
Pullman	$ 40.-

Cada vez que realiza una venta representa un asiento y se tiene la siguiente información:

a. número de ticket
b. número de destino
c. número de clase

Se desea saber

1. cantidad total de boletos vendidos
2. promedio del valor de boleto vendido en dinero
3. porcentual que representa la cantidad vendida en cada clase sobre el total
4. cuántos boletos se vendieron a necochea en pullman
5. cuál destino es el más solicitado en total
6. cuál clase en total es la menos solicitada

Tipos de datos estructurados

Los dos tipos de datos estructurados más importantes son el array (vectores y matrices) y los registros.

En un arreglo las componentes son del mismo tipo y se accede a ellas mediante un número correlativo llamado subíndice.

En los registros en cambio los componentes pueden ser de diversos tipos, especificadas por distintos nombres, y a su conjunto se lo puede acceder mediante distintas técnicas que veremos más adelante.

Vectores

Los vectores son estructuras de memoria que poseen un tamaño conocido de ocurrencias y en general almacenan datos de un mismo tipo.

Dado un vector de nombre vect de 10 ocurrencias que almacena valores numéricos

12	45	56	78	15	23	56	15	1000	41
1	2	3	4	5	6	7	8	9	10

La forma de escribir un vector sería

vect (subíndice)

donde el subíndice sería normalmente un numero entero, representado por un dato, una variable o una constante.

El valor del subíndice puede ser desde 0, 1 o un valor arbitrario, eso dependerá del lenguaje a utilizar.

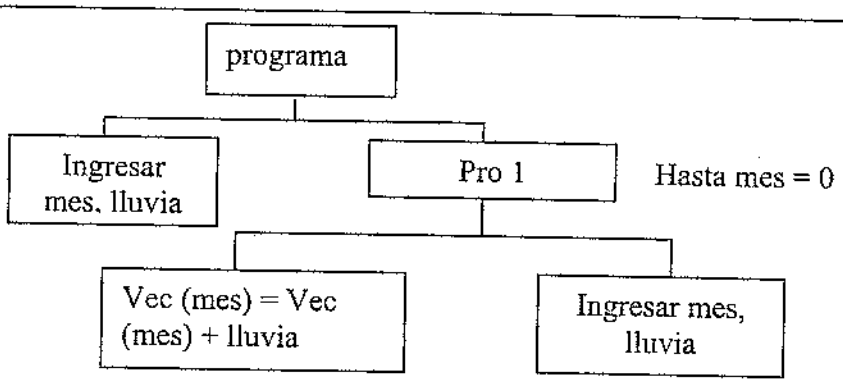
En el ejemplo anterior tenemos que
Si la posición o valor del subíndice es k 6
el valor guardado en el vector en esa posición es 23

Carga directa de un vector

Sea un lote de datos con los siguientes campos

mes tipo numérico de 1 a 12
lluvia-caida tipo numérico

desea cargar un vector con dicha información.



Como vemos en este ejemplo se utilizó como subíndice de carga un dato del lote que era numérico, entero y conocido

En seudo código seria

```
Comienzo
Defino vec (12 )
Ingresar "ingrese el mes y la cantidad de lluvia caida"
Ingresar mes,lluvia
Hacer hasta mes = 0
        Vec(mes)=vec(mes)+lluvia
        Ingresar "ingrese el mes y la cantidad de lluvia caida"
        Ingresar mes,lluvia
Repetir
Fin
```

En lenguaje *C* seria:

```
//*****************************************************************
 SIN FUNCIONES   ( ACCESO DIRECTO SEGUN LA VARIABLE MES )
//*****************************************************************
#include <stdio.h>
#include <conio.h>
void main(void)
{
 int vec[12]={0};              // se inicializa todo el vector en cero
 int mes,lluvia,i;
 clrscr();

 do{                                  // conviene validar el dato
  printf("\nIngresar mes : ");        // cuando se ingresa directamente a
  scanf("%d",&mes);                  // una posicion de memoria, para
  }
while(mes < 0 ||  mes > 12);        // NO invadir memoria NO asignada.
```

```
while(mes)
{
printf("\nIngresar cantidad de lluvia caida : ");
scanf("%d",&lluvia);

vec[mes-1]+=lluvia;  // es lo mismo que vec[mes-1]=vec[mes-1]+lluvia;

do{
  printf("\nIngresar mes : ");
  scanf("%d",&mes);
  }while(mes < 0 ||  mes > 12);
}
clrscr();

printf("\n TOTAL DE LLUVIA CAIDA SEGUN EL MES \n\n\n");
printf("\n\n     MES           LLUVIA CAIDA (en ml)");
for( i=0;i<12;i++)
 printf( "\n     %2d            %6d ",i+1,vec[i]);

printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
getch();
}
```

CON FUNCIONES (ACCESO DIRECTO SEGUN LA VARIABLE MES)

```
//*************************************************************************
#include <stdio.h>
#include <conio.h>

////////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(int []);
void suma (int [], int, int );
void informe(int[]);
void FIN(void);
void main(void)
{
int vec[12]={0};              // se inicializa todo el vector en cero
clrscr();
carga_datos(vec);          // LLAMADAS A FUNCIONES
clrscr();
informe(vec);
FIN();
}

////////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(int x[])
{
```

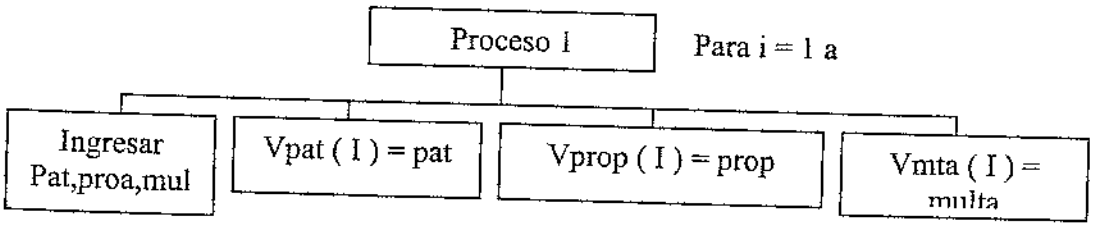
```
int mes, lluvia;

do{
  printf("\nIngresar mes : ");
  scanf("%d",&mes);
  }while(mes < 0 ||  mes > 12);

while(mes)
{
 printf("\nIngresar cantidad de lluvia caida : ");
 scanf("%d",&lluvia);

 suma(x,mes,lluvia);

 do{
  printf("\nIngresar mes : ");
  scanf("%d",&mes);
  }while(mes < 0 ||  mes > 12);

 }
}

////////////////////////////////////////////
void suma(int x[], int y, int z)
{
 x[y-1]+=z;
}
////////////////////////////////////////////
void informe (int x[])
{ int i;
 printf("\n TOTAL DE LLUVIA CAIDA SEGUN EL MES \n\n\n");
 printf("\n\n     MES          LLUVIA CAIDA (en ml)");
 for( i=0;i<12;i++)
  printf( "\n     %2d          %6d ",i+1,x[i]);
}

////////////////////////////////////////////
void FIN(void)
{
 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}
```

Carga indirecta de un vector

Sea un archivo secuencial con los siguientes campos

Patente	alfanumérico
Propietario	alfanumérico
Multa	numérico

Y se sabe que se confeccionaron 100 multas en el día, si deseamos cargar este archivo en vectores, vemos que no podemos utilizar ninguno de sus campos como subíndice, por lo que deberemos hacer lo siguiente



En seudo código seria

```
Comienzo
Defino Vpat (100), Vprop (100), Vmta (100)
Para i = 1 a 100
        Ingresar "ingrrese la patente"
        Ingresar " ingrese el propietario"
        Ingresar " ingresar la multa"
        Vpat(I)=pat
        Vprop(I)=proa
        Vmta(I)=multa
próximo
fin
```

En lenguaje *C* seria:

```
//*****************************************************************************
 SIN FUNCIONES    ( ACCESO SECUENCIAL )
//*****************************************************************************
#include <stdio.h>
#include <conio.h>
#define N 100

void main(void)
{
char Vpat[N][7], Vprop[N][31];
float Vmta[N];
int i;
clrscr();
for(i=0;i<N;i++)
```

```
{
 printf("\nIngresar Patente : ");
 scanf("%s",&Vpat[i]);
 fflush(stdin);              // se limpia el buffer por el <ENTER>
 printf("\nIngresar Propietario : ");
 scanf("%[^\n]",&Vprop[i]);      // cuando se ejecuta se leer una cadena de

 printf("\nIngresar la Multa :"); // caracteres de longitud no determinada,
 scanf("%f",&Vmta[i]);          // no mas de 79 caracteres
}
clrscr();
printf("\n\n LOS DATOS INGRESADOS SON ");
printf ( "\n    PATENTE    PROPIETARIO                MULTA   \n");

 for(i=0;i<N;i++)
  printf("\n     %-6s    %-30s  %8.2f", Vpat[i],Vprop[i],Vmta[i]);          // no mas de 79
caracteres

 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}


//*************************************************************************
 CON FUNCIONES    ( ACCESO SECUENCIAL )
//*************************************************************************

#include <stdio.h>
#include <conio.h>
#define N 100

////////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(char [][7],char[][31],float[]);
void informe (char [][7],char[][31],float[]);
void FIN (void);

 void main(void)
 {
  char Vpat[N][7], Vprop[N][31];
  float Vmta[N];
  clrscr();
  carga_datos(Vpat,Vprop,Vmta);
  clrscr();
  informe(Vpat,Vprop,Vmta);
  FIN();
```

```
}

//////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(char x[][7],char y[][31],float z[])
{
 int i;
 for(i=0;i<N;i++)
 {
  printf("\nIngresar Patente : ");
  scanf("%s",&x[i]);
  fflush(stdin);
  printf("\nIngresar Propietario : ");
  scanf("%[^\n]",&y[i]);
  printf("\nIngresar la Multa :");
  scanf("%f",&z[i]);
 }
}

////////////////////////////////////////////
void informe (char x[][7],char y[][31],float z[])
{
 int i;
 printf("\n\n LOS DATOS INGRESADOS SON ");
 printf ( "\n    PATENTE    PROPIETARIO                MULTA  \n");

 for(i=0;i<N;i++)
  printf("\n     %-6s     %-30s  %8.2f", x[i],y[i],z[i]);
}

////////////////////////////////////////////
void FIN(void)
{
  printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
  getch();
}
```

