UNIDAD 5

"LENGUAJE C ESTRUCTURADO"
OBJETIVO: APRENDER A MANEJAR EL LENGUAJE DE PROGRAMACIÓN C PARA RESOLVER PROBLEMAS PREVIAMENTE PLANTEADOS.

5.1 TEORIA DEL DISEÑO DE PROGRAMAS. VINCULACIÓN DEL DISEÑO DE PROGRAMAS AL CONOCIMIENTO ALGORITMICO.
5.2 CARACTERÍSTICAS BÁSICAS DE UN PROGRAMA EN LENGUAJE C. CONSTANTES, VARIABLES, OPERADORES, INSTRUCCIONES Y DECLARACIONES.
5.3 INSTRUCCIÓN DE ASIGNACIÓN. FUNCIONES INTRINSECAS ELEMENTALES.
5.4 INSTRUCCIONES DE ENTRADA Y SALIDA.
5.5 REPRESENTACIÓN DE LAS ESTRUCTURAS DE CONTROL.
5.6 FUNCIONES Y SUBPROGRAMAS.
5.7 ELABORACIÓN DE PROBLEMAS BÁSICOS DE INGENIERÍA.

Antecedentes:

"Si C tuviera escudo de armas, su lema sería multum in parvo. Mucho a partir de poco".
Les Hancock.

Se usan muchos lenguajes para programar una computadora. El más básico es el lenguaje máquina (una colección de instrucciones muy detalladas que controlan los circuitos internos de la computadora). Muy pocos programas se escriben en lenguaje máquina por dos razones:

- Es muy incomodo para trabajar.
- La mayoría de las máquinas tiene su propio repertorio de instrucciones.

Lo más frecuente es utilizar lenguajes de alto nivel, cuyas instrucciones son más compatibles con el lenguaje del ser humano. La mayoría son lenguajes de propósito general como: Pascal, Basic, Fortran, Cobol, etc. Hay también lenguajes de propósito especial su aplicación esta diseñada para una aplicación personal como el LISP (Lenguaje Orientado al Tratamiento de Listas).

Por norma general una sola instrucción escrita en lenguaje de alto nivel será equivalente a varias de lenguaje máquina. Además un lenguaje de alto nivel ofrece tres ventajas importantes respecto al lenguaje máquina:

- Sencillez.
- Uniformidad.
- Portabilidad.

Un programa escrito en lenguaje de alto nivel, se puede ejecutar en cualquier máquina sin modificaciones (o muy pocas). El programa a de ser traducido a lenguaje máquina antes de ser ejecutado. A esto se le conoce como compilación o interpretación, dependiendo como se lleve a cabo. En cualquier caso la traducción se lleva a cabo en forma automática por la computadora. De hecho, los programadores recién iniciados no se dan cuenta de que esta acción esta ocurriendo, ya que típicamente solo ven el programa original en alto nivel, los datos de entrada y la salida resultante.

El lenguaje C es un lenguaje de programación estructurada de propósito general. Sus instrucciones constan de términos que se parecen a expresiones algebraicas. Además de ciertas palabras clave como: if-else, for, do y while. En este sentido C recuerda a otros lenguajes de programación estructurada como Pascal. También tiene algunas características adicionales que permiten su uso a un nivel más bajo, cubriendo así el vacío entre lenguaje máquina y lenguaje de alto nivel. Esta flexibilidad permite el uso del lenguaje C en la programación de sistemas (diseño de sistemas operativos) así como en la programación de aplicaciones (programas matemáticos, de facturas, etc).

Tiene un conjunto de instrucciones relativamente pequeño, aunque las implementaciones actuales incluyen numerosas funciones de biblioteca que manejan las instrucciones básicas. También permite al usuario escribir funciones de biblioteca adicionales para su uso propio. De esta forma las características y capacidades del lenguaje se pueden ampliar fácilmente por el usuario.

Historia del lenguaje C:

Lenguaje C fue desarrollado en 1972 por Dennis Ritchie en Bell Telephone Laboratories Inc. (ahora AT&T Laboratories). Estuvo confinado al uso de laboratorios Bell hasta 1978, cuando Brian Kern y Dennis Ritchie publicaron una descripción definitiva del lenguaje C (The C programming languaje).

Para la mitad de los 80 's la popularidad de C se había extendido por todas partes. Se habían escrito numerosos compiladores e intérpretes de C para computadoras de todos los tamaños y se habían desarrollado muchas aplicaciones comerciales. Es más muchas aplicaciones que se habían escrito originalmente en otro lenguaje se rescribieron en C para tomar partido de su eficiencia y portabilidad.

Como se dieron algunas diferencias entre las implementaciones, la American National Estándar Institute (ANSI) comenzó a trabajar con una definición normalizada del lenguaje C (Comité ANSI X3J11), se conoce como estándar ANSI o "ANSI C".
El sistema operativo UNIX fue originalmente escrito en C por el mismo equipo de investigadores de AT&T.

Genealogía de lenguaje C:

1. ALGOL 60.
2. CPL (Lenguaje de Programación Combinado). Cambridge y la Universidad de Londres, 1963.
3. BCPL(Lenguaje Basico de Programación Combinado). Martín Richards, Cambridge, 1967.
4. B. Ken Thompson, Laboratories Bell.
5. C. Dennis Ritchie, Laboratories Bell.

Características de C:

- Es un lenguaje de propósito general.
- Es un lenguaje pequeño.
- Es muy poderoso debido a sus capacidades de lenguaje de bajo nivel.
- Es fácil de aprender.
- Existe una estrecha relación con UNIX.
- Es portátil.
- Es elegante.

Estructura de un programa en C:

Todo programa en C consta de una o más funciones, una de las cuales se llama main. El programa siempre comenzará por la ejecución de la función main. Cada función debe contener.

- Una cabecera de función, que consta del nombre de la función, seguida de una lista opcional de argumentos encerrados en paréntesis.
- Una lista de declaración de argumentos.
- Una sentencia compuesta, que contiene el resto de la función.

Los argumentos son símbolos que representan información que se le pasa a la función desde otra parte del programa.

Cada sentencia compuesta está encerrada por un par de llaves { } . las llaves pueden contener combinaciones de sentencias elementales (denominadas sentencias de expresión) y otras sentencias compuestas. Así, las sentencias compuestas pueden estar anidadas una dentro de otra. Cada sentencia de expresión debe acabar con un punto y coma (;).

Los comentarios pueden aparecer en cualquier parte del programa, mientras estén situados dentro de los delimitadores (/* */). Los comentarios son útiles para identificar a los elementos principales de un programa, o para explicación de algoritmo.

EJEMPLO:

```
/* Programa que determina el área de un circulo/ /*Comentario dando identificación al programa*/
 #include<stdio.h>                    /* Acceso a archivo de biblioteca*/
 main()                                          /* Cabecera de función */
   {                                             /* Se abre a las sentencias */
      float radio, area;                         /* Declaración de variables */
      printf("Radio = ¿? ");                     /* Sentencia de salida */
      scanf("%f",&radio);                        /* Sentencia de entrada */
      area=3.1416*radio*radio;                   /* Sentencia de asignación */
      printf("Area = %f", area);                 /* Sentencia de salida */
   }                                             /* Se cierra a la sentencia */
```

NOTA: Se hicieron los comentarios al final de cada línea para explicar la organización del programa en conjunto. Normalmente un programa en C no se parecerá al programa anterior, sino más bien al programa que aparece a continuación.

```
/* Programa número 1 */
/* Programa que determina el area de un circulo*/
#include<stdio.h>
main()
{
      float radio, area;
      printf("Radio = ¿? ");
      scanf("%f",&radio);
      area=3.1416*radio¨radio;
      printf("Area = %f", area);
}
```

NOTA: Pero nosotros vamos a usar devc++ por lo que el programa aparecerá como se describe a continuación.

```
#include <cstdlib>
#include <iostream>
/*programa numero 1 version 1*/
/*programa que determina el area de un circulo*/
using namespace std;
int main()
{
  system("cls");
   float radio, area;
   printf("\n\n\t\tPrograma No 1  version 1\n");
   printf("\n\n\t\tPrograma que determina el area de un circulo\n");
   printf("\n\n\t\tDame el valor de tu radio    ");
   scanf("%f",&radio);
   area=3.1416*radio*radio;
   printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   system("PAUSE");
   return 0;
}
```

Características de un programa en lenguaje C:

Tomando el primer ejemplo, podemos señalar las siguientes características:

1. El programa esta escrito en minúsculas sin acentuar (salvo en comentario). En C las letras mayúsculas y minúsculas no son equivalentes.
2. La primera línea es un comentario que describe el propósito del programa.
3. La segunda línea (include stddio.h) hace una referencia a un archivo especial que contiene información que se debe incluir (include) en el programa cuando se compila. La inclusión requerida de esta información será manejada automáticamente por el compilador.
4. La tercera línea (main) es la cabecera de la función, los paréntesis vacíos indican que ni incluye argumentos.
5. Las siguientes cinco líneas adentradas y encerradas por un par de llaves ({}). Estas cinco líneas integran la sentencia compuesta dentro de main.
6. La primera línea adentrada es una declaración de variables. Se establecen los nombres simbólicos de radio y área como variables de coma flotante.
7. La segunda línea adentrada (printf) genera una solicitud de información (el valor del radio) que aparecerá en la pantalla.
8. La tercera línea adentrada (scanf) indica que se va a introducir el valor del radio a la computadora por medio del teclado.
9. La cuarta línea adentrada (área=) es un tipo particular de sentencia de expresión llamada sentencia de asignación. Esta sentencia hace el cálculo del área a partir del valor de radio dado. Dentro de estas sentencias el * significa la multiplicación.
10. La quinta línea adentrada (printf) desplegará el valor calculado de área. El valor numérico será precedido por un breve mensaje (“Area = “).
11. Nótese que cada sentencia de expresión dentro de la sentencia compuesta acaba con un punto y coma (;). Esto es necesario en toda sentencia de expresión.
12. Por último se señala el uso de espaciados y tabuladores, creando espacio en blanco dentro del programa. Las líneas en blanco separan partes diferentes del programa en componentes lógicamente identificables. La adentración indica relaciones de subordinación entre varias instrucciones. Estos elementos no son esenciales gramaticalmente, pero su presencia es fundamental en la práctica de una buena programación.

