ANALISIS SOBRE LOS ELEMENTOS BASICOS DEL LENGUAJE EN C++

En un programa de C++ pienso que lo primordial son las funciones y en especial la función main(), aunque se sabe de antemano que ningún programa en C++ funcionaria sin alguna variable, carácter u cualquier tipo de dato estuviera mal escrito o no estuviera en su lugar, pero pienso que la función main es primordial ya que todo empieza a ejecutarse por esta función.

Se trata de la función de entrada, y debe existir siempre, será la que tome el control cuando se ejecute un programa en C++. Esta se define primeramente escribiendo el tipo del valor de retorno de la función, después el nombre de la función y colocando entre paréntesis las variables o parámetros que usara la función.

Otros elementos básicos de un programa en C++ son:

Componentes, los que a su vez se componen de:

- Sentencias terminadas en ; (punto y coma)
- /*Inicio de comentarios y fin de comentarios*/ (esta para comentarios de mas de dos renglones) y //Inicio y fin de comentarios (esta para comentarios de un solo renglón).

  Los Comentarios son anotaciones (observaciones, recordatorios, etc.) en el programa los cuales son solamente para uso de nosotros los programadores, y son eliminados del código fuente en la fase de pre procesado; antes del análisis sintáctico, es decir que el compilador de C++ ignora todo lo que este entre los símbolos ya dichos anteriormente.
- Variables, constantes, operadores, expresiones, funciones.
- Bloques, los cuales son estatutos incluidos entre llaves ({ … })

Directivas, que empiezan por el carácter #, son órdenes para el pre procesador de C++ que revisa ciertas tareas previas a la compilación, e incluye funciones de alguna biblioteca como por ejemplo la iostream, la cual es un archivo de cabecera para la entrada o salida en el C + +, dos de los objetos que usa la biblioteca iostream son el cin y el cout, la cual cin es para la entrada y cout para la salida.

La directiva para incluir bibliotecas en C++ es la siguiente:

#include <iostream.h>

Declaraciones Globales, Una declaración notifica las propiedades de una variable o función. Las declaraciones globales son declaraciones que van por fuera de una función y permite que estas declaraciones sean usadas en cualquier función del programa.

El uso de este tipo de variables suele considerarse como una mala práctica, por el riesgo que conlleva esa deslocalización: una variable global puede ser modificada en cualquier parte del programa (a menos que resida en una sección de memoria protegida) y cualquier parte del programa depende de ella. Es por ello que una variable global tiene un potencial ilimitado para crear dependencias, factor éste que aumenta la complejidad. Sin embargo, en algunas ocasiones, las variables globales resultan muy útiles. Por ejemplo, se pueden usar para evitar tener que pasar variables usadas muy frecuentemente de forma continua entre diferentes subrutinas.

```
#include <iostream>

int global = 3; // Esta es la variable global.

void ChangeGlobal()
{
  global = 5; // Se referencia la variable global en una función.
}

int main()
{
  std::cout << global << '\n'; // Se referencia la variable global en una segunda
función.
  ChangeGlobal();
  std::cout << global << '\n';
  return 0;
}
```

Dado que la variable es de ámbito global, no hay necesidad de pasarla como parámetro a otras funciones. La variable global pertenece a todas las funciones del programa.

Tipos de datos, Los tipos de datos que se manejan en C++ son los siguientes:

- **Datos enteros**, es de tipo INT, que puede representar un subconjunto finito de los números enteros. El número mayor que puede representar depende del tamaño del espacio usado por el dato y la posibilidad (o no) de representar números negativos. Los tipos de dato entero disponibles y su tamaño dependen del lenguaje de programación usado así como la arquitectura en cuestión.

Ejemplo de int:

```
Void main()
{
Int a;
Cout<<”digite dato”;
Cin>>a;
Cout<<”el dato ingresado fue ”<<a;

}
```

Donde a puede ser un numero entero.

O donde también se puede usar con un switch. Ejemplo:

```
Void main
{
Int sw=0;
If (a>3)
{
Sw=1;
}
If sw=0
{
Cout<<”el numero no fue mayor a tres”;
}
Else
{
```

```
Cout<<”el numero fue mayor a tres”;

}

}
```

- **Tipo Float,** Las variables del tipo float (coma flotante) se usan para guardar números en memoria que tienen parte entera y parte decimal. Por tanto, los valores de las variables en coma flotante en un ordenador solamente se aproximan a los verdaderos números reales en matemáticas.