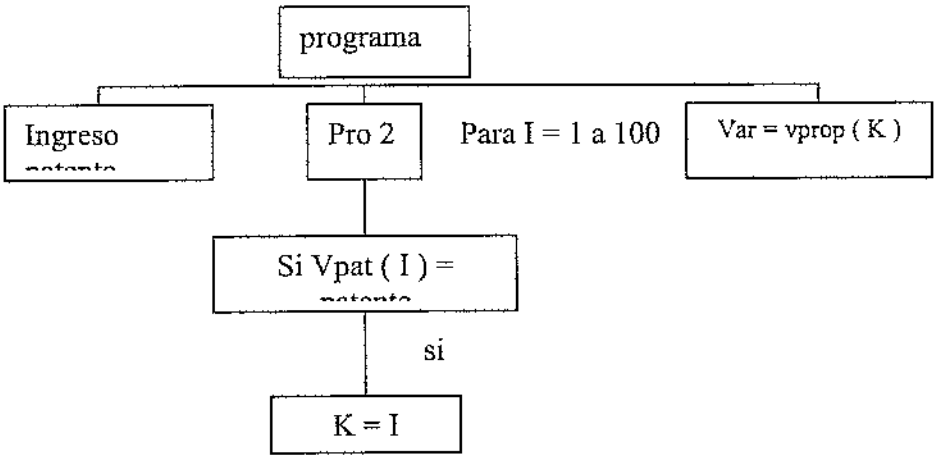
Búsqueda en vectores

Para extraer un dato de un vector necesitaremos conocer esencialmente la posición que deseamos recuperar

Para ello podemos utilizar un dato que nos suministran o un ciclo exacto hasta encontrar el valor deseado o una constante.

Var = Vec (mesaux)	Var = Vec (4)

Si me piden del ejemplo anterior decir quien es el propietario de una determinada patente ya cargada, deberemos hacer



En seudo código sería

```
Comienzo
Ingresar "ingrese la patente"
Ingresar patente
Para I= 1 a 100
        Si  Vpat(I)=patente
                K=I
        Fin si
Próximo
Imprimir "el propietario de la patente es Vprop(I)"
Fin
SIN FUNCIONES     ( BUSQUEDA SECUENCIAL )
//*****************************************************************************
#include <stdio.h>
#include <conio.h>
#include <string.h>
#define N 100

void main(void)
{
 char Vpat[N][7], Vprop[N][31],Consul_pat[7];
 int i, encontrado;
 clrscr();
 // CARGA DE DATOS PATENTE - PROPIETARIO
 for(i=0;i<N;i++)
```

```
{
 printf("\nIngresar Patente : ");
 scanf("%s",&Vpat[i]);
 fflush(stdin);
 printf("\nIngresar Propietario : ");
 scanf("%[^\n]",&Vprop[i]);
 }
// BUSQUEDA DEL PROPIETARIO SEGUN LA PATENTE INGRESADA

printf("\nIngrese la patente cuyo propietario quiere consultar : ");
scanf("%s",Consul_pat);

encontrado =-1;
        // se usa la variable encontrado para determinar si:
        //              encontrado  =-1 ------> No encontrado
        //              encontrado >= 0 ------> Si encontrado
for(i=0;i<N && encontrado == -1;i++)
 if(!(strcmpi(Vpat[i],Consul_pat)))
                // para comparar string se usa una funcion predefinida.
                // Esta funcion, si son iguales retorna un cero, en caso
                // contrario un valor distinto de cero.
    encontrado =i;

if (encontrado >= 0)
  printf("\n\nEl propietario de la patente %s es : %s",Consul_pat,Vprop[encontrado]);
else
  printf("\nNO SE ENCONTRO EL PROPIETARIO ");

printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
getch();
}


//*****************************************************************
 CON FUNCIONES    ( BUSQUEDA SECUENCIAL )
//*****************************************************************
#include <stdio.h>
#include <conio.h>
#include <string.h>
#define N 100

/////////////////////// DECLARACIONES O PROTOTIPOS DE FUNCIONES
void carga_datos(char [][7],char[][31]);
int busqueda(char[],char[][7]);
void informe(int, char[],char[][31]);
```

```
void FIN(void);

void main(void)
{
 char Vpat[N][7], Vprop[N][31],Consul_pat[7];
 int i, encontrado;
 clrscr();
 // CARGA DE DATOS PATENTE - PROPIETARIO
 carga_datos(Vpat,Vprop);


 // BUSQUEDA DEL PROPIETARIO SEGUN LA PATENTE INGRESADA
 printf("\nIngrese la patente cuyo propietario quiere consultar : ");
 scanf("%s",Consul_pat);

 encontrado =busqueda(Consul_pat,Vpat);
        // se usa la variable encontrado para determinar si:
        //              encontrado = -1 ------> No encontrado
        //              encontrado >= 0 ------> Si encontrado
        //                                      (valor de la posicion)
 informe(encontrado,Consul_pat,Vprop);

 FIN();

}
////////////////////// DEFINICIONES DE FUNCIONES

void carga_datos(char x[][7],char y[][31])
{
 int i;
 for(i=0;i<N;i++)
 {
  printf("\nIngresar Patente : ");
  scanf("%s",&x[i]);
  fflush(stdin);
  printf("\nIngresar Propietario : ");
  scanf("%[^\n]",&y[i]);
  }
}

//////////////////////////////////////////////
int busqueda(char x[],char y[][7])
{
 int i, encontrado =-1;
 for(i=0;i<N && encontrado < 0;i++)
```

```
 if(!(strcmpi(x,y[i])))
                // para comparar string se usa una funcion predefinida.
                // Esta funcion, si son iguales retorna un cero, en caso
                // contrario un valor distinto de cero.
     encontrado =i;
 return encontrado;
}

////////////////////////////////////////////
void informe(int x, char y[],char z[][31])
{ if (x >= 0)
   printf("\n\nEl propietario de la patente %s es : %s",y,z[x]);
 else
   printf("\nNO SE ENCONTRO EL PROPIETARIO ");
}

////////////////////////////////////////////
void FIN(void)
{
 printf("\n\n\n TIPEE UNA TECLA PARA FINALIZAR  " );
 getch();
}

////////////////////////////////////////////
```

EJERCICIOS DE VECTORES

1. Ingresar datos y cargar un vector de 50 elementos, calcular

   - la suma de todos los elementos
   - el producto de todos los elementos
   - mostrar del vector el elemento 50 al 1
   - imprimir las componentes de indice par
   - imprimir las componentes de indice impar

   Ingresar datos y cargar un vector de 30 elementos, imprimir

   a. cantidad de valores positivos
   b. cantidad de valores negativos
   c. cantidad de ceros
   d. promedio de los positivos
   e. promedio general

   Ingresar datos y cargar un vector de 10 elementos, imprimir

1. cantidad de elementos del vector cuyo valor sea igual a 1
2. suma de los elementos del vector
3. porcentual que representa cada elemento sobre el total

Ingresar 25 edades, y calcular

1. edad promedio
2. imprimir las edades mayores a 34 años
3. imprimir las edades menores a 21 años

Ingresar 10 sueldos y edades de una empresa y calcular

1. sueldo promedio
2. sueldo promedio de los empleados que tienen entre 23 y 40 años
3. edad promedio
4. cantidad de empleados mayores a 30 años y sueldo menor a $ 1000.
5. cantidad de empleados con edades menor a la edad promedio

Dadas las 40 notas y edades de los alúmnos de un colegio calcular

1. nota promedio
2. cantidad de alúmnos aplazados
3. cantidad de alúmnos promocionados
4. nota promedio de los alúmnos mayores a 15 años

Nota: todos los ejercicios de Ejercicios Combinados pueden resolverse con vectores

MAXIMOS Y MINIMOS DE UN VECTOR

Tener en cuenta que pueden existir máximos o mínimos múltiples

1. Ingresar los datos en un vector de 35 elementos y calcular su valor máximo

2. Ingresar el sueldo y nombre de 40 empleados, calcular

    1. sueldo máximo
    2. nombre del empleado que menos cobra
    3. promedio de sueldos
    4. diferencia entre el sueldo máximo y el promedio

3. En una competencia automovilística corren 25 autos numerados del 1 al 25, se ingresan los tiempos realizados por cada uno de ellos, calcular

    1. cuál fue el auto que llego primero
    2. cuál fue el último
    3. el primero en llegar fue el auto número 5

4. El mismo ejercicio anterior pero los números de los autos son aleatorios y no correlativos.

ORDENAMIENTO DE VECTORES

1. Ingresar los sueldos de 60 empleados, ordenarlos de mayor a menor

2. Ingresar los nombres y las notas promedios de 40 alúmnos e imprimirlos en forma ordenada ascendente por notas

3. Ingresar los datos del ejercicio de la carrera automovilística nro 4 e imprimirlos

    - ordenados por tiempo
    - ordenados por número de auto

4. Ingresar los precios, descripción y cantidad vendida de los 30 platos que tiene un restaurant, imprimirlos ordenados por precio.