EJERCICIO:

Realizar los siguientes programas:

```
#include <cstdlib>
#include <iostream>
/*Programa No 2  version 1*/
/*Programa que despliega un mensaje*/
using namespace std;
int main()
{
   system("cls");
   printf("\n\n\t\tPrograma No 2  version 1\n");
   printf("\n\n\t\tPrograma que despliega un mensaje\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 2  version 1\n");
   printf("\n\n\t\tHOLA A TODOS\n");
   printf("\n\n\t\tBIENVENIDOS AL MUNDO\n");
   printf("\n\n\t\tDE LA PROGRAMACION\n\n\n\t\t");
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
/*Programa No 2  version 2*/
/*Programa que despliega un mensaje*/
using namespace std;
#define mensaje1 "\n\n\t\tPrograma No 2  version 2\n"
#define mensaje2 "\n\n\t\tHOLA A TODOS\n"
#define mensaje3 "\n\n\t\tBIENVENIDOS AL MUNDO\n"
#define mensaje4 "\n\n\t\tDE LA PROGRAMACION\n\n\n\t\t"
int main(int argc, char *argv[])
{
   system("cls");
   printf("\n\n\t\tPrograma No 2  version 2\n");
   printf("\n\n\t\tPrograma que despliega un mensaje en pantalla\n\n\t\t");
   system("PAUSE");system("cls");
   printf(mensaje1);
   printf(mensaje2);
   printf(mensaje3);
   printf(mensaje4);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
/*Programa No 2  version 3*/
/*Programa que despliega un mensaje*/
using namespace std;
int hola(void)
  {
   system("cls");
   printf("\n\n\t\tHOLA A TODOS\n");
   return EXIT_SUCCESS;
  }
int (mundo void)
{
   printf("\n\n\t\tBIENVENIDOS AL MUNDO\n");
   printf("\n\n\t\tDE LA PROGRAMACION\n\n\n\t\t");
   return EXIT_SUCCESS;
}
int main()
{
   system("cls");
   printf("\n\n\t\tPrograma No 2  version 3\n");
   printf("\n\n\t\tPrograma que despliega un mensaje\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 2  version 3\n");
   hola();
   mundo();
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Aquí se explicará las características de cada programa (checar que cada uno es diferente, pero despliega el mismo resultado).
CHECAR QUE:

hola(); es una sentencia
hola() es una función.

Identificadores:
Son nombres que se utilizan para referenciar variables, funciones, etiquetas y otros objetos definidos por el usuario. No puede ser una palabra reservada (while, break, if, main, return, etc).

El identificador puede estar formado por uno o varios caracteres. El primer carácter debe ser una letra o un subrayado.
El caracter _ es utilizado como carácter de inicio de identificadores dentro de las rutinas de la biblioteca estándar.
En turbo C solamente los primeros 32 caracteres son significativos.

EJEMPLO:

Son incorrectos los siguientes identificadores.

- 1contador
- hola!tu
- auto..balance

Son correctos los siguientes identificadores.

- contador
- holatu
- auto_balance

Tipos y tamaños de datos:

Un tipo de datos es un conjunto de valores y un conjunto de operaciones que se pueden realizar entre ellos.
Existen tres grupos básicos de tipos en C.

- Enteros
- De punto flotante
- Carácter

Enteros:

El tamaño de los tipo entero depende de la máquina.
Los enteros signados son:

- short [int]
- int
- long [int]

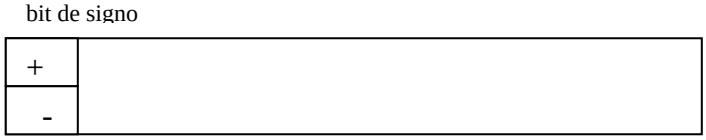
Los enteros sin signo son:

- unsigned short [int]
- unsigned int
- unsigned long [int]

Caracter:

Los tipo carácter son:

- char



desde –128 hasta 127

Los de carácter sin signo son:

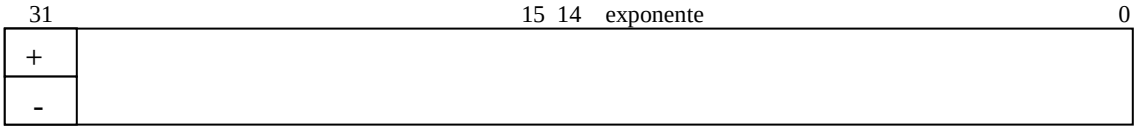
- [unsigned] char

desde 0 hasta 255

De punto flotante:

Los tipos de punto flotante son:

- float (precisión normal)
- double (doble precisión)
- long double (precision extendida)



bit de signo bit de signo (exponente)

Tamaño y rango de los tipos de datos básicos en C:

Tipo	Ancho en bit	Rango	
Char	8	-128	127
unsigned char	8	0	255
signed char	8	-128	127
Int	16	-32,768	32,767
unsigned int	16	0	65,535
short int	16	-32,768	32,767
signed short int	16	-32,768	32,767
unsigned short int	16	0	65,535
long int	32	-2,147´483,648	2,147´483,647
signed long int	32	-2,147´483,648	2,147´483,647
unsigned long int	32	0	4,294´967,295
Float	32	3.4 E -38	3.4 E 38
double	64	1.7 E -308	1.7 E 308
long double	64	1.7 E -308	1.7 E 308

Constantes:

Enteras:

- Decimal; 12, 126 (%d)
- Octal; 007, 057 (%D)
- Hexadecimal OXA95. 0XFF23 (%X)

De punto flotante:
Pueden ser escritas como:

- 0.0034 (%f)
- 12.5 (%f)
- 3 e –5 (%e)
- 3 E –5 (%e)

De carácter:

- Se almacena el valor numérico del carácter (%c).
- Pueden ser utilizados en expresiones numéricas. c= ´1´;
- Se escriben como: ´a´, ´+´, ´1´ (para definir una constante de cadena).
- Algunos caracteres se representan con mas de un carácter: \n, \t, \b.
- También se pueden representar como; ´\O33´ , ´\ Oxf ´.

```
#include <cstdlib>
#include <iostream>
/*Programa No 3  version 1*/
/*Programa que determina el tamaño de los tipos basicos*/
using namespace std;
int main()
{
   system("cls");
   printf("\n\n\t\tPrograma No 3  version 1\n");
   printf("\n\n\t\tPrograma que determina el tamaño de los tipos basicos\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 3  version 1\n");
   printf("\n\n\t\tEl tipo char ocupa %d bytes",sizeof(char));
   printf("\n\n\t\tEl tipo long ocupa %d bytes",sizeof(long));
   printf("\n\n\t\tEl tipo int ocupa %d bytes",sizeof(int));
   printf("\n\n\t\tEl tipo short ocupa %d bytes",sizeof(short));
   printf("\n\n\t\tEl tipo float ocupa %d bytes",sizeof(float));
   printf("\n\n\t\tEl tipo double ocupa %d bytes\n\n\t\t",sizeof(double));
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Operadores:

Turbo C es rico en operadores incorporados. Un operador es un símbolo que le dice al compilador que realice manipulaciones matemáticas o lógicas específicas.
Los operadores aritméticos son (+), (-), (/) y (*) y funcionan de la misma manera que en la mayoría de las computadoras.
El operador módulo (%) da el resultado de una división entera.

```
#include <cstdlib>
#include <iostream>
/*Programa No 4  version 1
Programa que visualiza el uso del operador modulo*/
using namespace std;
int main()
{
   int a,b;
   system("cls");
   printf("\n\n\t\tPrograma No 4  version 1\n");
   printf("\n\n\t\tPrograma que visualiza el uso del operador modulo\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 4  version 1\n");
   printf("\n\n\t\tIntroduzca dos numeros que quiera dividir");
   printf("\n\n\n\t\tDame el primer numero ");
   scanf("%d",&a);
   printf("\n\n\t\tDame el segundo numero  ");
   scanf("%d",&b);
   printf("\n\n\t\tEl cociente es %d ",a/b);
   printf("\n\n\t\tEl residuo es %d  \n\n\t\t",a%b);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Resumiendo:

Operadores aritméticos.

Binario: + , - , / , * , % .

Unarios: + , - .

Precedencia: + , - (unarios) y de otros * , / , + , - .

Asociatividad: es de izquierda a derecha. Ejemplo:

```
10 + 4*3 + 5
10 + 12 + 5
22 + 5
27
```

Los operadores de relación, igualdad y lógicos:

Los operadores de relación son:

< menor	<= menor igual
> mayor	>= mayor igual

En los operadores <= y >= no se permiten espacios.

Los operadores de igualdad son:

¡= desigualdad	== igualdad

Los operadores lógicos son:

&& and	II or	¡ not

la asociatividad de los operadores también es de izquierda a derecha.