  Ejemplo:

```
Void main()

{

float a;

Cout<<”digite dato”;

Cin>>a;

Cout<<”el dato ingresado fue ”<<a;


}
```

  Donde a puede ser un decimal.

- **Tipo Char,** Un carácter está siempre rodeado de comillas simples como 'A', '9', 'ñ', etc. El tipo de dato char sirve para guardar estos caracteres.

  el tipo char solo tienen 1 byte de tamaño y usa la misma representación que los número enteros. La única diferencia es su tamaño y que el char es usado para representar los 255 caracteres de la tabla ASCII. En cuanto a la forma de declarar variables de tipo char es la misma forma que con los otros tipos.

  Ejemplo:

```
Void main()

{

char a;

Cout<<”digite letra”;

Cin>>a;

Cout<<”la letra ingresada fue ”<<a;


}
```

Estatutos de Lectura, Como ya se menciono en la parte de directivas, los estatutos de lectura están en la biblioteca iostream y esos estatutos son cin y cout, donde cout es el estatuto de escritura y cin es el estatuto de lectura.

Cada uno de estos estatutos puede tener mas variables separadas por <<(esta perteneciente a cout) y >> (esta perteneciente a cin), ejemplo:

```
Void main()

{

Int a,b,c

Cout<<”digite datos”;

Cin>>a>>b>>c;

Cout<<”los datos ingresados fueron ”<<a<<b<<c;


}
```

Existen caracteres especiales que se pueden usar solamente dentro del cout los cuales son conocidos como **caracteres de escape** y se usan dentro de los mensajes que se van a mandar a pantalla,

Dos de los caracteres especiales mas usados son el **\n** el cual realiza un cambio de línea, y el **\t** que es el tabulador. A continuación se muestran ejemplos de cómo usar ambos.

```
Void main()

{

float b;

Cout<<”\n digite dato:\n”;

Cin>>b;


Cout<<”\t el dato ingresado fue \t ”<< \t a\t ;


}
```

En pantalla se ve algo asi:

Digite el dato:

(Ej:) 5

El dato ingresado fue

5

Otro carácter especial es en <<endl; que hace exactamente lo mismo al \n, solo que este si se puede usar tanto en el cin como en el cout y va afuera de las comillas. Ejemplo:

```
Void main()

{

Int a,b,c


Cout<<”digite datos”<<endl;

Cin>>a<<endl;

Cin>>b<<endl;

Cin>>c<<endl;
```

```
Cout<<”los datos ingresados fueron ”<<a<<b<<c;

}
```

FUNCION MAIN Y ESTRUCTURA DE UN PROGRAMA

```
#include<iostream.h>    / /Directivas del preprocesador
Variables
Funciones
Prototipo de Funciones
Main()    // Función principal
{
    Variables // declaraciones
                // locales
    Conjunto de instrucciones
}
Function ()
{
}
```

Una función es un conjunto de líneas de código que realizan una tarea específica y puede retornar un valor. Las funciones pueden tomar parámetros que modifiquen su funcionamiento. Las funciones son utilizadas para descomponer grandes problemas en tareas simples y para implementar operaciones que son comúnmente utilizadas durante un programa y de esta manera reducir la cantidad de código. Cuando una función es invocada se le pasa el control a la misma, una vez que esta finalizó con su tarea el control es devuelto al punto desde el cual la función fue llamada.
En el llamado de una funcion una vez que en su programa se ha definido una función, esta puede ser llamada las veces que sean necesarias. Para llamar a una función basta con hacer referencia a su nombre y si la misma requiere de parámetros estos deberán indicarse dentro de parentesis. Para llamar a una función que no requiera de parámetros se deberá indicar el nombre de la misma seguida de parentesis vacios. Por ejemplo, para llamar a la función cuadrado() vista anteriormente, podemos emplear:

```
cout << cuadrado(25);
cout << cuadrado(X);
R = cuadrado(X);  // guardar en R el cuadrado de X.
```

BIBLIOGRAFIA

http://www.zator.com/Cpp/E3_1.htm

http://www2.udec.cl/~gonzahidalgo/lp/apuntes/fundamentos_c.pdf

http://es.wikibooks.org/wiki/Programación_en_C