EJERCICIOS COMBINADOS DE VECTORES

1- Una empresa comercializa 10 productos, se cargan los precios y códigos de producto, Calcular e imprimir

1. precio máximo y código de artículo al que pertenece
2. precio mínimo y código de artículo al que pertenece
3. cantidad de artículos con precio superior al promedio
4. cantidad de artículos con precio inferior a 10 pesos

2- Una empresa procesa 50 facturas con los siguientes datos:

- nro de factura
- nro de vendedor
- importe

Si existen 4 vendedores, calcular:

1. caja del día
2. porcentual que representa lo vendido por cada vendedor del total

3. valor promedio de las facturas

3- Una empresa comercializa 20 libros contenidos en 5 géneros.cada vez que realiza una venta tiene los siguientes datos

- nro de libro
- nro de género
- cantidad vendida
- precio unitario

Se desea saber:

- facturación total
- facturación por libro
- facturación por genero
  . precio promedio de factura
  . cantidad de facturas de más de $100.

4- Un banco realiza operaciones de depósito y extracción.
En cada operación se registra lo siguiente, tiene 10 sucursales

- nro de operación
- nro de sucursal
- tipo de transacción
- monto

Se desea saber

1. cuánto recaudó en concepto de depósitos cada sucursal
2. cuánto entregó cada sucursal en concepto de extracción
3. de qué sucursal se extrajo más que lo que se depositó

5- Una empresa textil maneja sus ventas por medio de corredores que cobran comisiones sobre el total de las mismas.
Tiene 4 artículos y son 10 corredores con las siguientes comisiones:

- corredor 1,2,3 = 5 %
- corredor 4,5,6 = 8 %
- corredor 7,8,9 = 10 %
- corredor 10 = 12 %

el precio de sus artículos son
- art 1 = $10

- art 2 = $15
- art 3 = $20
- art 4 = $12

cada vez que realiza una venta sabe

- nro de corredor
- nro de artículo
- cantidad
- descuento

Calcular e imprimir

a. importe total vendido por cada corredor
b. porcentual que representa sobre la venta de la empresa
c. comisión a cobrar por cada corredor
d. cantidad de ventas realizadas por cada corredor
e. promedio del valor vendido por cada corredor
f. nro de corredor que vendió más
g. imprimir ordenado por monto total vendido por cada corredor, el monto, la cantidad de facturas hechas y la comisión recibida

6- Una empresa de informática tiene 100 empleados que pertenece a una determinada categoría distribuidos en 3 departamentos

nro de categoría	**descripción**	**sueldo**
1	analista senior	2500
2	analista junior	2000
3	programador	1500
4	operador	800

los datos que se tienen son: el nombre de empleado, la categoría, y el departamento

Se desea saber:

a. sueldo total a pagar por departamento
b. sueldo total a pagar por categoría
c. categoría que más cobra en total
d. departamento que menos cobra en total
e. cantidad de empleados por categoría
f. sueldos totales a pagar por la empresa
g. categoría con menos empleados

h. imprimir ordenados por sueldos por categoría, los sueldos y la cantidad de empleados por categoría

7- Una empresa transportadora de caudales posee 8 camiones blindados para realizar sus viajes

Los valores transportados son 4 :

código del valor	**descripción**	**costo**
1	oro	300
2	billetes	250
3	piedras preciosas	200
4	documentos	150

Cada vez que realiza un viaje tiene la siguiente información que finaliza con cliente = 0

- código de cliente
- número de camión
- tipo de valor
- cantidad

Se desea saber:

a. cantidad de viajes realizados por camión
b. total transportado por camión del valor 1
c. recaudación de cada valor
d. recaudación de cada camión
e. nombre del valor más transportado en total
f. cliente que transportó menos en un viaje
g. porcentual que representa lo recaudado por cada camión sobre el total
h. imprimir ordenado por la cantidad de viajes realizados por camión en forma ascendente:
   - cantidad de viajes realizados por camión
   - recaudación por camión
   - número de camión

8- Una empresa de combustible tiene 20 estaciones de servicios, donde se expanden 3 tipos de combustible

tipo	descripción	precio
1	ecológica	$ 1.50
2	super	$ 1.30
3	común	$ 1.10

Al realizar una venta se tiene los siguientes datos, que finalizan con número de boleta = 90

- número de boleta
- número de estación
- tipo de combustible
- cantidad

Se desea saber:

a. cantidad de litros vendidos en cada estación
b. recaudacion de cada estación
c. cantidad de boletas emitidas por estación
d. total de litros vendidos por cada tipo de combustible
e. la estación 1 fue la que más recaudó?
f. número de boleta con mayor importe
g. imprimir las tres primeras preguntas ordenadas por recaudación por estación en forma ascendente

9- Una empresa de micros realiza viajes a 4 destinos y tiene 3 tipos de tarifa iguales para todos los destinos

código destino	descripción
	Salta
	Jujuy
	Tucumán
	Formosa

código de tarifa	descripción	valor
1	pullman	$ 85
2	primera	$ 70
3	turista	$ 50

cada vez que se vende un boleto se tienen los siguientes datos que finalizan con nro de ticket = 0

- número de ticket
- código de destino
- código de tarifa
- cantidad de boletos

si se compran más de 10 boletos se le hace 10 % de rebaja

Se desea saber:

1. cantidad de pasajes vendidos a cada destino
2. ticket de mayor valor
3. recaudación por destino
4. porcentaje que representa lo recaudado por destino sobre el total
5. cantidad de pasajes vendidos por clase
6. salta fue el destino más solicitado ?
7. si debe pagar el 21% de IVA, cuál es el monto a pagar
8. imprimir ordenado descendentemente por recaudación por destino:
   - nombre del destino
   - código del destino
   - recaudación por destino
   - cantidad de pasajes por destino

10- Una entidad de beneficiencia tiene autorizados 30 puestos de venta de alimentos en la vía pública, en ellas se vende:

código de mercadería	**descripción**	**precio**
pan	pancho	1.00
ham	hamburguesa	2.00
gas	gaseosa	0.75

cada vez que se vende se emite un ticket con los siguientes datos

- número de ticket
- número de puesto
- tipo de mercadería
- cantidad

los datos finalizan con un numero de ticket = 0

Se desea saber:

1. ticket de mayor valor
2. recaudación de cada puesto
3. porcentual que representa lo recaudado por cada mercadería sobre el total
4. mercadería más vendida
5. puesto menos rentable en dinero
6. si el 10% se destina a impuestos, cuánto debe pagar cada puesto
7. cantidad de tickets vendidos en cada puesto
8. imprimir ordenado por número de puesto:
    i. número de puesto
    ii. recaudación por puesto
    iii. cantidad de mercadería vendida por puesto