```
#include <cstdlib>
#include <iostream>
/*Programa No 5  version 1
Programa que visualiza el uso de los operadores racionales*/
using namespace std;
int main()
{
   int a,b;
   system("cls");
   printf("\n\n\t\tPrograma No 5  version 1\n");
   printf("\n\n\t\tPrograma que visualiza el uso de los operadores racionales\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 5  version 1\n");
   printf("\n\n\t\tIntroduzca un numero entero    ");
   scanf("%d",&a);
   printf("\n\n\t\tIntroduzca otro numero entero    ");
   scanf("%d",&b);
   printf("\n\n\t\t el numero %d < %d  es %d ",a,b,a<b);
   printf("\n\n\t\t el numero %d > %d  es %d ",a,b,a>b);
   printf("\n\n\t\t el numero %d <= %d  es %d ",a,b,a<=b);
   printf("\n\n\t\t el numero %d >= %d  es %d ",a,b,a>=b);
   printf("\n\n\t\t el numero %d != %d  es %d ",a,b,a!=b);
   printf("\n\n\t\t el numero %d == %d  es %d \n\n\t\t",a,b,a==b);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Operadores de incremento y decremento:

Estos operadores son: ++ , -- .

Pueden ser usados como prefijo o posfijo.

++X	X++
--X	X—

- ++X incrementa x antes de utilizar su valor.
- X++ incrementa x después de utilizar su valor.
- Se aplican únicamente a variables.

```
#include <cstdlib>
#include <iostream>
/*Programa No 6  version 1
Programa que visualiza el uso de los operadores de incremento y decremento*/
using namespace std;
int main(void)
{
   int a=0,b=0,c=0;
   system("cls");
   printf("\n\n\t\tPrograma No 6  version 1\n");
   printf("\n\n\t\tPrograma que visualiza el uso de los operadores de incremento y decremento\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 6  version 1\n");
   a= ++b + ++c;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   a= b++ + c++;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   a= b++ + ++c;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   a= --b + --c;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   a= b-- + c--;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   a= b-- + --c;
   printf("\n\n\t\t%d   %d   %d\t\t",a,b,c);/*que se imprime*/
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Operadores de asignación:

En turbo C el operador de asignación es el signo igual (=). Este operador asigna el valor de la derecha a la variable de la izquierda. La asignación es una expresión, que da como resultado el valor y tipo del operando izquierdo.

```
#include <cstdlib>
#include <iostream>
/*Programa No 7  version 1
Programa que visualiza el uso de los operadores de asignacion*/
using namespace std;
int main(int argc, char *argv[])
{
   int a,b;
   system("cls");
   printf("\n\n\t\tPrograma No 7  version 1\n");
   printf("\n\n\t\tPrograma que visualiza el uso de los operadores de asignacion\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 7  version 1\n");
   printf("\n\n\t\tIntroduzca un numero entero   ");
   scanf("%d",&a);
   printf("\n\n\t\tIntroduzca otro numero entero   ");
   scanf("%d",&b);
   a+=b;
   printf("\n\n\t\tEl resultado de a+=b es a=a+b es %d",a);
   a-=b;
   printf("\n\n\t\tEl resultado de a-=b es a=a-b es %d",a);
   a*=b+5;
   printf("\n\n\t\tEl resultado de a*=b+5 es a=a*(b+5) es %d",a);
   a-=b;
   printf("\n\n\t\tEl resultado de a-=b es a=a-b es %d\n\n\t\t",a);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Operadores de bits:

Operadores binarios lógicos de bits:
Estos operadores operan de bit en bit, y los operadores lógicos son:
& (and), l (or) y ^ (xor). La siguiente tabla muestra su comportamiento.

X	Y	X&Y	X^Y	XlY
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	1

Ejemplo: Asumiendo que se tiene una representación de enteros de dos bytes.

```
N=34;  /* 0000 0000 0010 0010  */
X=16;  /* 0000 0000 0001 0000  */
C=N&X;        /* 0000 0000 0000 0000  */
```

Operador de complemento a uno:
El operador de complemento a uno (~), es un operador unario. Su comportamiento se muestra en la siguiente tabla.

X	~X
0	1
1	0

Ejemplo: Asumiendo que se tiene una representación de enteros de dos bytes.

```
N= 499;  /* 0000 0001 1111 0011  */
X=16;    /* 0000 0000 0001 0000  */
Y=~N     /* 1111 1110 0000 1100   */
Z=~X       /* 1111 1111 1110 1111  */
```

Operador de corrimiento de bits:
Los operadores de corrimientos de bits son binarios y son:
>> y <<.

En el caso de <<, se desplazará a la izquierda n bits indicados por el operador de la derecha en el operador de la izquierda.
Los bits en exceso son descartados.
Se colocan bits cero (0) en la derecha.
Ejemplo: Asumiendo que se tiene una representación de enteros de dos bytes.

```
n=16;    /* 0000 0000 0001 0000  */
c=n<<3;          /* 0000 0000 1000 0000  */   /*128*/
```

En el caso >>, se desplazará a la derecha n bits indicados por el operador de la derecha en el operador de la izquierda.
Los bits en exceso son descartados.
Se colocan bits cero (0) en la derecha.
Ejemplo: Asumiendo que se tiene una representación de enteros de dos bytes.

```
n=16;    /* 0000 0000 0001 0000  */
c=n>>3;          /* 0000 0000 0000 0010  */   /*2*/
```

Entrada y salida de datos:

Como se ha visto en los programas anteriores, la salida de datos se da con la función printf();
Su formato es el siguiente:

```
printf( cadena de control, lista de argumentos);
```

donde:

Cadena de control: es una cadena con los códigos que controlarán la forma como se desplegarán los resultados en el dispositivo de salida.
Ejemplo: printf("%d centimetros son %f pulgadas",cm,in);
Lista de argumentos: Es la lista con las variables o las expresiones que serán desplegadas.
La entrada de datos también se ha visto se da con la función scanf();
Su formato es el siguiente:

```
scanf( cadena de control, lista de argumentos);
```

donde:

Cadena de control: es una cadena con los códigos que controlarán la forma como se recibirán los datos en el dispositivo de entrada.
Lista de argumentos: Es la lista con las direcciones de las variables que serán leídas.
Ejemplo: scanf("%f",&total);

Los caracteres y cadenas se usa cuando se requiere de variables que no contengan números, sino letras y símbolos. A esta variable se le conoce como variable carácter. La forma general para definir una variable carácter es:

char lista de variables;
Ejemplo: char letra;

Lenguaje C ofrece un código de barras invertidas:

Código	Significado
\b	Espacio atrás
\f	Salto de página
\n	Salto de línea
\T	Salto de carro
\t	Tabulador
\”	Comillas
\´	Apóstrofo
\0	Nulo
\\	Barra invertida
\v	Tabulador vertical
\a	Alerta
\N	Constante octal
\XN	Constante hexadecimal

Código	Formato
%c	Carácter
%d	Decimal con signo
%i	Decimal con signo
%e	Notación científica
%E	Notación científica
%f	Coma flotante
%g	Usar %e o %f
%G	Usar %E o %F
%o	Octal sin signo
%s	Cadena de caracteres
%u	Enteros decimales sin signo
%x	Hexadecimal sin signo
%X	Hexadecimal sin signo
%p	Mostrar puntero
%n	El argumento asociado en un puntero a entero al que se asigna el número de caracteres escritos.
%%	Inprimir signo %
%[ ]	Muestra un conjunto de caracteres

Las funciones getch(); y getche(); . Estas dos funciones leen un solo caracter del teclado y son mas recomendables para entornes interactivos. La función getch(); no hace eco en la pantalla del caracter leído, es decir, no se visualiza en pantalla. La función getche(); si visualiza el caracter en pantalla.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 8  version 1
Programa que visualiza el uso de los caracteres*/
using namespace std;
int main(void)
{
   char letra;
  system("cls");
   printf("\n\n\t\tPrograma No 8  version 1\n");
   printf("\n\n\t\tPrograma que visualiza el uso de los caracteres\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 8  version 1\n");
   printf("\n\n\t\tPulse una letra para continuar  ");
   letra=getch();
   printf("\n\n\t\tNo se desplego ningun caracter en pantalla  ");
   printf("\n\n\t\tAhora pulse otra letra para continuar   ");
   letra=getche();
   printf("\n\n\t\tSe desplego el carscter %c que tu pulsaste\n\n\t\t",letra);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Cadena; conjunto de caracteres unidos como un solo objeto. A estas variables se le conoce como variables string (stong = cadena). El apellido de una persona, su domicilio y su ocupación son ejemplos de cadena de caracteres.
Una variable de cadena debe declararse como carácter e indicando entre corchetes la longitud máxima que tendrá dicha cadena.

