Lenguaje de programación C++

Módulo 3: Funciones

3. Parámetros

Los parámetros son la forma de comunicar valores, que se han almacenado en variables locales, entre las funciones de un programa.

Declaración de los parámetros en la definición de la función

tipo nombre (lista de parámetros)

```
{
        declaración de variables locales;

         estatutos;

         return valor;

}
```

Lista de parámetros es lista de cero, una o más declaraciones de variables separadas por coma. Cada parámetro debe de tener su

tipo antes del nombre. Un error común es cuando se tiene más de un parámetro del mismo tipo solo se le pone tipo al primero y se

separa por comas.

Ejemplo: Supongamos que queremos declarar una función que regresará un valor doble y recibe dos valores enteros num1 y num2.

La declaración correcta es: double ejemplo (int num1, int num2)

El típico error es poner: double ejemplo (int num1, num2) ó no poner tipo double ejemplo (num1, num2)

Tipos de parámetros en C++

1. Por valor

2. Por referencia

Parámetros por valor

Cuando un argumento es pasado por valor, se hace una copia del valor del argumento y se

pasa a la función que es llamada. Los

cambios a la copia no afectan el valor original de la variable que aparece en la llamada. Una desventaja del paso por valor es que una gran

cantidad de datos es pasado y copiado puede tomar un cantidad considerable de tiempo de ejecución y espacio de memoria. A

este tipo de parámetros se les conoce como parámetros de *"entrada"*

Ejemplos

Ejemplo 1: Programa que calcula el área de un círculo

```
#include <iostream>
#include <math>
using namespace std; //usando el namespace


double r;

double calc_area (double rad)   // rad es parámetro por valor recibe el valor
de r
{ double area;                  // declaracion de la variable local
  area = 3.14*pow(rad,2);
  return area;                  // se regresa a la función main() el resultado
}

int main()
{
cout << "Dame el radio "<<endl,
cin >> r;
cout << "El area es del circulo es "<<calc_area(r)<<" con un radio "<< r<<endl;
return 0;                     // el valor de r no cambia al efectuarse la
llamada a la función calc_area
}
```

Ejemplo 2: Desarrolla un programa que tenga la función factorial para calcular la aproximación de e (número de Euler).
La aproximación debe basarse en la siguiente fórmula:

$$1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \ldots\ldots$$

```
#include <iostream.h>

int fact(int n)
{  int pot=1;
   for (int i=1;i<=n;i++)
      pot = pot *i;
    return pot;
}

void main()
{
   int euler;
   double suma=1;
   cout << "Dame la aproximacion que quieres "<<endl;
   cin >> euler;
   for (int i=1;i<=euler;i++)
      suma = suma + 1.0/fact(i);
```

```
   cout << "El valor de la aproximacion es "<< suma<<endl;
}
```

Notas importantes:

- Para evitar ambigüedad, se recomienda no usar el mismo nombre en los argumentos pasados a una función y los

  correspondientes parámetros en la definición de la función.

- En la llamada a una función el tipo y número de parámetros debe coincidir con los de la definición de la función.

Referencias y parámetros de referencia

Un parámetro por referencia es un alias para su argumento correspondiente en la llamada a la función. Para indicar que un

parámetro es pasado por referencia, se pone un & después del tipo de dato y antes del nombre de la variable. Estos parámetros se

nombran comúnmente como parámetros de "entrada/salida", pues estos alteran el valor el valor del argumento al momento de

acabar la llamada a la función.

Este tipo de parámetros es usado cuando se requiere una función que regrese más de un valor.

Ejemplos

Ejemplo 1: Función llamada Separa que reciba como parámetro un número X de tipo double, y que regrese dos parámetros de tipo

double: en el primero de ellos deberá regresar la parte entera del número X y en el segundo la parte fraccionaria del número X.

Por ejemplo:

al llamar Separa (32.45, a, b);

la variable a tomará el valor de 32.0 y la variable b tomará el valor de 0.45.

```
#include <iostream.h>

int Uno (double x1, int &a1, double &b1)
{
  a1 = x1; // al pasarlo a un entero pierde la parte fraccionaria
```

```
  b1=(x1-a1)*100;
  return 0;
}

int main()
{
 double x,b;
 int a;
 x=45.12;
 Uno(x,a,b);   // Llamada a la función con dos argumentos que cambiarán de valor
 cout<<"Parte entera = "<<a<<endl;
 cout<<"Parte fraccionaria = "<<b<<endl;
 return 0;
}
```

Ejemplo 2: Programa que lea un entero n que representa la cantidad de números a leer y la mande a una función que deberá regresar el valor más grande y más chico de los números leídos.

```
int bigsmall(int n, int &mayor, int &menor)
{   int num;
   // vamos a leer el primero asumiendo que es tanto mayor como menor
   // para después comparar con los leídos
   cout <<"Dame el numero "<<endl;
   cin >> num;
   mayor = num;
   menor = num;
   for (int i=2;i<=n;i++)
   {
      cout <<"Dame el numero "<<endl;
      cin >> num;
      if (num > mayor)
          mayor = num;
      if (num < menor)
          menor = num;
   }
   return 0;
}

void main()
{
   int numer,may,men;
   cout << "Dame la cantidad de numeros a leer "<<endl;
   cin >> numer;
   bigsmall(numer,may,men);
   cout << "El valor mayor es "<< may<<endl;
   cout << "El valor menor es "<< men<<endl;
}
```

Ejercicio

Escribe una función que reciba como parámetro la fecha de nacimiento de una persona y la fecha actual y que calcule la edad de la persona, si tiene menos de un año que dé la edad en meses.

Por ejemplo:

Si la fecha de nacimiento es 12 de Oct de 2000 y la fecha actual es 1 de Oct de 2003 el programa debe decir que la persona tiene 2 años cumplidos.

Si la fecha de nacimiento es 15 de Marzo de 1980 y la fecha actual es 29 de Marzo de 2003, el programa debe decir que la persona tiene 13 años.

Si la fecha de nacimiento es el 12 de Noviembre de 2002 y la fecha actual es el 15 de Agosto de 2003 el programa debe decir que la persona tiene 9 meses de edad.

ver solución

Ligas sugeridas

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

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

Regresar

Siguiente módulo