Las operaciones matemáticas

La computadora es en realidad una gran máquina, la cual si estás escuchando música, viendo una película o jugando al Counter Strike, sólo realiza cálculos.

Este capítulo le enseñará cómo realizar la mayoría de los cálculos de una computadora puede hacer. Reutilizamos lo que hemos aprendido, es decir, las variables.

La idea es sólo hacer cálculos con variables: sumar, multiplicar, guardar el resultado en otro, etc.

Los cálculos de base:

Es bueno tener en mente que la computadora solo realiza cálculos muy sencillos:

- Adición
- Sustracción
- Multiplicación
- División
- Modulo

Ahora, si queremos realizar cosas más complicadas deberemos decirlo a la maquina que hacer (por ejemplo si queremos realizar una potencia o un logaritmo), afortunadamente el lenguaje C tiene una librería matemática, que contiene muchas funciones que nos simplificaran la vida.

Vamos arrancando por algo sencillo

La suma y otras

Para realizar la suma utilizamos el signo +

Recordamos que si queremos almacenar el resultado deberemos crear una variable por ejemplo aquí la variable entero resultado.

```
int resultado=0;

resultado = 5+3;
```

Bien, no es muy complicado en la variable resultado tendremos 8

Si quisiéramos ver por pantalla esta variable podríamos utilizar un printf como ya hemos hecho:

printf(“5+3 = %d”,resultado);

Bien para las otras operaciones será similar solo cambiaran los símbolos:

- Adición (suma): +
- Sustracción (resta): -
- Multiplicación: *
- División: /
- Modulo: %

Vamos a hablar un poco más acerca de las dos últimas operaciones:

La división

La división sobre una computadora funciona bien cuando no tenemos resto, por ejemplo 4/2 nos dará 2

Ahora bien si tenemos 5/2 debería ser 2.5

Probamos las siguientes líneas:

```
int resultado = 0;

resultado = 5 / 2;
printf ("5 / 2 = %d", resultado);
```

Luego de ejecutar vemos que nos da 2!!

Bueno si queremos el resultado correcto deberemos cambiar el tipo de variable con el que estamos trabajando, por ejemplo tipo double:

```
double resultado = 0;

resultado = 5 / 2;
printf ("5 / 2 = %d", resultado);
```

El modulo

Esta es una operación un poco menos conocida que las anteriores, pero básicamente es el resto de la división lo que obtenemos, siempre que trabajemos con variables enteras por ejemplo:

```
int resultado = 0;

resultado = 5 % 2;
```

printf ("5 % 2 = %d", resultado);

Cálculos entre variables

Hasta lo que hemos visto solo podemos operar con valores que nosotros ingresamos, pero no sería más interesante trabajar con variables y cambiar directamente sus valores.

Comencemos por

resultado = numero1 + numero2;

Esta línea suma las variables numero1 y numero2, y a esta suma la guarda en la variable resultado
Para realizar cualquiera de las otras operaciones entre variables solo debemos cambiar el símbolo en la expresión anterior.
Y si por ejemplo quisiéramos agregar otra variable lo podemos hacer libremente por ejemplo:

resultado = numero1 + numero2+numero3;

Veamos un ejemplo, aquí se pide al usuario entrar dos números y se muestra en pantalla el resultado.

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

int main(int argc, char *argv[])
{
  int resultado = 0, numero1 = 0, numero2 = 0;

  // Le pedimos los números al usuario :

  printf("Entre el numero 1 : ");
  scanf("%d", &numero1);
  printf("Entre el numero 2 : ");
  scanf("%d", &numero2);

  // Realizamos el cálculo:

  resultado = numero1 + numero2;

  // Mostramos resultado en pantalla:
```

```
  printf ("%d + %d = %d\n", numero1, numero2, resultado);
   getch();
  return 0;
}
```

Algunos Atajos

- Si queremos aumentar una variable de uno en uno podemos escribir:

  var = var +1;

  o una forma equivalente de escribir esto seria

  var++

- Si deseáramos reducir de a uno

  var = var -1;

  que es equivalente a

  var--

- Similares a los anteriores podemos encontrar para los otros símbolos por ejemplo:

  num = num * 2;

  Podemos escribirlo como:

  num *=2;

  A manera de ejemplo vemos los siguientes:

```
int numero = 2;

numero += 4; // numero vale 6
numero -= 3; // ... numero vale ahora 3
numero *= 5; // ... numero vale 15
numero /= 3; // ... numero vale 5
```

numero %= 3; // ... numero vale 2

Biblioteca matemática

En C existen bibliotecas llamadas estándar, esto quiere decir que son siempre utilizables. Y que son una especie de biblioteca de base que usaremos más seguido.

Recordamos que es una biblioteca: una biblioteca es un conjunto de funciones ya escritas y listas para usarcé.

La biblioteca matemática tiene una gran cantidad de funciones, las cuales son fáciles de consultar por ejemplo en internet googleando **math.h**, aquí veremos las funciones más comunes y algunos ejemplos:

Primero como cualquier otra biblioteca para poder utilizarla debemos utilizar una directiva al procesador:

#include<math.h>

Luego de utilizar esta directiva podremos utilizar cualquiera de las funciones contenidas en esta biblioteca.

Comencemos a ver las funciones más comunes:

fabs

Esta función devuelve (da como resultado) el valor absoluto de un número, recordemos un poco de matemática el valor absoluto de un número es el valor positivo del mismo por ejemplo:

- Si le damos a esta función el valor -53, nos devolverá 53
- Si le damos a esta función el valor 53, nos devolverá el 53

Un ejemplo práctico:

```
#include <stdio.h>
#include<conio.h>
#include <math.h>
int main()
{
   int a= -8;
```

```
   double absoluto;
   printf( "%d",a);
   absoluto=fabs(a);
   printf("\n %f",absoluto);
   getch();
}
```

Algo importante a notar es que esta función devuelve un valor tipo doublé, entonces el tipo de variable que utilicemos para almacenar este resultado debe ser de este tipo.

** Existe otra función muy similar a esta llamada **abs** que la encontramos en la librería **stdlib.h**. Esta otra función funciona de la misma manera, salvo que utiliza enteros (int), entonces devuelve un número entero.

ceil

Esta función redondea un número decimal al número entero superior. Por ejemplo si le damos el número 26.523, la función devuelve 27.

Esta función se utiliza de la misma manera que la anterior y devuelve también un tipo doublé:

```
   double a= 8.445;
   double absoluto;
   absoluto=ceil(a);
```

floor

Esta es la inversa de la función anterior, es decir nos devuelve el número redondeado al entero inferior. Si le damos el número 37.91, la función floor devuelve 37.

pow

Esta función permite calcular la potencia de un número. Se le deben indicar 2 valores: el número y la potencia a la que queremos elevar el número, el esquema de la función es el siguiente:

pow(numero,potencia);

Veamos un ejemplo:

```
int resultado = 0, b = 2;
resultado = pow(b, 3);
```

Aquí calculamos la potencia 3 de la variable b, es decir b^3= b*b*b
Como b=2 entonces resultado será igual a 8.

Y si quisiéramos hacer una raíz de algún numero como podríamos hacerlo, recordemos el exponente fraccionario como era?
Por ejemplo:

$$\sqrt[b]{a} = a^{\frac{1}{b}}$$

Entonces con solo con escribir en el lugar de la potencia (1/b) ya lograríamos lo que queremos.

sqrt

Esta función calcula la raíz cuadrada de un número. Es importante saber que esta función devuelve una variable tipo doublé.

```
double resultado = 0, b = 25;
resultado = sqrt(b);
```

sin, cos, tan

Estas son tres funciones conocidas de la trigonometría, son el seno, coseno y la tangente, estas funciones devuelven un valor tipo doublé. Un detalle es que el valor que devuelven estas funciones además de ser un doublé, esta expresado en radianes.

asin, acos, atan

Estas son las contraparte de las anteriores y se usan de la misma manera, ellas son el arc seno, arc coseno y el arc tangente. También devuelven un tipo doublé

exp

Esta función calcula el exponencial de un número es decir:

$$e^x$$

También devuelve un tipo doublé.

log

Calcula el logaritmo natural de un número

log10

Calcula el logaritmo base 10 de un número

Constantes

Además de funciones también están definidas en esta librería ciertas constantes que pueden ser muy útiles para nuestros programas:

- `M_E` La base de los logaritmos naturales *e*.
- `M_LOG2E` El logaritmo de *e* de base 2.
- `M_LOG10E` El logaritmo de *e* de base 10.
- `M_LN2` El logaritmo natural de 2.
- `M_LN10` El logaritmo natural de 10.
- `M_PI` π
- `M_PI_2` $\pi/2$
- `M_PI_4` $\pi/4$
- `M_1_PI` $1/\pi$
- `M_2_PI` $2/\pi$
- `M_2_SQRTPI` $2/\sqrt{\pi}$
- `M_SQRT2` La raíz cuadrada positiva de 2
- `M_SQRT1_2` La raíz cuadrada positiva de 1/2

Con todo lo anterior tenemos una idea del contenido de esta biblioteca, aunque existen más funciones estas son las más comúnmente usadas, sin embargo esto dependerá de lo que cada programador este haciendo.

A poner en práctica!

Luego de toda la teoría anterior pongamos en práctica estas funciones:

1) Une fácil calcule y muestre en pantalla:
   a) Suma de dos números ingresados por teclado
   b) El absoluto de estos dos numero
   c) La raíz cubica de la suma de estos números
2) Genere una función que convierta un numero pasado por valor, de radianes a grados
3) Similar a la anterior pero que convierta de grados a radianes.
4) Genere un programa donde si ingresamos la parte real y la parte imaginaria (sin la i) de una impedancia Z nos devuelva el valor del modulo y la fase de la misma. Puede utilizar las funciones que hizo en el punto 2 y 3
5) Realice un programa que calcule la serie de 3 resistencias ingresadas por teclado.
6) Similar al anterior pero que calcule el paralelo.