La lectura de una cadena se realiza por medio de scanf ( ); y el indicador de campo es %s ó bien con la función gets ( ); La función scanf ( ) leerá una cadena sin blancos. La función gets ( ); leerá la cadena hasta alcanzar el retorno del carro.

```
#include <cstdlib>
#include <iostream>
/*Programa No 9  version 1
Programa que visualiza las lecturas de las variables del tipo caracter\*/
using namespace std;
int main(void)
{
   char nombre[10];
   char apaterno[10];
   char amaterno[10];
   system("cls");
   printf("\n\n\t\tPrograma No 9  version 1\n");
   printf("\n\n\tPrograma que visualiza las lecturas de las variables del tipo caracter\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 9  version 1\n");
   printf("\n\n\t\tCual es tu nombre?  ");
   gets(nombre);
   printf("\n\n\t\tCual es tu apellido paterno?  ");
   gets(apaterno);
   printf("\n\n\t\tCual es tu apellido materno?  ");
   gets(amaterno);
   printf("\n\n\t\tTu nombre completo es %s  %s  %s  ",nombre,apaterno,amaterno);
   printf("\n\n\t\tTus iniciales so: %c   %c   %c  \n\n\t\t",nombre[0],apaterno[0],amaterno[0]);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
/*Programa No 9  version 2
Programa que visualiza las lecturas de las variables del tipo caracter\*/
using namespace std;
int main(void)
{
   char nombre[10];
   char apaterno[10];
   char amaterno[10];
   system("cls");
   printf("\n\n\t\tPrograma No 9  version 2\n");
   printf("\n\n\tPrograma que visualiza las lecturas de las variables del tipo caracter\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 9  version 2\n");
   printf("\n\n\t\tCual es tu nombre?  ");
   scanf("%s",&nombre);
   printf("\n\n\t\tCual es tu apellido paterno?  ");
   scanf("%s",&apaterno);
   printf("\n\n\t\tCual es tu apellido materno?  ");
   scanf("%s",&amaterno);
   printf("\n\n\t\tTu nombre completo es %s  %s  %s  ",nombre,apaterno,amaterno);
   printf("\n\n\t\tTus iniciales son: %c   %c   %c  \n\n\t\t",nombre[0],apaterno[0],amaterno[0]);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Constantes: Una constante es un valor que no cambia a lo largo de la ejecución del programa. Cuando un valor es declarado como constante, no será posible realizar modificaciones sobre el.
Ejemplo:

Pi= 3.1416
IVA = 0-10

Un programador que desee utilizar frecuentemente una constante, como el IVA, necesita definirlo una solo vez y después usar su nombre simbólicos. Si por alguna razón este valor cambia, se tendrá que modificar una solo línea y no en todos los lugares donde se haga referencia a este valor.
El estándar ANSI C define a una constante a través de la palabra const de la siguiente forma.

const tipo nombre de variable = constante;
Ejemplo:

const IVA = 0.1.;

```
#include <cstdlib>
#include <iostream>
/*programa numero 1 version 2*/
/*programa que determina el area de un circulo*/
using namespace std;
const float pi=3.1416;
int main()
{
   system("cls");
   float radio, area;
   printf("\n\n\t\tPrograma No 1  version 2\n");
   printf("\n\n\t\tPrograma que determina el area de un circulo\n");
   printf("\n\n\t\tDame el valor de tu radio    ");
   scanf("%f",&radio);
   area=pi*radio*radio;
   printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
/*programa numero 1 version 3*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
int main()
{
   system("cls");
   float radio, area;
   printf("\n\n\t\tPrograma No 1  version 3\n");
   printf("\n\n\t\tPrograma que determina el area de un circulo\n");
   printf("\n\n\t\tDame el valor de tu radio    ");
   scanf("%f",&radio);
   area=pi*radio*radio;
   printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
/*programa numero 1 version 4*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
#define inicio int main()
#define principio {
#define fin  }
#define real  float
#define borrar    system("cls")
#define escribir   printf
#define ingresa  scanf
#define retener   system("PAUSE")
#define retornar    return EXIT_SUCCESS
inicio
principio
borrar;
real radio,area;
   escribir("\n\n\t\tPrograma No 1  version 4\n");
   escribir("\n\n\t\tPrograma que determina el area de un circulo\n");
   escribir("\n\n\t\tDame el valor de tu radio    ");
   ingresa("%f",&radio);
   area=pi*radio*radio;
   escribir("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   retener;
   retornar;
fin
```

```
#include <cstdlib>
#include <iostream>
//programa numero 1 version 5
//programa que determina el area de un circulo en un ambiente c++
using namespace std;
const float pi=3.1416;
int main()
{
   system("cls");
   float radio,area;
   cout << "\n\n\t\t" << "Programa No 1  version 5\n";
   cout << "\n\n\t\t" <<"Programa que determina el area de un circulo\n";
   cout << "\n\n\t\t" << "En un ambiente de c++\n";
   cout << "\n\n\t\t" << "Dame el valor de tu radio    ";
   cin>>radio;
   area=pi*radio*radio;
   cout<<"\n\n\t\t" << "El area de tu circulo es "<<area<<endl;
   cout<<"\n\n\t\t";
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

```
#include <cstdlib>
#include <iostream>
#include <math.h>
/*programa numero 1 version 6*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
int main()
{
   system("cls");
   float radio, area;
   printf("\n\n\t\tPrograma No 1  version 6\n");
   printf("\n\n\t\tPrograma que determina el area de un circulo\n");
   printf("\n\n\t\tDame el valor de tu radio    ");
   scanf("%f",&radio);
   area=pi*pow(radio,2);
   printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

ALGORITMOS Y ESTRUCTURAS DE CONTROL

PRIMER ALGORITM0

```
#include <cstdlib b >
#include <iostream>
/*Programa No 1  version 1
Programa que calcul el sueldo de un empleado*/
using namespace std;
int main(void)
{
   char nombre[10];
   char apaterno[10];
   char amaterno[10];
   int hrstrab;
   float cuotahr,sueldo;
   system("cls");
   printf("\n\n\t\tPrograma No 1  version 1 de los algoritmos\n");
   printf("\n\n\t\tPrograma que determina el sueldo de un empleado\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 1  version 1 de los algoritmos");
   printf("\n\n\t\tCual es tu nombre?  ");
   gets(nombre);
   printf("\n\n\t\tCual es tu apellido paterno?  ");
   gets(apaterno);
   printf("\n\n\t\tCual es tu apellido materno?  ");
   gets(amaterno);
   printf("\n\n\t\tCuantas horas trabajaste   ");
   scanf("%d",&hrstrab);
   printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
   scanf("%f",&cuotahr);printf("\n\n\t\t");
   sueldo=hrstrab*cuotahr;
   system("PAUSE");system("cls");
   printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
   printf("\n\n\t\t %s  %s  %s            $ %f \n\n\n\n\t\t",nombre,apaterno,amaterno,sueldo);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Estructura if-else:

Las decisiones que una computadora es capaz de tomar vienen dadas por las expresiones de relación y de expresión. Una decisión se realiza inquiriendo por el valor que una expresión posee. Si la expresión es evaluada con un valor de verdadero entonces se ejecuta una sentencia o un bloque de ellas. En caso de que una expresión resulte falsa pueden suceder tres casos:

- Se continua la ejecución del programa sin ejecutar ninguna de las instrucciones contenidas en el bloque verdadero de if.
- Se ejecuta una sentencia correspondiente a un bloque falso de la decisión.
- Se ejecuta un bloque de instrucciones.

La sintaxi es la siguiente:

```
if (expresión)
        sentencia (verdadera)
else
        sentencia (falso)
```

La parte else es opcional.

En construcciones unidas, la parte else termina el if mas interno, el compilador no toma en cuenta el sangrado.