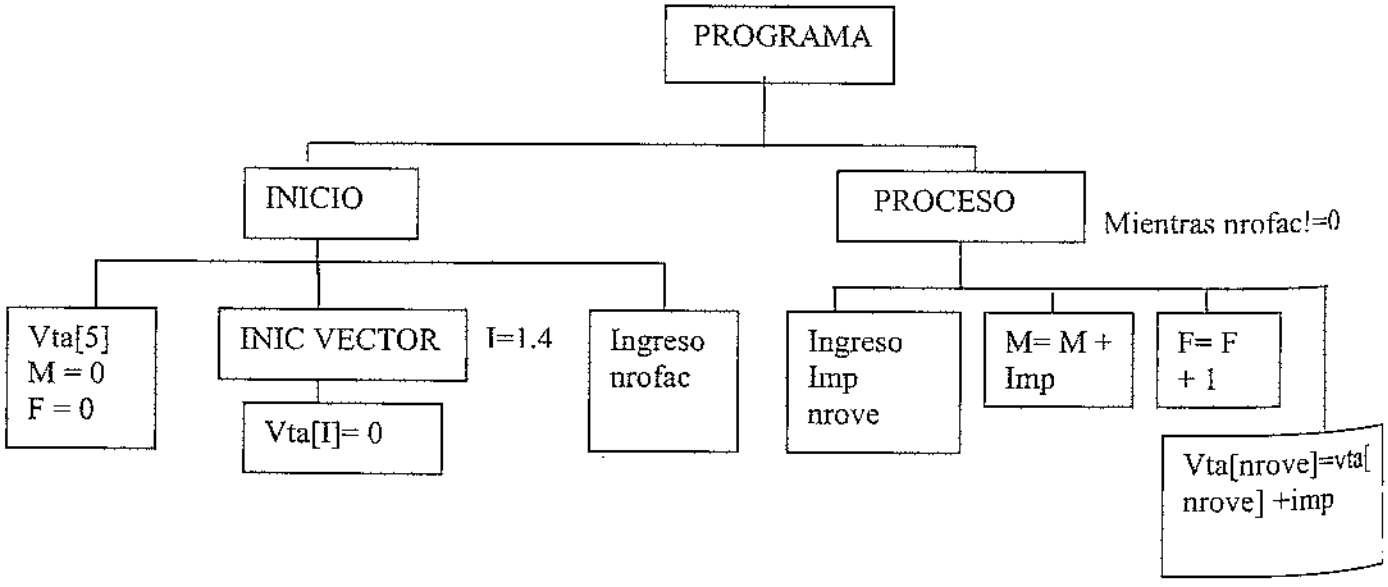
Ejercicio resuelto con vectores

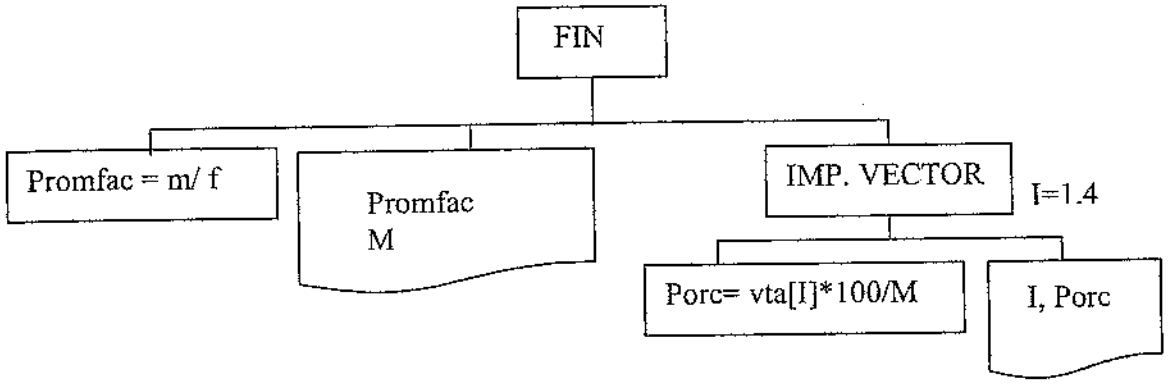
Una empresa procesa N facturas con los datos

- NRO FAC
- NRO VEND
- IMPORTE

Si existe 4 vendedores, calcular

1. Caja del día
2. Porcentual que representa lo vendido por cada vendedor del total.
3. Valor promedio de las facturas





Realizar la codificación correspondiente en lenguaje C.

Matrices

Las matrices son arrays de dos o más dimensiones según soporte el lenguaje a utilizar. Para nuestro estudio utilizaremos matrices bidimensionales

Mantienen básicamente las mismas características en lo que concierne al tipo de subíndices a utilizar, su representación y su contenido.

En lugar de manejar un solo subíndice manejaremos dos que nos representarán las filas y columnas de una matriz

45	65	12	58	47	69
15	48	59	23	52	42
78	84	9	120	451	784
65	85	53	42	73	95

En este ejemplo tenemos que la matriz llamada MATSDO tiene las siguientes características

Tiene 4 filas y 6 columnas MATSDO (4,6)

Si la fila vale 2 y la columna 5 el valor de MATSDO (2, 5) es 52

Sus formas de carga y de posible búsqueda de un dato en ella es similar a la de los arrays o vectores unidimensionales, pudiendo combinarse en sus subíndices una constante y un dato, una constante y una variable, dos datos, etc.

Supongamos que me dan los siguientes lotes de datos

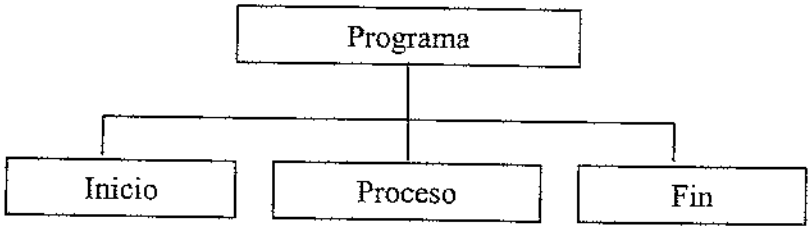
Lote depósito
codDep alfanumérico
Nombre alfanumérico

Lote artículo
Codart alfanumérico
Descripción alfanumérico

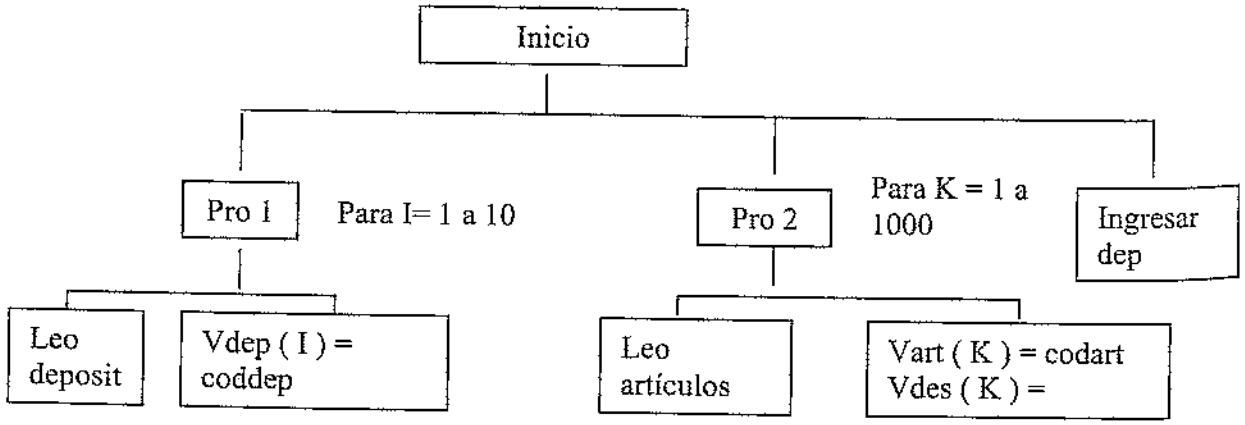
Me dicen que son 1000 artículos y 10 depósitos

Luego me dan el lote de datos del stock
Deposito alfanumérico
Articulo alfanumérico
Stock numérico

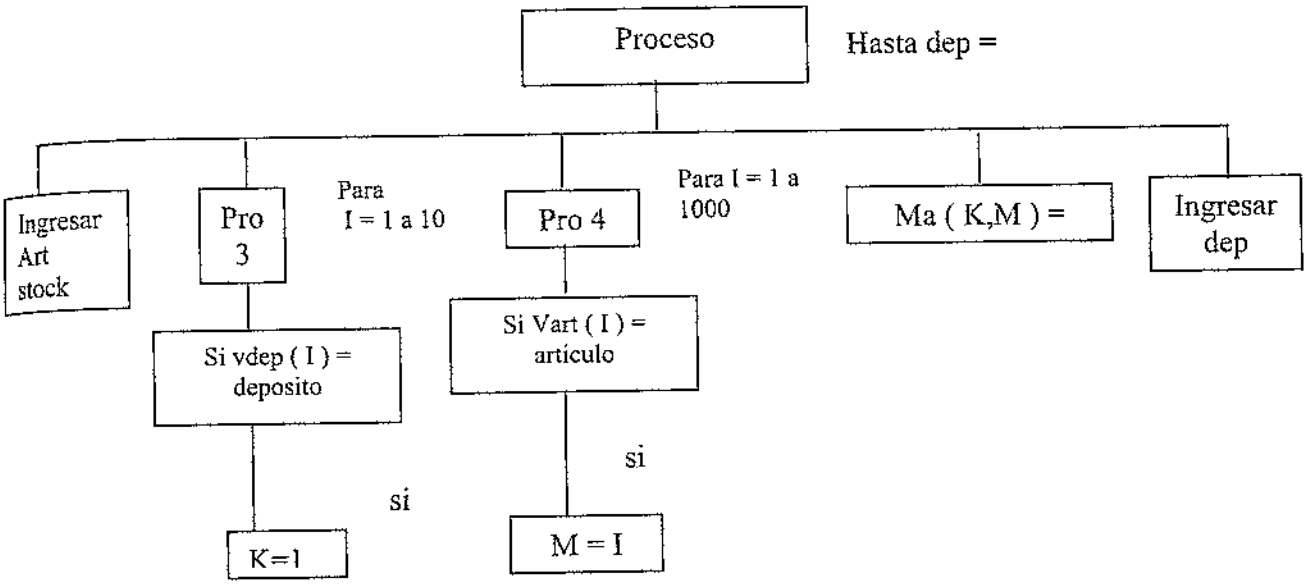
Pues bien lo primero que deberé hacer es cargar los archivos depósito y artículo en sendos vectores, uno de 10 posiciones y el otro de 1000 posiciones



