Lenguaje de programación C++

Módulo 3: Funciones

1. Declaración y Definición de Funciones

En C++ todas las acciones se llevan a cabo mediante funciones. En C++ el programa también es una función llamada main.

Declaración de una función

```
tipo nombre (lista de parámetros)
{
        declaración de variables locales;
         estatutos;
         return valor;
 }
```

donde:

tipo es el valor de retorno de la función. Si la función no regresa ningún valor se pone el tipo void para indicarlo.

nombre es un identificador, el cual se recomienda sea un nombre representativo de lo que hace la función.

lista de parámetros es una lista de cero, una o más declaraciones de variables (parámetros) separadas por coma. Cada

parámetro se debe de declarar con su tipo de dato. Este tema se explicará a detalle en la sección 3 del módulo.

Notas sobre las funciones en C++:

- En C++ las funciones pueden declararse en 2 lugares distintos:

1. Después de la declaración de variables globales y antes de la función main() o la función que la vaya a llamar.

2. Después de la función main() o de la función que la vaya a llamar, en cuyo caso deben ser identificadas antes, para

ello debe declararse solamente el encabezado de la función, al que se llama **prototipo**, después de la declaración de variables

y antes de la función main(). Esta función prototipo le indica al compilador el tipo de dato que regresará la función, el

número de parámetros que la función espera recibir, el tipo de estos parámetros y el orden en el cual los parámetros

deben ser esperados. El compilador usa esta función prototipo par validar las llamadas a las funciones. En algunas

versiones de C no se realiza este tipo de verificación, pero en el caso de C++ si.

- Una función no puede declararse dentro de otra función.
- Toda función que no es void debe tener un return.
- Al llegar al estatuto return, la función se deja de ejecutar y envía el valor especificado como valor de retorno.

Orden de ejecución de las funciones en C++:

Como ya dijimos C++ se basa en funciones siendo main() la función principal. por lo que:

- La ejecución de un programa empieza en la función main().
- Las otras funciones se ejecutan cuando se mandan llamar.
- Si una función no se manda llamar, nunca se ejecuta.
- Como se comentó en las notas anteriores, para que una función sea llamada tiene que declararse antes.

Llamada a una función

Hay varias formas de llamar a una función, dependiendo de lo que se desea hacer con el resultado o resultados que regrese, pero la

regla que siempre se conserva es que la lista de parámetros debe coincidir con los valores que recibe la función en su declaración

en tipo y cantidad.

- Un estatuto de asignación si el valor que regresa es necesario en otra operación

  a = **nombre** (lista de parámetros);

- Una expresión

  a = cantidad *** nombre** (lista de parámetros);

- El estatuto de salida, si solo nos interesa imprimir el resultado, en caso de que no haya parámetros de referencia

  cout << **nombre** (lista de parámetros);

- Si hay parámetros de referencia, debemos llamar a la función y luego imprimir los valores de los argumentos que hayan sido

  modificados

  nombre (lista de parámetros);
  cout << parámetros;

- Una condición

  if (**nombre** (lista de parámetros) > 7)

  while (**nombre** (lista de parámetros) > 7)

Ejemplos

Ejemplo 1: Se necesita una función que calcule la distancia entre 2 puntos con coordenadas (Xa, Ya) y (Xb, Yb) . Para calcularla se usa la fórmula:

DISTANCIA =

```
#include <iostream.h>
#include <math.h>      //librería para usar las operaciones pow y sqrt

double distancia (double xx1, double yy1, double xx2, double yy2)
{
   double result,p1,p2;
   p1 = pow((xx2-xx1),2);
   p2 = pow((yy2-yy1),2);
   result = sqrt(p1+p2);
   return result;
}

int main()
{
   double x1,y1,x2,y2;
   cout << "Dame las coordenadas del 1er punto "<<endl;
   cin >> x1>>y1;
   cout << "Dame las coordenadas del 2do punto "<<endl;
   cin >> x2>>y2;
```

```
   cout<<"La distancia entre los 2 puntos es = "<<distancia(x1,y1,x2,y2)<<endl;
   return 0;
}
```

Ejemplo 2: Se desea obtener el perímetro de un triángulo dadas las coordenadas de los 3 puntos.

Usaremos la función de distancia del problema anterior para resolverlo

```
#include <iostream.h>
#include <math.h>      //librería para usar las operaciones pow y sqrt

double distancia (double xx1, double yy1, double xx2, double yy2)
{
   double result,p1,p2;
   p1 = pow((xx2-xx1),2);
   p2 = pow((yy2-yy1),2);
   result = sqrt(p1+p2);
   return result;
}

int main()
{
   double x1,y1,x2,y2,x3,y3,p12,p13,p23;
   cout << "Dame las coordenadas del 1er punto "<<endl;
   cin >> x1>>y1;
   cout << "Dame las coordenadas del 2do punto "<<endl;
   cin >> x2>>y2;
   cout << "Dame las coordenadas del 3er punto "<<endl;
   cin >> x3>>y3;
   // calculando la distancia entre el punto 1 y 2
   p12 =distancia(x1,y1,x2,y2);
   // calculando la distancia entre el punto 1 y 3
   p13 =distancia(x1,y1,x3,y3);
   // calculando la distancia entre el punto 2 y 3
   p23 =distancia(x2,y2,x3,y3);
   cout << "El perimetro es " <<p12+p13+p23<<endl;
   return 0;
}
```

Ejercicio

Identifica si son correctas las siguientes llamadas a una función llamada Compara cuyo encabezado es el siguiente:

```
double Compara (int x, double y);
```

1.
```
int a,b;
double z;

cout << Compara (a,b,z);
```

2. double a,b;

```
    int  z;

    if Compara(a,z)


3. double a,b,c;
   do
   {
     .....
    }
    while (compara
(a,b));
```

ver solución

Ligas sugeridas

http://www.cplusplus.com/doc/tutorial/

http://www.cs.wustl.edu/~schmidt/C++/

Regresar

Siguiente módulo