Una decisión múltiple puede implementarse con una serie de if anidados. Sin embargo, al sangrar cada una de las sentencias provocaría que el tamaño de línea creciera demasiado, para ello se crea una construcción como la siguiente:

```
if (expresión)
      sentencia
else if (expresión)
       sentencia
else if (expresión)
       sentencia
else if (expresión)
       sentencia
else
       sentencia
```

En la construcción anterior las expresiones se evalúan en orden, cuando una de ellas es verdadera, la sentencia asociada se ejecuta y con esto termina la instrucción, ha ultima sentencia se ejecuta cuando ninguna expresión es verdadera.
Ejemplo:

```
#include <cstdlib>
#include <iostream>
/*Programa No 10  version 1
Este programa hace la conversión de diferentes bases numericas*/
using namespace std;
int main(void)
{
  int opcion;
  int valor;
  system("cls");
  printf("\n\n\t\tPrograma No 10  version 1\n");
  printf("\n\n\t\tEste programa hace la conversión de diferentes bases numericas\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que hace conversión de diferentes bases ");
  printf("\n\n\t\t1: decimal a hexadecimal \n");
  printf("\n\n\t\t2: hexadecimal a decimal \n");
  printf("\n\n\t\t3: decimal a octal \n");
  printf("\n\n\t\t4: octal a decimal \n");
  printf("\n\n\t\tintroduzca su opción      ");
  scanf("%d",&opcion);
  system("cls");
  if (opcion==1)
   {
            printf ("\n\n\t\t Introduzca un valor decimal:  ");
            scanf ("%d",&valor);
            printf ("\n\n\t\t%d en hexadecimal es: %x\n\n\t\t",valor,valor);
   }
  if (opcion==2)
   {
            printf("\n\n\t\t Introduzca un valor hexadecimal:  ");
            scanf("%x",&valor);
            printf("\n\n\t\t%x en decimal es: %d\n\n\t\t",valor,valor);
   }
  if (opcion==3)
   {
            printf("\n\n\t\t Introduzca un valor decimal:  ");
            scanf("%d",&valor);
            printf("\n\n\t\t%d en octal es: %o\n\n\t\t",valor,valor);
   }
  if (opcion==4)
   {
   printf("\n\n\t\t introduzca un valor octal:  ");
   scanf("%o",valor);
   printf("\n\n\t\t %o en decimal es %d\n\n\t\t",valor,valor);
   }
  system("PAUSE");
  return EXIT_SUCCESS;
}
```

SEGUNDO ALGORITMO

```
#include <cstdlib>
#include <iostream>
/*Programa No 2  version 1
Programa que calcul el sueldo de un empleado*/
using namespace std;
int main(void)
{
   char nombre[10];
   char apaterno[10];
   char amaterno[10];
   int hrstrab;
   float cuotahr,sueldo;
   system("cls");
   printf("\n\n\t\tPrograma No 2  version 1 de los algoritmos\n");
   printf("\n\n\t\tPrograma que determina el sueldo de un empleado\n\n\t\t");
```

```
    system("PAUSE");system("cls");
    printf("\n\n\t\tPrograma No 2  version 1 de los algoritmos");
    printf("\n\n\t\tCual es tu nombre?  ");
    gets(nombre);
    printf("\n\n\t\tCual es tu apellido paterno?  ");
    gets(apaterno);
    printf("\n\n\t\tCual es tu apellido materno?  ");
    gets(amaterno);
    printf("\n\n\t\tCuantas horas trabajaste   ");
    scanf("%d",&hrstrab);
    printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
    scanf("%f",&cuotahr); printf("\n\n\t\t");
    if (hrstrab>40)
      {
       sueldo=(hrstrab-40)*(cuotahr*2)+(cuotahr*40);
      }
     else
      {
       sueldo=hrstrab*cuotahr;
      }
    system("PAUSE");system("cls");
    printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
    printf("\n\n\t\t %s  %s  %s            $ %f \n\n\n\n\t\t",nombre,apaterno,amaterno,sueldo);
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Operador condicional:
Permite la implementación de una expresión condicional en una sola línea. Su sintaxis es la siguiente:

expresión1?expresión2:expresión3

En una expresión condicional

a) Primero se eleva la expresión 1
b) Si la expresión 1 es verdadera, se eleva la expresión 2
c) Si la expresión 1 es falsa, se eleva la expresión 3

```
#include <cstdlib>
#include <iostream>
/*Programa No 11  version 1
Programa que muestra el uso del operador condiciónal*/
using namespace std;
int main(void)
{
  int x,y;
  system("cls");
  printf("\n\n\t\tPrograma No 11  version 1\n");
  printf("\n\n\t\tPrograma que muestra el uso del operador condiciónal\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame el valor de un número entero   ");
  scanf("%d",&x);
  printf("\n\n\t\tdame el valor de un segundo número entero para compararlos    ");
  scanf("%d",&y);
  printf("\n\n\t\t%d es el número mayor entre %d y %d \n\n\n\t\t",((x>y)?x:y),x,y);
  system("PAUSE");
  return x,y;
}
```

```
#include <cstdlib>
#include <iostream>
/*Programa No 11  version 2
Programa que sustituye el uso del operador condiciónal*/
using namespace std;
int main(void)
{
  int x,y;
  system("cls");
  printf("\n\n\t\tPrograma No 11  version 2\n");
  printf("\n\n\t\tPrograma que sustituye el uso del operador condiciónal\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame el valor de un número entero   ");
  scanf("%d",&x);
  printf("\n\n\t\tdame el valor de un segundo número entero para compararlos    ");
  scanf("%d",&y);
  if(x>y)
  {
   printf("\n\n\t\t%d es el número mayor entre %d y %d \n\n\n\t\t",x,x,y);
  }
  else
  {
    printf("\n\n\t\t%d es el número mayor entre %d y %d \n\n\n\t\t",y,x,y);
  }
  system("PAUSE");
  return x,y;
}
```

Estructura go to:

No entra dentro de la programación estructurada, nos sirve para dar una salida emergente de nuestro programa.
La sintaxi es la siguiente:

```
goto etiquetas;
```

La etiqueta es un identificador que se utiliza para señalar la sentencia a la que se transferirá el control. Se puede transferir el control a cualquier otra sentencia del programa. La sentencia por la que se continuará la ejecución deberá encontrarse etiquetada, y la etiqueta debe encontrarse seguida de dos puntos (:), por lo tanto, la sentencia etiquetada aparecerá de la siguiente forma: etiqueta: sentencia.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 12  version 1
Programa que muestra el uso de la estructura goto*/
using namespace std;
int main(void)
{
  int a;
  system("cls");
  printf("\n\n\t\tPrograma No 12  version 1\n");
  printf("\n\n\t\tPrograma que muestra el uso de la estructura goto\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame un numero entre el cero y el tres  ");
  scanf("%d",&a);
   if (a==0) goto cero;
     else if (a==1) goto uno;
     else if (a==2) goto dos;
     else if (a==3) goto tres;
   else
       printf ("\n\n\t\t\a no diste un número entre cero y tres \a   ");getche();exit(1);
  cero:
      printf ("\n\n\t\t tecleaste un cero");getche();exit(1);
  uno:
      printf ("\n\n\t\t tecleaste un uno");getche();exit(1);
  dos:
      printf ("\n\n\t\t tecleaste un dos");getche();exit(1);
  tres:
      printf ("\n\n\t\t tecleaste un tres");getche();exit(1);
  printf ("\n\n\t\t ");
  system("PAUSE");
 return a;
}    /*Exit.- provoca una salida inmediata*/
```

TERCER ALGORITMO

```
#include <cstdlib>
#include <iostream>
/*Programa No 3  del los algoritmos
Programa que calcula el sueldo de un empleado*/
using namespace std;
int main(void)
{
   char nombre[10];
   char apaterno[10];
   char amaterno[10];
   int hrstrab;
   float cuotahr,sueldo;
   system("cls");
   printf("\n\n\t\tPrograma No 2  version 2 de los algoritmos\n");
   printf("\n\n\t\tPrograma que determina el sueldo de un empleado\n\n\t\t");
   system("PAUSE");system("cls");
   printf("\n\n\t\tPrograma No 2  version 2 de los algoritmos");
   printf("\n\n\t\tCual es tu nombre?  ");
   gets(nombre);
   printf("\n\n\t\tCual es tu apellido paterno?  ");
   gets(apaterno);
   printf("\n\n\t\tCual es tu apellido materno?  ");
   gets(amaterno);
   printf("\n\n\t\tCuantas horas trabajaste   ");
   scanf("%d",&hrstrab);
   printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
   scanf("%f",&cuotahr); printf("\n\n\t\t");
   if (hrstrab>40)
     {
      sueldo=(hrstrab*cuotahr)*(1.05);
     }
    else
     {
       printf("\n\n\t\t\a\aNO TRABAJASTE MAS DE 4O HORAS \n\n\t\t\a\a ");
      }
   system("PAUSE");system("cls");
   printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
   printf("\n\n\t\t %s  %s  %s            $ %f \n\n\n\n\t\t",nombre,apaterno,amaterno,sueldo);
   system("PAUSE");
   return EXIT_SUCCESS;
}
```

Estructura switch:
Permite la implementación de decisiones múltiples con valores enteros.

Su sintaxi es la siguiente:

```
switch (expresión)
{
case exp-const1; sentencia
case exp-const2; sentencia
default: sentencia;
}
```

donde: exp-const = expresión constante entera.