Dentro de inicio



Ahora bien para poder cargar el archivo de stock debo hacer que exista una relación entre las posiciones donde guarde los depósitos y los artículos en los vectores y la matriz que voy utilizar, es por ello que deberé obtener los subíndices de la matriz de estos vectores asociados, para ello en el proceso tendré que hacer lo siguiente :



En seudo codigo seria

```
Comienzo
Para I =1 a 10
        Ingresar "ingrese el código de deposito"
        Ingresar coddep
        Ingresar"ingrese el nombre del deposito"
        Ingresar nombre
        Vdep (I)=coddep
        Vnom (I)=nombre
Próximo
Para k = 1 a 1000
        Ingresar "ingrese el código del articulo"
        Ingresar codart
        Ingresar "ingrese la descripción del articulo"
        Ingresar descripción
        Vart (K)=codart
        Vdes (K)=descripción
Próximo
Ingresar "ingrese el deposito"
Ingresar dep
Hacer hasta dep = 0
```

```
        Ingresar art
        Ingresar stock
        Para I =1 a 10
                Si Vdep (I)=dep entonces
                        K=I
                Fin si

        Próximo
        Para I=1 a 1000
                Si Vart(I) = art entonces
                        M=I
        Fin si
        Próximo
        Ma(K,M) = stock
        Ingresar "ingrese el deposito"
        Ingresar dep
Repetir
Fin
```

EJERCICIOS DE MATRICES

1- Una empresa desea analizar los sueldos a sus 50 empleados durante el año, para ello conoce

- legajo del empleado
- cantidad de horas trabajadas
- valor de la hora
- mes

En un vector en memoria se deben cargar los nombres de los empleados

Se desea saber:

a. sueldo anual de cada empleado
b. total de sueldos pagados cada mes
c. máximo sueldo pagado cada mes y a quién pertenece
d. porcentaje que representa cada sueldo anual sobre el total
e. nombre del empleado que cobró más en el primer semestre
f. cantidad de meses en que el total de sueldos superó los $100000
g. imprimir ordenado en forma descendente por sueldo anual de cada empleado:
    1. legajo
    2. nombre
    3. sueldo anual de cada empleado

2- Una consultora tiene 10 analistas y 15 proyectos en los que trabajan indistintamente, a fin de mes cada analista eleva una planilla con los siguientes datos

- número de analista
- número de proyecto
- cantidad de hs trabajadas

En memoria se debera cargar previamente el nombre de los analistas y el valor de la hora

Se desea saber:

a. cantidad de hs. trabajadas en total por cada analista.
b. total de horas trabajadas por cada analista en cada proyecto.
c. total de hs trabajadas sobre cada proyecto.
d. que analista trabajó menos en el proyecto 1.
e. sueldo de cada analista.
f. nombre del analista que cobró más.
g. cantidad de analistas que hayan trabajado menos de 5 hs en alguno de los proyectos.
h. imprimir ordenado en forma descendente por sueldo de los analista:

   - nombre
   - valor de la hora
   - sueldo

3- Una acopiadora de cereales tiene 20 silos donde almacena 4 tipos de cereal

código de cereal	**descripción**	**valor**
1	trigo	10
2	maíz	12
3	soja	06
4	cebada	11

mensualmente se tiene los datos del cereal almacenado,

- número de silo
- tipo de cereal
- cantidad

Se desea saber:

a. total de kg almacenados de cada cereal
b. total de kg almacenados de cada cereal en cada silo
c. stock valorizado de cada silo
d. nombre del cereal de más almacenaje para cada silo

e. de los silos cuál o cuáles recaudó más
f. cantidad de silos con más de 20000 kg almacenados en total
g. cantidad de silos con más de 2000 kg almacenados para cada cereal
h. porcentaje que representa lo valorizado por cada silo sobre el total

4- Un instituto tiene 20 cursos que comercializa por medio de 5 vendedores, los nombres de los cursos y su costo se deberan cargar en memoria, al cabo del mes cada vendedor presenta un resumen de las ventas realizadas:

- número de vendedor
- número de curso
- cantidad de inscriptos
- porcentual de comisión

Se desea saber:

a. total de alumnos inscriptos por curso
b. total de alumnos inscriptos por vendedor
c. comisión a cobrar por cada vendedor
d. recaudación por curso
e. en qué curso y de qué vendedor se dio la mayor inscripción de alumnos
f. que vendedor inscribio más alumnos en el curso 4
g. cantidad de cursos donde no se inscribió nadie
h. cantidad de cursos donde no inscribió a nadie el vendedor 5

5- Una empresa embotelladora comercializa 5 gaseosas. El costo de embotellamiento es de $ 0.10 de mano de obra, $ 0.15 de lavado de botella, más el valor del liquido a embotelar.

La empresa tiene 20 centros distribuidos a lo largo del país y recarga un 10 % al costo si los centros son el N°. 3, 6, 8 y un descuento del 5% si el centro es el N°. 1. El flete a aplicar es de $ 0.01 por km

Se sabe lo siguiente:

- N° de centro
- nombre
- distancia

Ademas al cabo del mes se realiza una planilla que finaliza con centro igual a 0 donde se tienen los siguientes datos:

- número de centro
- número de gaseosa
- cantidad embotellada
- costo del líquido

Se desea saber:

a. cantidad envasada por centro
b. cantidad envasada por gaseosa
c. cantidad de gaseosa que embotelló cada centro
d. costo total de embotellamiento de cada centro
e. para cada gaseosa indicar en qué centro se embotelló más
f. porcentaje que representa el costo de embotellamiento de cada centro sobre el total
g. si se vende todo lo que se produce con una ganacia del 200% cuánto se recauda por cada centro
h. si de esa recaudación se paga un 30 % de impuestos, que valor es este
i. nombre de la gaseosa de mayor facturación para cada centro
j. nombre de la gaseosa de menor facturación en total

6- Una universidad desea analizar los sueldos abonados a sus 60 docentes, para ello sabe:

Lote 1:
- Legajo del docente.
- Mes.
- Categoría.
- Cantidad de hs. trabajadas.

Lote 2:
- El valor de cada categoría y su descripción se deberan cargar en memoria y son 3, titular, asociado y adjunto.
- No todos los docentes trabajan todos los meses
- La entrada de datos finaliza con número de legajo = 0

Se desea saber:

a. sueldo anual de cada profesor
b. total de sueldos pagados por cada mes
c. para cada profesor cantidad de sueldos mayores a su sueldo promedio
d. máximo sueldo pagado cada mes
e. porcentaje que representa cada sueldo anual sobre el total pagado
f. profesor que cobró menos en el segundo semestre del año
g. que profesor trabajo mas horas en diciembre
h. imprimir ordenado por sueldo anual de menor a mayor
    1. sueldo anual
    2. legajo del profesor
    3. sueldo promedio
    4. porcentaje (punto 5)

7-Un laboratorio fabrica remedios con distintas drogas, son 15 remedios que combinan 20 drogas. El valor de cada remedio depende de las drogas a utilizarse, se tienen dos lotes de datos, el primero es:

-código de droga
-valor por unidad

El segundo lote es el siguiente y termina con remedio igual a 0

-número de remedio
-código de droga
-cantidad

Se desea saber:

a. recaudación del laboratorio
b. recaudación por cada remedio
c. valor de cada remedio
d. porcentaje que representa lo vendido por cada remedio sobre el total
e. precio promedio
f. cuál remedio usa más de la droga 8
g. droga menos usada
h. droga más cara
i. remedio más barato
j. imprimir ordenado por valor de cada remedio,
   valor de cada remedio
   cantidad de remedios vendidas en unidades
   recaudación de cada remedio

8- Una empresa desea analizar los sueldos de sus 80 empleados, para ello cuenta con los siguientes datos en dos lotes distintos

El primero:

- código de categoria (del 1 al 6)
- descripción
- sueldo básico
- valor hora extra

El segundo lote finaliza con legajo = 100

- número de legajo
- categoría
- cantidad de horas extras trabajadas
- mes

En julio y diciembre hay que abonarle el medio aguinaldo

Se desea saber:

a. sueldo anual de cada profesional
b. total pagado por mes
c. para cada legajo cuántos sueldos superan el promedio de cada uno
d. máximo sueldo pagado en un mes y a quién pertenece
e. porcentaje que representa el sueldo anual de cada legajo sobre el total
f. legajo que cobró menos en el segundo semestre
g. quien cobró más en diciembre
h. quien trabajó menos en el primer trimestre del año
i. imprimir ordenado por sueldo anual del profesional
    1. sueldo anual del profesional
    2. legajo
    3. cantidad total de horas extras trabajadas por cada legajo

9- Un taller de computación arma 4 modelos de equipos en 5 puntos del interior del pais,se tienen tres lotes de datos con la siguiente información:

el primero
- número de modelo
- costo

el segundo
- lugar de armado
- costo del flete

Si el lugar de armado es el 4 se le aplica un descuento del 15% al costo por incentivo zona desfavorable
El precio de venta tendra un 150% de ganancia

El tercer lote de datos tiene la siguiente información
- número de modelo
- lugar de armado
- cantidad

Se desea saber:

a. cantidad armada en cada lugar
b. cantidad vendida de cada modelo
c. total facturado por cada modelo
d. porcentual que representa esa facturación sobre el total
e. precio promedio
f. cuál fue la ganancia de la empresa
g. cuál fue el lugar de más producción de equipos en total
h. cuál equipo y adonde se fabricó menos

i. imprimir el punto 2 y 3 ordenado descendentemente por total facturado

10. Una empresa naviera tiene 10 destinos en el caribe y su barco realiza viajes circulares entre ellos, o sea que sale de un puerto y retorna al mismo pasando por todos los puertos intermedios.
Los pasajeros pueden tomar este tipo de viaje o subir en uno y bajar en otro.