La expresión se evalúa y el resultado se compara con las expresiones constantes, si alguna de ellas coincide el control del programa se traslada a ese punto. Las expresiones constantes deben ser enteras y no se deben repetir. Las sentencias después de la expresión constante no se necesitan agrupar como bloque. La cláusulas default es opcional e indica el lugar a donde se traslada el control del programa en el caso que ninguna de las etiquetas case coincidan con el valor de la expresión.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 12  version 2
Programa que muestra el uso de la estructura switch-case*/
using namespace std;
int main(void)
{
  int a;
  system("cls");
  printf("\n\n\t\tPrograma No 12  version 2");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame un numero entre el cero y el tres  ");
  scanf("%d",&a);
  switch(a)
    {
       case 0: printf ("\n\n\t\t\a tecleaste un cero:"); break;
       case 1: printf ("\n\n\t\t\a tecleaste un uno:"); break;
       case 2: printf ("\n\n\t\t\a tecleaste un dos:"); break;
       case 3: printf ("\n\n\t\t\a tecleaste un tres:"); break;
       default: printf ("\n\n\t\t\a No diste un número entre cero y tres \a ");break;
    }
   printf ("\n\n\t\t ");
   system("PAUSE");
  return a;
}
```

La proposición break provoca una salida inmediata del switch; for do y while.

CUARTO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de algoritmo 4
Programa que muestra el uso de la estructura switch-case*/
using namespace std;
int main(void)
{
  int numdia;
  system("cls");
  printf("\n\n\t\tPrograma No 4 version 1 de los algoritmos");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame el numero de dia de la SEMANA  ");
  scanf("%d",&numdia);
  switch(numdia)
    {
       case 1: printf ("\n\n\t\t\a HOY ES DOMINGO"); break;
       case 2: printf ("\n\n\t\t\a HOY ES LUNES"); break;
       case 3: printf ("\n\n\t\t\a HOY ES MARTES"); break;
       case 4: printf ("\n\n\t\t\a HOY ES MIERCOLES"); break;
       case 5: printf ("\n\n\t\t\a HOY ES JUVES"); break;
       case 6: printf ("\n\n\t\t\a HOY ES VIERNES"); break;
       case 7: printf ("\n\n\t\t\a HOY ES SABADO"); break;
       default: printf ("\n\n\t\t\a NO SABES EN QUE DIA ESTAS \a ");break;
    }
   printf ("\n\n\t\t ");
   system("PAUSE");
  return numdia;
}
```

Tambien se puede hacer con estructura goto.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de algoritmo 4 version 2
Programa que muestra el uso de la estructura goto*/
using namespace std;
int main(void)
{
  int numdia;
  system("cls");
  printf("\n\n\t\tPrograma No 4 version 2 de los algoritmos");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tdame el numero de dia de la SEMANA  ");
  scanf("%d",&numdia);
      if (numdia==1) goto uno;
      else if (numdia==2) goto dos;
      else if (numdia==3) goto tres;
      else if (numdia==4) goto cuatro;
      else if (numdia==5) goto cinco;
      else if (numdia==6) goto seis;
      else if (numdia==7) goto siete;
   else
       printf ("\n\n\t\t\a  NO SABES EN QUE DIA ESTAS \a\a   ");getche();exit(1);
  uno:
       printf ("\n\n\t\t\a HOY ES DOMINGO");getche();exit(1);
  dos:
       printf ("\n\n\t\t\a HOY ES LUNES");getche();exit(1);
  tres:
       printf ("\n\n\t\t\a HOY ES MARTES");getche();exit(1);
  cuatro:
       printf ("\n\n\t\t\a HOY ES MIERCOLES");getche();exit(1);
  cinco:
       printf ("\n\n\t\t\a HOY ES JUEVES");getche();exit(1);
  seis:
       printf ("\n\n\t\t\a HOY ES VIERNES");getche();exit(1);
  siete:
       printf ("\n\n\t\t\a HOY ES SABADO");getche();exit(1);
  printf ("\n\n\t\t ");
  system("PAUSE");
  return numdia;
}
```

La estructura de control REPEAT no se ncuentra dentro del Lenguaje C pero la podemos sustituir por un goto.

```
#include <cstdlib>
#include <iostream>
#include <math.h>
#include <conio.h>
/*programa numero 1 version 7*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
int main()
{
   float radio, area;
   char a;
comienza:
   system("cls");
   printf("\n\n\t\tPrograma No 1  version 7\n");
   printf("\n\n\t\tPrograma que determina el area de un circulo\n");
   printf("\n\n\t\tDame el valor de tu radio    ");
   scanf("%f",&radio);
   area=pi*pow(radio,2);
   printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
   printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
     scanf("%c",&a);a=toupper(getche());getche();
```

```
        if(a=='s'||a=='S')
        {
                goto comienza;
        }
        else
        {
                goto fin;
        }
    fin:;
       return EXIT_SUCCESS;
    }
```

QUINTO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de algoritmo 5
Programa que muestra el uso de la estructura goto(repeat)*/
using namespace std;
int main(void)
{
  char nombre[10];
  char apaterno[10];
  char amaterno[10];
  int hrstrab;
  float cuotahr,sueldo;
  char desea;
repetir:
  system("cls");
  printf("\n\n\t\tPrograma No 5 version 1 de los algoritmos");
  printf("\n\n\t\tUsando la estructura goto sustituyendo a la estructura repeat\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que determina el sueldo de un empleado");
  printf("\n\n\t\tPrograma No 5  version 1 de los algoritmos");
  printf("\n\n\t\tCual es tu nombre?  ");
  gets(nombre);
  printf("\n\n\t\tCual es tu apellido paterno?  ");
  gets(apaterno);
  printf("\n\n\t\tCual es tu apellido materno?  ");
  gets(amaterno);
  printf("\n\n\t\tCuantas horas trabajaste   ");
  scanf("%d",&hrstrab);
  printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
  scanf("%f",&cuotahr); printf("\n\n\t\t");
  sueldo=(hrstrab*cuotahr);
  system("PAUSE");system("cls");
  printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
  printf("\n\n\t\t %s  %s  %s            $ %f \n\n\t\t",nombre,apaterno,amaterno,sueldo);
  system("PAUSE");
  printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
  scanf("%c",&desea);desea=toupper(getche());getche();
        if(desea=='s'||desea=='S')
        {
                goto repetir;
        }
        else
        {
                goto fin;
        }
fin:;
   return EXIT_SUCCESS;
}
```

Estructura for:
Esta es una estructura de iteración que se ejecutara de cero o más veces dependiendo de la expresión. La sintaxi es la siguiente:

for (expresión 1; expresión 2; expresión 3)

Sentencia

La secuencia de ejecución es la siguiente:
1.- Se ejecuta la expresión 1.
2.- Se evalúa la expresión 2.
Si la evaluación es verdadera, termina el for.
Si la evaluación es falso, se continúa al paso 3.
3.- Se ejecuta la sentencia.
4.- Se evalúa la expresión 3.
5.- Se regresa al paso 2.

Cualquiera de las expresiones se puede omitir.

Pero si se omite la segunda expresión, se hace un ciclo infinito.
Ejemplo:
for (i=o; ;i++) = > ciclo infinito ; cuidado!

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 13  version 1
Programa que muestra el uso de la estructura for*/
using namespace std;
int main(void)
{
  int factor_1,factor_2,producto;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 13  version 1");
  printf("\n\n\t\tPrograma que muestra el uso de la estructura for");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que despliega tablas de multiplicar del 1 al 10");
  printf("\n\n\t\t tablas de multiplicar");
    for  (factor_1=1; factor_1<=9;factor_1++)
      {
        system("cls");
       printf("\n\n\n\t\t\a\a\a Tabla de %d multiplicar\n\n\n\t\t",factor_1);
        for  (factor_2=1;factor_2<=10;factor_2++)
        {
          producto=factor_1*factor_2;
          printf("\n\t %5d   *  %5d  =   %5d", factor_1,factor_2,producto);
            getche();
        }
        printf("\n\n\t\t\a\aPULSE CUALQUIER TECLA PARA CONTINUAR");
        espera=getch();
       }
  printf("\n\n\t Fin de las tablas de multiplicar ");
  printf ("\n\n\t\t ");
  system("PAUSE");
  return 0;
}
```

SEPTIMO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 7  version 1 de los algoritmos
Programa que muestra el uso de la estructura for*/
using namespace std;
int main(void)
{
  int indice,sumatoria=0;
```

```
  system("cls");
  printf("\n\n\t\tPrograma No 7  version 1 de los algoritmos");
  printf("\n\n\t\tPrograma que muestra el uso de la estructura for\n\n\t\t");
  printf("\n\nPrograma que calcula e imprime la sumatoria de los numeros del 1 hasta el 100");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tSUMATORIA DE NUMEROS DEL 1 HASTA EL 100\n\n\t\t");
    for  (indice=1;indice<=100;indice++)
      {
                system("cls");
                sumatoria++;
        printf("\n\t\t Sumatoria= %d  ",sumatoria);
                getche();
       }
   printf("\n\n\t\tLa suma total del 1 al 100 es :  %d ",sumatoria);
   printf ("\n\n\t\t ");
   system("PAUSE");
   return sumatoria;
}
```

El programa anterior solo lleva el conteo de uno en uno hasta el 100. Si queremos que nos vaya sumando los acumulados el programa quedaria de la siguiente manera. En este caso para hacer la prueba de escritorio se puso como limite superior el numero 10.

SEPTIMO V-2 ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 7  version 2 de los algoritmos
Programa que muestra el uso de la estructura for*/
using namespace std;
int main(void)
{
  int indice,sumatoria=0;
  system("cls");
  printf("\n\n\t\tPrograma No 7  version 2 de los algoritmos");
  printf("\n\n\t\tPrograma que muestra el uso de la estructura for\n\n\t\t");
  printf("\n\nPrograma que calcula e imprime la suma de los numeros del 1 hasta el 100");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tSUMA DE NUMEROS DEL 1 HASTA EL 100\n\n\t\t");
    for  (indice=1;indice<=10;indice++)
      {
          system("cls");
          sumatoria=sumatoria+indice;
          printf("\n\t\t Sumatoria= %d  ",sumatoria);
          getche();
       }
   printf("\n\n\t\tLa suma total del 1 al 100 es :  %d ",sumatoria);
   printf ("\n\n\t\t ");
   system("PAUSE");
   return sumatoria;
}
```

Estructura while:

En esta estructura se evalúa primero una condición y si resulta verdadera entonces se ejecuta la o las instrucciones que se encuentren a continuación de la condición. De lo contrario se ejecutara la siguiente instrucción que no forme parte del bloque de instrucciones de while.

La sintaxi es la siguiente:

```
while (expresión)
        sentencia
```

La sentencia se ejecuta mientras la evaluación de la expresión sea verdadera.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 13  version 2
Programa que muestra el uso de la estructura while*/
using namespace std;
int main(void)
{
  int factor_1,factor_2,producto;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 13  version 2");
  printf("\n\n\t\tPrograma que muestra el uso de la estructura while");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que despliega tablas de multiplicar del 1 al 10");
  printf("\n\n\t\t tablas de multiplicar");
  factor_1=0;
  while(factor_1<=9)
    {
     factor_1++;
     system("cls");
     printf("\n\n\n\t\t\a\a\a Tabla de %d multiplicar\n\n\n\t\t",factor_1);
              factor_2=0;
               while(factor_2<=9)
                {
                   factor_2++;
                   producto=factor_1*factor_2;
                   printf("\n\t %5d   *  %5d  =   %5d", factor_1,factor_2,producto);
                   getche();
                 }
               printf("\n\n\t\t\a\aPULSE CUALQUIER TECLA PARA CONTINUAR");
               espera=getch();
     }
  printf("\n\n\t\t\a\a FIN DE LAS TABLAS DE MULTIPLICAR\a\a");
  printf ("\n\n\t\t ");
  system("PAUSE");
  return 0;
}
```

Programa 1 version 8 con la estructura while.

```
#include <cstdlib>
#include <iostream>
#include <math.h>
#include <conio.h>
/*programa numero 1 version 8*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
int main()
{
   float radio, area;
   char desea;
   desea='s';
   while(desea!='N')
    {
     system("cls");
     printf("\n\n\t\tPrograma No 1  version 8\n");
     printf("\n\n\t\tPrograma que determina el area de un circulo\n");
     printf("\n\n\t\tDame el valor de tu radio    ");
     scanf("%f",&radio);
     area=pi*pow(radio,2);
     printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
       printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
     scanf("%c",&desea);
     /*system("PAUSE");*/
     desea=toupper(getche());
     }
   return 0;
}
```

OCTAVO ALGORITMO V.1

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de algoritmo 8 version 1
Programa que muestra el uso de la estructura WHILE sustituyendo a la estructura DO-WHILE*/
using namespace std;
int main(void)
{
  char nombre[10];
  char apaterno[10];
  char amaterno[10];
  int hrstrab;
  float cuotahr,sueldo;
  char desea;
  system("cls");
  printf("\n\n\t\tPrograma No 8 version 1 de los algoritmos");
  printf("\n\n\tUsando la estructura while sustituyendo a la estructura do-while\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que determina el sueldo de los empleados");
  printf("\n\n\t\tPrograma No 8  version 1 de los algoritmos");
  printf("\n\n\t\t\a\aExiste algun empleado a procesar S/N ?  ");
  scanf("%c",&desea);
  desea=toupper(getche());
  while(desea!='N')
   {
    system("cls");printf("\n\n\t\t");system("PAUSE");
    printf("\n\n\t\tCual es tu nombre?  ");
    scanf("%s",&nombre);
    printf("\n\n\t\tCual es tu apellido paterno?  ");
    scanf("%s",&apaterno);
    printf("\n\n\t\tCual es tu apellido materno?  ");
    scanf("%s",&amaterno);
    printf("\n\n\t\tCuantas horas trabajaste   ");
    scanf("%d",&hrstrab);
    printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
    scanf("%f",&cuotahr); printf("\n\n\t\t");
    sueldo=(hrstrab*cuotahr);
    system("PAUSE");system("cls");
    printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
    printf("\n\n\t\t %s  %s  %s            $ %f \n\n\t\t",nombre,apaterno,amaterno,sueldo);
    system("PAUSE");
            printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
    scanf("%c",&desea);
    desea=toupper(getche());
   }
  return 0;
}
```

Estructura do while:

Al contrario de los bucles for y while que comprueban la condición, el bucle do while siempre ejecuta al menos una vez después la condición.

Sintaxi

```
do
Sentencia
while (expresión)
```

La secuencia de ejecución es la siguiente:

a) se ejecuta la sentencia
b) se ejecuta la expresión

si la evaluación es falsa termina el ciclo
si la evaluación es verdadera se vuelve al paso 1.
Las llaves{ } se ponen para no confundir al programa

```
#include <cstdlib>
#include <iostream>
#include <math.h>
#include <conio.h>
/*programa numero 1 version 9*/
/*programa que determina el area de un circulo*/
using namespace std;
#define pi 3.1416
int main()
{
   float radio, area;
   char desea;
   desea='s';
   do
    {
     system("cls");
     printf("\n\n\t\tPrograma No 1  version 9\n");
     printf("\n\n\t\tPrograma que determina el area de un circulo\n");
     printf("\n\n\t\tDame el valor de tu radio    ");
     scanf("%f",&radio);
     area=pi*pow(radio,2);
     printf("\n\n\t\tEl area de tu circulo es %f    \n\n\n\t\t",area);
       printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
     scanf("%c",&desea);
     desea=toupper(getche());
    }
   while(desea!='N');
   return 0;
}
```

OCTAVO ALGORITMO V.2

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de algoritmo 8 VERSION 2
Programa que muestra el uso de la estructura DO-WHILE*/
using namespace std;
int main(void)
{
  char nombre[10];
  char apaterno[10];
  char amaterno[10];
  int hrstrab;
  float cuotahr,sueldo;
  char desea;
  system("cls");
  printf("\n\n\t\tPrograma No 8 version 2 de los algoritmos");
  printf("\n\n\t\tUsando la estructura do-while\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que determina el sueldo de los empleados");
  printf("\n\n\t\tPrograma No 8  version 2 de los algoritmos");
  printf("\n\n\t\t\a\aExiste algun empleado a procesar S/N ?  ");
  scanf("%c",&desea);
  desea=toupper(getche());
  do
   {
    system("cls");printf("\n\n\t\t");system("PAUSE");
    printf("\n\n\t\tCual es tu nombre?  ");
    scanf("%s",&nombre);
    printf("\n\n\t\tCual es tu apellido paterno?  ");
    scanf("%s",&apaterno);
    printf("\n\n\t\tCual es tu apellido materno?  ");
    scanf("%s",&amaterno);
    printf("\n\n\t\tCuantas horas trabajaste   ");
    scanf("%d",&hrstrab);
    printf("\n\n\t\tCual es tu cuota por hora trabajada  ");
    scanf("%f",&cuotahr); printf("\n\n\t\t");
    sueldo=(hrstrab*cuotahr);
    system("PAUSE");system("cls");
    printf("\n\n\t\t      NOMBRE DEL EMPLEADO                    SUELDO  ");
```

```
    printf("\n\n\t\t %s  %s  %s            $ %f \n\n\t\t",nombre,apaterno,amaterno,sueldo);
    system("PAUSE");
            printf("\n\n\t\t\a\aDESEA PROCESAR OTRO EMPLEADO S/N ?  ");
    scanf("%c",&desea);
    desea=toupper(getche());
   }
  while(desea!='N');
  return 0;
}
```