Se tiene la siguiente información:

- Número de puerto
- Nombre

Además:

- Número de puerto de salida
- Número de puerto de llegada
- Cantidad de pasajeros

Se desea saber:

a. Cantidad de pasajeros que salieron de un puerto
b. Cantidad de pasajeros que llegaron a un puerto
c. Cantidad de pasajeros que hicieron viajes circulares
d. Puerto donde arribaron más pasajeros
e. Puerto de donde salieron menos pasajeros
f. Porcentual que representan los pasajeros que hicieron viajes circulares sobre el total

Ejercitación general

- Se deberá realizar el díagrama en bloques en jackson
- Las rutinas en seudo código y en programación lineal
- Como mínimo, las rutinas de impresión deberán hacerse por medio de funciones con pasaje de parámetros por medio de valor o de referencia según corresponda

Ejercicio N° 1

La empresa concesionaria de las autopistas estableció 7 categorías de vehículos, para el cobro de los peajes de sus distintas rutas de acceso a la capital federal.
Tiene actualmente la concesión de 4 autopistas

Cada vez que realiza un ticket se tiene la siguiente información

N° de ticket
N° de autopista
Categoría del vehículo
fecha
Hora (en minutos partiendo de las 0 de cada día)
Valor

Se desea saber al termino de cada día de trabajo (N° de ticket = 0) lo siguiente

1. Recaudación general de la empresa
2. Autopista por donde pasaron mas vehículos
3. Porcentaje en pesos de lo abonado por cada tipo de vehículo sobre el total
4. Autopista mas rentable
5. A que hora se hizo el primer ticket del día y en que autopista

Ejercicio N° 2

Una empresa farmacéutica tiene un local para la venta de sus productos.
Dicho local atiende de 08hs a 13hs y de 14hs a 20hs.
Consta de 3 empleados y un farmacéutico los que atienden al publico durante todo el horario seis veces a la semana.
La venta esta centrada sobre sus 8 productos que pertenecen a dos rubros, farmacia y cosmética.

Cada vez que realizan un ticket de venta se registra

N° de ticket

N° de remedio **(1,2,3,4,5,6,7 u 8)**
N° de rubro **(1 o 2)**
Cantidad
Valor unitario
N° de vendedor **(1,2,3 o 4)**

Siendo el empleado 1 el farmacéutico
El ingreso de datos finaliza cuando el N° de ticket es igual a 0

Se desea saber

1. Facturación total de la empresa
2. Valor promedio de los tickets
3. Comisión a cobrar por cada empleado si es igual al 2% del valor de su venta
4. Si el farmacéutico cobra además un 3% sobre la facturación total, cuanto cobrara?
5. Cual fue el N° de ticket de mayor valor y quien lo vendió
6. De que rubro se vendieron mas cantidad de artículos

Ejercicio N° 3

Un club desea controlar las actividades que realizan sus socios

Se sabe que posee 4 categorías de socio que realizan cualquiera de sus 10 disciplinas deportivas a las que están federadas

Para realizarlas deben pagar un plus a la cuota societaria según el deporte y la categoría de socio

Se conoce esos datos que vienen en un lote de datos de 40 ternas

categoría de socio
número de deporte
monto a abonar

Las cuotas mensuales según las categorías son
categoría
monto
Cuando se liquida la cuota mensual se sabe lo siguiente, finalizando este lote con N° de socio = 0

número de socio
categoría
deporte
mes

un socio puede realizar mas de una disciplina deportiva

Se desea saber

1. socio que realiza mas disciplinas deportivas
2. socio que abona mas por mes
3. disciplina deportiva donde no se anoto nadie
4. disciplina deportiva con menos socios (exceptuar los 0)
5. categoría con mas socios

Ejercicio N° 4

Una empresa de turismo realiza excursiones a las cuatro regiones del mundo

región	**nombre**	**costo**
1	América	3500
2	Europa	4000
3	Asia	4500
4	África	2000

Cada vez que recibe un pedido tomado por alguna de sus 6 agencias, tiene la siguiente información, finalizando con número de pedido 0 .

número de pedido
agencia
mes
destino
cantidad de personas mayores
cantidad de personas menores

Se sabe que los menores pagan un 75% del valor del tour

Se desea conocer

1. que porcentaje representa lo recaudado por cada agencia sobre el total
2. cuantos pasajeros viajaron a cada destino en cada mes
3. que agencia tomo mas pedidos en el mes de julio
4. cuantos pedidos superaron los 11 pasajeros en total [mayores + menores]
5. imprima ordenado por destino
   la recaudación
   la cantidad total de pasajeros transportados

Ejercicio N° 5

una empresa de sepelios tiene distintos servicios para prestar a sus clientes

tipo de servicio	valor	cantidad de autos	cantidad de personal
1	1000	3	6
2	1500	5	8
3	700	2	5
4	3000	5	10

cada vez que le solicitan un servicio tiene la siguiente información que finaliza con N° de servicio = 0

N° de servicio
día del servicio (1 al 31)
tipo de servicio (1 al 4)
zona del servicio (1 al 100)

Se desea:

1. emitir la factura a cada número de servicio
2. cantidad de autos y de personal necesarios para cada día del mes
3. para cada zona, que tipo de servicio es el mas solicitado
4. cuantos servicios hay en cada zona por día
5. imprima ordenado por recaudación por la recaudación de cada tipo de servicio
   tipo de servicio
   recaudación
   cantidad de servicios realizados

Ejercicio N° 6

Un torneo de football consta de 8 equipos que juegan entre sí para la obtención del campeonato.

Cada vez que se realiza un partido, cada uno de los 4 referees indica lo siguiente

N° de partido
Equipo 1
Equipo 2
Referee
Equipo ganador
Cantidad de penales

Siempre hay un ganador, ya sea por el resultado del partido o por la ejecución de penales si el encuentro finalizo empatado.

En total se juegan 57 partidos .

Se desea saber

- Equipo ganador del campeonato
- Referee que cobro o adjudico menos penales en un mismo partido
- Cantidad promedio de goles por partido
- Porcentaje que representa los partidos ganados por el equipo campeón sobre el total de partidos realizados

Ejercicio N° 7

Una pizzería cocina 6 tipos de pizza que son repartidas por medio de sus 4 motos a su numerosa clientela en un radio de 20 cuadras a la redonda de su local.

Cuando el pedido es enviado se controla

N° de ticket
Código de pizza
Cantidad
N° de moto
Monto de la venta

Los datos finalizan con el N° de ticket igual a 0.

Se desea saber

- Ticket de mayor valor y que moto llevo ese pedido
- Moto que hizo menos viajes
- Valor promedio de los tickets
- Que porcentaje representa la cantidad de pizzas "A" sobre el total de envíos realizados
- Facturación total del negocio

Ejercicio N° 8

Un administrador de consorcios de edificios atiende a 10 edificios .como todos los edificios fueron realizados por medio de planes de fomento, son todos iguales y tienen cada uno de ellos 50 departamentos de tres ambientes .

Mensualmente se realizan distintas tareas de mantenimiento en estos edificios debiendo ser las mismas pagadas en forma proporcional en partes iguales por los habitantes de cada uno de ellos .

Cada vez que se realiza un arreglo se obtiene la siguiente información

N° de factura
N° de edificio
Código del proveedor
Monto de la factura
Fecha de vencimiento

Estos datos finalizan con N° de factura igual a 0 .

Los proveedores son 5, ascensorista, electricista, pintor, plomero y limpieza.
Estos proveedores son los mismos para todos los edificios.

Se desea saber

- Monto total gastado por cada edificio
- Mayor gasto efectuado, a que edificio y proveedor pertenece
- Valor a pagar en calidad de expensas
- Proveedor que no trabajo en ninguno de los edificios
- Valor promedio de las facturas

Ejercicio N° 9

Ingresar los siguientes campos:
Código de Empleado
Categoría. Puede ser de 1 a 4.
Antigüedad. Es un valor en $.

El ingreso finaliza cuando se ingresa una categoría 0.

El sueldo básico es según la categoría.
Para categoría 1: $3000
Para categoría 2: $2000
Para categoría 3: $1000
Para categoría 4: $500

Mostrar por pantalla:

* De cada empleado el Código y su Sueldo.

Al final, una estadística que muestre por pantalla:

* Valor del máximo y mínimo sueldos.
* Total de sueldos por categoría.
* Máximo por categoría
* Cantidad de empleados por categoría

Ejercicio N° 10

Una compañía de transportes de larga distancia, realiza viajes a cuatro destinos del interior del país:

1. Capital Federal (Buenos Aires)
2. Rosario (Santa Fe)
3. Resistencia (Chaco)
4. San Rafael (Mendoza)

Se recopilan al cabo de cada semana los datos que figuran en cada uno de los pasajes:

a. Número de pasaje
b. Ciudad de partida.
c. Ciudad de destino.
d. Número de coche (dispone de 4 coches).
e. Precio del pasaje.

El ingreso de datos finaliza con un número de pasaje igual a 0 (cero).

Se pide:
1. Monto total de pasajes (en pesos) vendido en el mes.
2. Cantidad de pasajes que tuvieron como origen la Capital Federal y destino la ciudad de San Rafael.
3. Número de coche que más viajes realizó en la semana.
4. Porcentaje de viajes a Resistencia, respecto del total de viajes realizados.

Ejercicio N° 11

Una empresa comercializadora de máquinas expendedoras automáticas ofrece el servicio de cafetería mediante la gestión de 3 empresas representantes.

Cada vez que un representante necesita reponer stock emite *un pedido* solicitando la cantidad de unidades que necesita, en un formulario donde consta:

- **Número de pedido**
- **Número de representante**
- **Número de producto**
- **Cantidad de unidades**

Los productos que se comercializa y sus costos son:

1-	café	$ 1.00-
2-	leche chocolatada	$ 1.40.-
3-	agua mineral	$ 0.80.-
4-	gaseosa en lata	$ 1.00.-

La información termina con número de pedido igual a cero.

Se desea procesar toda la información poseída y luego emitir un informe respondiendo a los siguientes requerimientos:

1. Recaudación total de la empresa.
2. Cantidad de unidades pedidas por producto.
3. N°. de Pedido de máxima cantidad de unidades pedidas en un pedido.
4. Sabiendo que se paga en impuesto el 20% de la recaudación total, cual es dicho valor.
5. Porcentaje en recaudación por representante respecto al total de la empresa.

Ejercicio N° 12

Se ingresan los siguientes datos, correspondientes a los empleados de una empresa:
La información finaliza con N°. de legajo = 0.

- N°. de legajo
- Categoría (A, B ó C)
- Años de antigüedad
- Horas extras trabajadas (cantidad)

La forma de calcular el sueldo bruto es la siguiente:

- Para la categoría A el sueldo básico es de $ 600
- Para la categoría B el sueldo básico es de $ 800
- Para la categoría C el sueldo básico es de $ 1.200

Asimismo se le deberá adicionar el concepto antigüedad, según la siguiente escala:

- Para la categoría A se computará $ 50 por c/año.
- Para la categoría B se computará $ 40 por c/año.
- Para la categoría C se computará $ 30 por c/año.

Para el cálculo definitivo del sueldo bruto deberá adicionarse el valor de las horas extras que se calcularán en base a la siguiente fórmula:

Valor de la hora extra = (Sueldo básico + Antigüedad) /180

Se pide determinar e informar:

a) El sueldo bruto total para cada empleado.
b) El sueldo neto para cada empleado, considerando que c/u tiene de descuento por retenciones un 17% sobre el sueldo bruto.
c) La cantidad total de sueldos netos pagados para cada categoría.
d) Sueldo neto promedio
e) Sueldo máximo y a que legajo y categoría pertenece.
f) Porcentual en cantidad de empleados de c/categoría sobre el total.

Ejercicio N° 13

Una empresa de transporte y logística une Bs.As con 3 destinos empleando una dotación de personal de ventas de solo 6 personas.

1. Rosario (320 km).
2. Córdoba (790 km).
3. Mendoza (1070 km).

Cada vez que se realiza un transporte se emite una carta de porte con la siguiente información:

- Número de destino
- Número de empleado.
- Monto del transporte
- Kilometraje recorrido

El lote de cartas de porte finaliza con el número 0.

Se pide:

1. Elaborar el diagrama de Jackson para el problema.
2. Calcular los kilómetros recorridos por los camiones de la empresa.
3. Cantidad de transportes realizados para cada destino.
4. Monto total vendido por cada empleado
5. Porcentaje que representa lo vendido para cada destino sobre el monto total vendido.

Ejercicio N° 14

Una cadena de farmacias tiene 6 sucursales y vende medicamentos de 3 droguerías.
Cada vez que vende un medicamento, se anotan los siguientes datos que terminan con número de factura 0:

- Número de factura.
- Número de sucursal.
- Número de droguería.
- Valor de la venta.

Se pide:

1. Elaborar el diagrama de Jackson para el problema.
2. Venta total para cada droguería.
3. Cantidad de facturas emitidas para cada droguería.
4. Venta total efectuada.
5. Cuál droguería vendió más en dinero.

Ejercicio N° 15

Una editorial vende libros por Internet, los que se clasifican en cinco (5) categorías: :

1. Electrónica
2. Informática
3. Marketing
4. Gestión
5. Publicidad

El último día de cada mes se recopilan los datos sobre las ventas realizadas, de las cuales se conocen:

Código de la venta
Categoría del libro
Código del producto
Precio

En cada transacción se vende sólo un producto. El ingreso de datos finaliza con un número de transacción igual a 0 (cero).

Se pide:

1. Total en pesos vendido en el mes.
2. Código y precio del producto más caro.
3. Porcentaje de libros de Electrónica vendidos en el mes respecto del total
4. Total de ventas de libros de Marketing

Ejercicio N° 16

Una empresa dedicada al comercio exterior realiza operaciones de importación y exportación a 8 regiones del mundo
cada vez que realiza una operación tiene los siguientes datos

número de operación
tipo de operación (1-importación / 2- exportación)

cantidad de artículos
precio FOB unitario
mes

estos datos finalizan con número de operación = 0

dependiendo del tipo de operación y de la región con que se trabaje se deberán abonar impuestos sobre el precio FOB, obteniendo así el precio CIF

N° de región
tipo de operación
impuesto (en porcentaje)

Se desea saber

1. saldo de la balanza comercial de la empresa (importación – exportación)
2. que región compro mas artículos
3. a que región debemos menos dinero, según el saldo (exceptuar los 0)
4. cual fue la operación que más pago en impuestos
5. que mes fue el de mayor cantidad de exportaciones a la región 5

Ejercicio N° 17

Una empresa farmacéutica tiene un local para la venta de sus productos.
Dicho local atiende de 08hs a 13hs y de 14hs a 20hs.
Consta de 3 empleados y un farmacéutico los que atienden al publico durante todo el horario seis veces a la semana.
La venta esta centrada sobre sus 8 productos que pertenecen a dos rubros, farmacia y cosmética.

Cada vez que realizan un ticket de venta se registra

N° de ticket
N° de remedio **(1,2,3,4,5,6,7 u 8)**
N° de rubro **(1 o 2)**
Cantidad
Valor unitario
N° de vendedor **(1,2,3 o 4)**

Siendo el empleado 1 el farmacéutico
El ingreso de datos finaliza cuando el N° de ticket es igual a 0

Se desea saber:

1. Facturación total de la empresa.
2. Valor promedio de los tickets.
3. Comisión a cobrar por cada empleado si es igual al 2% del valor de su venta.
4. Si el farmacéutico cobra además un 3% sobre la facturación total, cuánto cobrará?
5. Cuál fue el N° de ticket de mayor valor y quién lo vendió?
6. De qué rubro se vendieron mas cantidad de artículos?

Ejercicio N° 18

Una empresa textil comercializa sus productos por medio de sus 6 sucursales en distintos puntos de la capital federal

Sus productos tienen los siguientes talles y valores

Código de producto	**talle**	**Precio de venta**
1	S	50
2	M	50
3	L	55
4	Xl	60
5	XXL	60
6	BIG	80

Cada vez que realiza una venta emite una factura con el siguiente grupo de datos

N° de sucursal
Código de producto
Cantidad vendida

Se desea conocer:

1. Cual factura fue la de mayor valor
2. Cuantas prendas se vendieron del talle S
3. Recaudación de cada sucursal por cada producto
4. Que sucursal vendió mas prendas en total
5. Imprimir ordenado por sucursal, su recaudación total y la cantidad total de prendas vendidas en ella

Ejercicio N° 19

Una empresa arenera tiene 20 barcos que transportan arena a 30 puertos

Se conoce de cada uno de ellos

N° de barco
puerto en que se encuentra
cantidad de carga actual

Cada vez que solicitan un nuevo cargamento envían la siguiente información

N° de pedido
puerto en el cual se necesita
cantidad de Kg. necesarios

Determinar

1. cuantos pedidos pudieron ser satisfechos
2. cuantos barcos hay en el mismo puerto
3. puerto donde no hay ningún barco

Ejercicio N° 20

Un laboratorio recibe muestras para su análisis
cada tipo de muestra posee un varios tipos de análisis

tipo de muestra
tipo de análisis
costo de realización
valor a cobrar

Los tipos de muestra son 40 y a cada una de ellas se le hace a lo sumo 10 tipos de análisis

Cuando recibe una muestra se sabe:

número de muestra
tipo de muestra
tipo de análisis

finaliza con N° de muestra = 0

Se desea saber:

1. muestra en la que se gano mas dinero
2. que tipo de análisis es el menos realizado
3. para cada tipo de muestra cual es el tipo de análisis menos rentable