Otro programa.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa de 14
Programa que dibuja un triangulo*/
using namespace std;
int main(void)
{
  int renglon,columna,limite;
  char desea;
  system("cls");
  printf("\n\n\t\tPrograma No 14 version 1");
  printf("\n\n\tUsando la estructura while dibuja un triangulo\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tPrograma que dibuja un triangulo");
  printf("\n\n\t\tPrograma No 14  version 1");
  printf("\n\n\t\t\a\aDesea correr este programa S/N ?  ");
  scanf("%c",&desea);
  desea=toupper(getche());
   while(desea!='N')
    {
     system("cls");printf("\n\n\t\t");system("PAUSE");
     printf("\n\n\t\t Digite el limite de tu triangulo [0-20]");
     limite=-1;
        while(limite<0||limite>20)
          {
         scanf("%i",&limite);
           printf("\n\n\t\tTriangulo con %i renglones",limite);
           renglon=0;
             while(renglon<limite)
              {
                        renglon++;
              printf("\n");
                        columna=0;
                          while(columna<renglon)
                           {
                                    columna++;
                            printf("%i",columna);
                           }
               }
    }
    printf("\n\n\t\t\a\aDesea volver a dibujar otro triangulo S/N ?  ");
    scanf("%c",&desea);
    desea=toupper(getche());
   }
  return 0;
}
```

Arreglos:

Es una colección de variables del mismo tipo, con el mismo nombre, y que se diferencian unos a otras a través de un subíndice. La forma general para declarar un arreglo es:

tipo nombre de variables (número – de – elementos)

El compilador reserva una cantidad de espacios suficiente en la memoria para poder contenerlo. Todo el arreglo es consignado en localidades contiguas de memoria. El primer elemento de un arreglo no es el 1 sino el O. Es decir, si e declara un arreglo con 10 elementos, el último subíndice es el 9. Es importante validar los limites del arreglo, de lo contrario, de podría estar escribiendo en un dato útil. Lo peor es que podría sobrescribir en el código ejecutable y el programa terminaría produciendo resultados inesperados, ó inclusive provocando caídas en el sistema.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 15  version 1
Programa que muestra el uso de los arreglos unidimensionales con la estructura for*/
using namespace std;
int main(void)
{
  int cantidad[12];
  int contador;
  long int suma;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 15  version 1");
  printf("\n\n\t\tPrograma que muestra el uso de los arreglos unidimensionales");
  printf("\n\n\t\tusando la estructura de control for");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\t\t Acumulados mensuales ");
   for (contador=0;contador<=11;contador++)
            {
     printf("\n\t\t Digite la ventas del mes %2d;  ",contador+1);
     scanf("%d",&cantidad[contador]);
     }
     suma=0;system("cls");
     printf("\n\n\t\t       TABLA DE ACUMULADOS ");
     printf("\n\t\t   = = = = = = = = = = = = = = = =");
     printf ("\n\t\t Mes         Ventas   Acumulados ");
   for (contador=0;contador<=11;contador++)
            {
     suma=suma+cantidad[contador];
     printf("\n\t\t [%2d] %10d %10d",contador+1,cantidad[contador],suma);
    }
     printf("\n\n\t\t\a\aPULSE CUALQUIER TECLA PARA CONTINUAR");
              espera=getch();
              printf ("\n\n\t\t");
     return 0;
}
```

NOVENO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 9  version 1 de los algoritmos
Programa que muestra el uso de los arreglos unidimensionales con la estructura for*/
using namespace std;
int main(void)
{
  char nombre[10];
  char apaterno[10];
  char amaterno[10];
  int ventas[30];
  int i;
  float totventa;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 9  version 1 de los algoritmos");
  printf("\n\n\t\tPrograma que muestra el uso de los arreglos unidimensionales");
  printf("\n\n\t\tusando la estructura de control for");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  printf("\n\n\t\tCual es tu nombre?  ");
  scanf("%s",&nombre);
  printf("\n\n\t\tCual es tu apellido paterno?  ");
  scanf("%s",&apaterno);
  printf("\n\n\t\tCual es tu apellido materno?  ");
  scanf("%s",&amaterno);
  system("cls");
  for (i=0;i<=9;i++)
```

```
            {
      printf("\n\t\t Digite la venta del dia %2d;  ",i+1);
      scanf("%d",&ventas[i]);
      }
      totventa=0;system("cls");
      printf("\n\t\tNombre del vendedor: %s %s %s ",nombre,apaterno,amaterno );
      printf ("\n\n\n\t\t  Dia              Ventas  ");
   for (i=0;i<=9;i++)
            {
      totventa=totventa+ventas[i];
      printf("\n\t\t [%2d]        %10d ",i+1,ventas[i]);
     }
      printf("\n\n\t\t Total de la venta del mes %f ",totventa);
      printf("\n\n\t\tPULSE CUALQUIER TECLA PARA CONTINUAR");
            espera=getch();
            return 0;
}
```

NOTA: En el algoritmo se nos pide para 30 dias pero en este programa se hizo para 10 dias y asi poder hacer su prueba de escritorio.

DECIMO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 10  version 1 de los algoritmos
Programa que muestra el uso de los arreglos unidimensionales con la estructura for*/
using namespace std;
int main(void)
{
  float a[10],b[10],c[10];
  int i;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 10  version 1 de los algoritmos");
  printf("\n\nPrograma que lee dos arreglos unidimensionales y los suma en un tercer arreglo ");
  printf("\n\n\t\tusando la estructura de control for");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  system("cls");
  for (i=0;i<=9;i++)
      {
      printf("\n\t\t Digite el valor %d del primer arreglo ",i+1);
      scanf("%f",&a[i]);
      }
     system("cls");
   for (i=0;i<=9;i++)
       {
       printf("\n\t\t Digite el valor  %d del segundo arreglo ",i+1);
       scanf("%f",&b[i]);
       }
     system("cls");
     printf("\n\t\t Desplegando la informacion \n\n");
      printf("\n\t La suma de los arreglos a  +  b  =  c\n\n");
    for (i=0;i<=9;i++)
      {
      c[i]=a[i]+b[i];
      printf("\n\t\t %f  +  %f  =  %f  ",a[i],b[i],c[i]);
     }
      printf("\n\n\t\tPULSE CUALQUIER TECLA PARA CONTINUAR");
        espera=getch();
        return 0;
}
```

ONCEAVO ALGORITMO

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 11  version 1 de los algoritmos
Programa que muestra el uso de los arreglos bidimensionales con la estructura for*/
using namespace std;
int main(void)
{
  int matriz[4][4];
  int ren,col,suma;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 11  version 1 de los algoritmos");
  printf("\n\n\t\tPrograma que despliega una matriz de 4 x 4 ");
  printf("\n\n\t\tusando la estructura de control for");
  printf("\n\n\t\t");
  system("PAUSE");system("cls");
  for (ren=0;ren<=3;ren++)
              {
    for (col=0;col<=3;col++)
                {
         printf("\n\t\t Digite el valor de la matriz [ %d   %d ] ",ren+1,col+1);
         scanf("%d",&matriz[ren][col]);
        }
      }
    system("cls");
   printf("\n\t\t Despliegue de la matriz\n\n\n");
  for (ren=0;ren<=3;ren++)
              {
    for (col=0;col<=3;col++)
                {
         printf("%10d ",matriz[ren][col]);
        } printf("\n");
      }
  printf("\n\n\t\tPULSE CUALQUIER TECLA PARA CONTINUAR");
  espera=getch();
  return 0;
}
```

Un arreglo puede tener mas de dos dimensiones, de tal manera que forme matrices. Generalmente los arreglos no suelen recomendarse para más de tres dimensiones, ya que se vuelven muy difíciles de entender.

El primer índice representa los renglones y el segundo las columnas.

PODEMOS MANIPULAR EL PROGRAMA ANTERIOR PARA HACER UNA SUMA DE MATRICES

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
/*Programa No 11  version 2 de los algoritmos
Programa que muestra una suma de matrices con el uso de los
arreglos bidimensionales con la estructura for*/
using namespace std;
int main(void)
{
  int matriz1[3][3],matriz2[3][3],matriz3[3][3];
  int ren,col,suma;
  char espera;
  system("cls");
  printf("\n\n\t\tPrograma No 11  version 2 adicional");
  printf("\n\n\t\tPrograma que despliega dos matrices de 3 x 3 ");
  printf("\n\n\ty despues las suma usando la estructura de control for");
  printf("\n\n\t\t");
  system("PAUSE");
  /*empezamos con la primera matriz*/
  system("cls");
```

```
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           printf("\n\t\t Digite el valor de la matriz [ %d   %d ] ",ren+1,col+1);
           scanf("%d",&matriz1[ren][col]);
          }
       }
/*continuamos con la segunda matriz*/
  system("cls");
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           printf("\n\t\t Digite el valor de la matriz [ %d   %d ] ",ren+1,col+1);
           scanf("%d",&matriz2[ren][col]);
          }
       }
/*desplegamos la primera matriz*/
  system("cls");
  printf("\n\t\tMATRIZ 1\n");
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           printf("%10d ",matriz1[ren][col]);
          } printf("\n");
       }
/*continuamos con el despliegue de la segunda matriz*/
printf("\n\t\tMATRIZ 2\n");
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           printf("%10d ",matriz2[ren][col]);
          } printf("\n");
       }
 /* hacemos la suma de matrices*/
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           matriz3[ren][col]=matriz1[ren][col]+matriz2[ren][col];
          }
       }
/*desplegamos la matriz resultado*/
printf("\n\n\t\tMATRIZ RESULTADO\n\n");
  for (ren=0;ren<=2;ren++)
          {
      for (col=0;col<=2;col++)
             {
           printf("%10d ",matriz3[ren][col]);
          } printf("\n");
       } printf("\n");
  printf("\n\n\t\tPULSE CUALQUIER TECLA PARA CONTINUAR");
  espera=getch();
  return 0;
  }
```