Ejercicio N° 21

El servicio penitenciario debe albergar los delincuentes según su peligrosidad en distintos pisos
En la sede en estudio se sabe:

N° de piso
grado de peligrosidad
cantidad de lugares libres
costo

Posee 10 pisos con 5 grados de peligrosidad cada uno

Cuando reciben del juzgado las planillas con los nuevos internos deben indicar

1. nombre y juzgado de los que no pueden ser alojados por falta de espacio
2. cual es el nuevo presupuesto que deben solicitar por los reclusos aceptados
3. cual piso es el que tiene más reclusos
4. cual piso es el que necesita mas presupuesto
5. cuantos fueron rechazados por falto de espacio en total

Cada vez que se recibe la planilla se tiene los siguientes datos

N° de recluso
grado de peligrosidad
apellido y nombre
juzgado

los datos finalizan con N° de recluso = 0

Ejercicio N° 22

El servicio penitenciario tiene que albergar a los nuevos reclusos que les envía el juzgado

Se conoce la capacidad del penal por medio de los siguientes datos:

N° de piso
N° de celda
capacidad libre

existen 10 pisos con 40 celdas cada uno

cada vez que recibe un recluso tiene la siguiente información

N° de recluso
apellido y nombre

finaliza el ingreso de datos con N° de ingreso = 0

Se desea saber:

1. cuantos reclusos no pudieron ser alojados por falta de lugar
2. apellido y nombre de los rechazados
3. para cada piso que celda tiene aun disponibilidad
4. piso mas lleno

NOTA : se debe verificar en todo el penal la disponibilidad de lugar y adjudicar ese lugar en el primero que se encuentre, por lo cual dentro del proceso existirá una rutina para buscar este lugar que será hecha por medio de ciclos FOR

Ejercicio N° 23

Un instituto de ingles realiza la inscripción a sus cursos anuales para ello conoce

N° de curso
nivel de ingles
cantidad de lugares libres
cupo máximo
costo

sabe que tiene 20 cursos en 6 niveles distintos cada uno.

cada vez que viene un postulante se le toma un examen para determinar su nivel

se registra pues la inscripción que finaliza con N° de legajo = 0

N° de legajo del alumno
nivel alcanzado

Se desea saber :

1. cuantos alumnos se presentaron en cada nivel
2. porcentaje que representa la capacidad ocupada sobre el cupo máximo de cada curso
3. recaudación final de la inscripción
4. curso mas lleno

Ejercicio N° 24

Una empresa de transporte y logística une Bs.As con 3 destinos empleando una dotación de personal de ventas de solo 6 personas.

1. Rosario (320 km).
2. Córdoba (790 km).
3. Mendoza (1070 km).

Cada vez que se realiza un transporte se emite una carta de porte con la siguiente información:

- Número de destino
- Número de empleado.
- Monto del transporte
- Kilometraje recorrido

El lote de cartas de porte finaliza con el número 0.

Se pide:

1. Elaborar el diagrama de Jackson para el problema.
2. Calcular los kilómetros recorridos por los camiones de la empresa.
3. Cantidad de transportes realizados para cada destino.
4. Monto total vendido por cada empleado
5. Porcentaje que representa lo vendido para cada destino sobre el monto total vendido.

Ejercicio N° 25

Una cadena de farmacias tiene 6 sucursales y vende medicamentos de 3 droguerías.
Cada vez que vende un medicamento se anotan los siguientes datos que terminan con número de factura 0:

- Número de factura.
- Número de sucursal.
- Número de droguería.
- Valor de la venta.

Se pide:

1. Venta total para cada droguería.
2. Cantidad de facturas emitidas para cada droguería.
3. Venta total efectuada.
4. Cuál droguería vendió más en dinero.

Ejercicio N° 26

Se ingresan los siguientes datos, correspondientes a los empleados de una empresa:

La información finaliza con N°. de legajo = 0.

- N°. de legajo
- Categoría (A, B ó C)
- Años de antigüedad
- Horas extras trabajadas (cantidad)

La forma de calcular el sueldo bruto es la siguiente:

- Para la categoría A el sueldo básico es de $ 600
- Para la categoría B el sueldo básico es de $ 800
- Para la categoría C el sueldo básico es de $ 1.200

Asimismo se le deberá adicionar el concepto antigüedad, según la siguiente escala:

- Para la categoría A se computará $ 50 por c/año.
- Para la categoría B se computará $ 40 por c/año.
- Para la categoría C se computará $ 30 por c/año.

Para el cálculo definitivo del sueldo bruto deberá adicionarse el valor de las horas extras que se calcularán en base a la siguiente fórmula:

Valor de la hora extra = (Sueldo básico + Antigüedad) /180

Se pide determinar e informar:

1. El sueldo bruto total para cada empleado.
2. El sueldo neto para cada empleado, considerando que c/u tiene de descuento por retenciones un 17% sobre el sueldo bruto.
3. La cantidad total de sueldos netos pagados para cada categoría.
4. Sueldo neto promedio
5. Sueldo máximo y a que legajo y categoría pertenece.
6. Porcentual en cantidad de empleados de c/categoría sobre el total.

Ejercicio N° 27

Una editorial vende libros por Internet, los que se clasifican en cinco (5) categorías: :

4. Electrónica
5. Informática
6. Marketing
7. Gestión
8. Publicidad

El último día de cada mes se recopilan los datos sobre las ventas realizadas, de las cuales se conocen:

a. Código de la venta
b. Categoría del libro
c. Código del producto
d. Precio

En cada transacción se vende sólo un producto. El ingreso de datos finaliza con un número de transacción igual a 0 (cero).

Se pide:

1. Total en pesos vendido en el mes.
2. Código y precio del producto más caro.
3. Porcentaje de libros de Electrónica vendidos en el mes respecto del total
4. Total de ventas de libros de Marketing

Ejercicio N° 28

Una compañía de transportes de larga distancia, realiza viajes a cuatro destinos del interior del país:

2. Capital Federal (Buenos Aires)
3. Rosario (Santa Fe)
4. Resistencia (Chaco)
5. San Rafael (Mendoza)

Se recopilan al cabo de cada semana los datos que figuran en cada uno de los pasajes:

a. Número de pasaje
b. Ciudad de partida.
c. Ciudad de destino.
d. Número de coche (dispone de 4 coches).
e. Precio del pasaje.

El ingreso de datos finaliza con un número de pasaje igual a 0 (cero).

Se pide:

1. Monto total de pasajes (en pesos) vendido en el mes.
2. Cantidad de pasajes que tuvieron como origen la Capital Federal y destino la ciudad de San Rafael.
3. Número de coche que más viajes realizó en la semana.
4. Porcentaje de viajes a Resistencia, respecto del total de viajes realizados.

Ejercicio N° 29

Una empresa comercializadora de máquinas expendedoras automáticas ofrece el servicio de cafetería mediante la gestión de 3 empresas representantes.

Cada vez que un representante necesita reponer stock emite *un pedido* solicitando la cantidad de unidades que necesita, en un formulario donde consta:

- **Número de pedido**
- **Número de representante**
- **Número de producto**
- **Cantidad de unidades**

Los productos que se comercializa y sus costos son:

1-	café	$ 1.00-
2-	leche chocolateada	$ 1.40.-
3-	agua mineral	$ 0.80.-
4-	gaseosa en lata	$ 1.00.-

La información termina con número de pedido igual a cero.

Se desea procesar toda la información poseída y luego emitir un informe respondiendo a los siguientes requerimientos:

1. Recaudación total de la empresa.
2. Cantidad de unidades pedidas por producto.
3. N°. de Pedido de máxima cantidad de unidades pedidas en un pedido.
4. Sabiendo que se paga en impuesto el 20% de la recaudación total, cual es dicho valor.
5. Porcentaje en recaudación por representante respecto al total de la empresa.

Ejercicio N° 30

Un supermercado tiene dividido a sus productos en 14 sectores que reciben distintos descuentos según el día de la semana .
Todos los productos son facturados por medio de sus 20 cajas que a su vez son atendidas en forma rotativa por 30 empleadas.

Los descuentos se publican una vez a la semana y contienen la siguiente información:

N° de sector (1 al 14)
Día de la rebaja (1 al 7)
Porcentaje de descuento

Cada vez que se realiza un ticket se anotan los siguientes datos, que finalizan con sector igual a 0 .

Sector (1 al 14)
Día (1 al 7)
Caja (1 al 20)
Empleada (1 al 30)
Monto del ticket sin descuento

Se desea saber:

- Recaudación de cada caja en cada día
- Sector del que se emitieron mas tickets
- Que empleada emitió menos tickets y en cual caja
- Día de mayor recaudación
- Porcentaje de lo recaudado por cada sector sobre el total (aplicado el descuento)

Se pueden bajar ejercicios para practicar, como así también ejercicios resueltos con menú y funciones, de la página de la facultad de Tecnología Informática de la UAI:

www.uai.edu.ar

Opción: - FALCULTADES
/Tecnología Informática
/Material de cátedra para docentes y alumnos

Bibliografia consultada

Programación en **C**, Metodología, algoritmos y estructuras de datos, Aguilar – Martínez, Mc Grau Hill

C con ejemplos, Perry, Prentice May.

Manual de programación Estructurada, Ing R. Brizuela, UAI, 1998.