**+Instituto Superior de Formación Técnica Nº 151 logo151-trans.gif  
Carrera: Analista de Sistemas  
1 Año. Algoritmos y Estructuras de Datos I**

|  |  |
| --- | --- |
| **Trabajo Práctico Nº 2** | **Unidad 2** |
| **Modalidad:** Semi-Presencial | **Estratégica Didáctica:** Trabajo Grupal. |
| **Metodología de Desarrollo:** acordar | **Metodología de Corrección:** acordar docente |
| **Carácter de Trabajo:** Obligatorio – con Nota | **Fecha Entrega:** A confirmar por el Docente. |

**Marco Teórico:**

1. ¿Describir los módulos que componen un programa en C++?
2. ¿Qué función cumple el “Insertor”?
3. ¿Qué es Cout? ¿para qué sirve?
4. ¿En qué librería está Incluido?
5. ¿Pertenece a algún espacio de Nombres?
6. ¿Qué función cumple un espacio de Nombres?
7. ¿Describir los Operadores y dar ejemplos?
8. Qué es una variable, qué función cumple
9. Dar ejemplos de Variables
10. ¿Qué diferencia hay entre Integer y Decimal?
11. Dar ejemplos de operaciones entre tipos ( + - \* / ) explicar que pasa en la operación.

**Marco Práctico:**  
Desarrollar en C++ los siguientes programas:

1. Resolver y Realizar en c++
   1. La suma de 45 + 34 mostrar x pantalla el resultado
   2. La resta de 45 -40 mostrar le resultado x pantalla
   3. La división de 46 y 3
   4. Un programa que multiplique 65 \* 3 con la salida en pantalla
2. Resolver usando Variables y operadores, mostrar el resultado x pantalla.
   1. Calcula el volumen, de una habitación que tiene 5 m de largo, 4 m de ancho y 2.5 m de alto.
   2. Escribe un programa en C++ que nos diga cuál es el volumen de un cono con un radio de la base de 14,5 y una altura de 26,79. La fórmula que debes usar es: (Pi x (radio)2 / altura) /3, recordar Pi=3.14.
   3. ¿Modificar el programa anterior para que usen variables Dobles, que diferencia nota? ¿Por qué?

Lic. Oemig José Luis.

**Marco Teórico:**

1. Los lenguajes de programación requieren el uso de determinadas *estructuras de control* para expresar los algoritmos como programas. Hay cuatro formas básicas de estructurar sentencias (instrucciones) en la mayoría de los lenguajes de programación: de **modo secuencial**, **condicional**, **repetitivo** y con **subprogramas**. Una secuencia es una serie de sentencias que se ejecutan una después de otra. La estructura de control condicional ejecuta sentencias diferentes dependiendo de determinadas condiciones. La estructura de control repetitiva, el ciclo, repite sentencias mientras se satisfacen ciertas condiciones. El subprograma permite estructurar un programa al descomponerlo en unidades más pequeñas. Cada una de estas formas de estructurar sentencias controla el orden en el cual la computadora ejecuta las sentencias, razón por la que se llaman **estructuras de control**.

En C++, los subprogramas se denominan **funciones**, y un programa C++ es una colección de una o más funciones. Cada función realiza alguna función particular y, de manera colectiva, todas cooperan para resolver un problema completo. Cada programa C++ debe tener una función llamada *main*. La ejecución del programa comienza siempre con la función *main*.

Una llave izquierda ({) y una llave derecha (}) marcan el comienzo y el final de las sentencias que serán ejecutadas. Las sentencias que aparecen entre llaves se conocen como cuerpo de la función.

1. El operador de inserción ( << ) convierte a su operando derecho en una secuencia de caracteres y los inserta en (o, de manera más precisa, agrega a) el flujo de salida.
2. En C++ los valores de variables y expresiones se escriben por medio de una variable especial llamada cout junto con el operador de inserción (<<):

cout << "Hello";

Esta sentencia muestra los caracteres Hello en el dispositivo de salida estándar, por lo general la pantalla.

La variable cout está predefinida en sistemas C++ para denotar un flujo de salida.

1. La variable cout está incluida en la librería iostream. La línea #include <iostream> instruye al compilador C++ que inserte en el programa el contenido del archivo llamado iostream. El mismo contiene información que C++ necesita para enviar valores a un flujo como cout.
2. El cout pertenece al *namespace* (espacio de nombres) std.
3. Muchos archivos de encabezado (y en general todo archivo de encabezado estándar) declara todos sus identificadores en un *namespace* (en el caso de los estándar, en std)

Por ejemplo: **El siguiente programa**:

int main()

{

cout << "Feliz cumpleaños" << endl;

return 0;

}

**no compilará**, incluso si se añade la directiva al preprocesador #include <iostream> .

El problema es que el archivo de encabezado iostream declara que todos sus identificadores están en un namespace llamado std

Se puede tener acceso directo a un identificador declarado dentro de un bloque de espacio de nombre (namespace) sólo mediante las sentencias dentro de ese bloque. Para tener acceso a un identificador que está “oculto” dentro de un espacio de nombres, el programador tiene varias opciones. Aquí se describen dos de ellas.

La primera opción es usar un nombre calificado para el identificador. Un nombre calificado consta del nombre del namespace, luego el operador :: (el operador de resolución de alcance) y después el identificador deseado:

std::cout

Con este método el programa se parece a lo siguiente:

#include <iostream>

int main()

{

std::cout << "Feliz cumpleaños" << std::endl;

return 0;

}

Aquí tanto cout como end1 deben ser calificadas.

La segunda opción es usar una sentencia denominada directiva using:

using namespace std;

Cuando se coloca esta sentencia cerca de la parte superior de un programa antes de la función main, se logra que todos los identifi cadores del espacio de nombres std tengan acceso al programa sin tener que calificarlos:

#include <iostream>

using namespace std;

int main()

{

cout << "Happy Birthday" << endl;

return 0;

}

1. Existen distintos tipos de operadores:

* Operadores aritméticos:

( + ) : Suma

( - ) : Resta

( \* ) : Multiplicación

( / ) : División

* Operadores Lógicos:

( && ) : AND o Producto Lógico

( || ) : OR o Suma Lógica

( ! ) : NOT o Negador Lógico

1. Una **variable** es un *dato* que se accede por medio de un *nombre.* Dicho dato puede *variar* (de ahí su nombre “variable”) a lo largo de la ejecución del programa. Su función es almacenar un dato.
2. Ejemplos de Variables:

int cantidad; // cantidad es una variable de tipo entera

double precio; // precio es una variable de tipo coma flotante

bool respuesta; // respuesta es una variable de tipo booleana o lógica

1. En C++ los diferentes tipos enteros que encontré fueron:

short int, int, long int, long long int

y decimales (con coma flotante):

float, double, long double

y la diferencia entre enteros y decimales es en el tipo de dato que pueden guardar, más allá de la precisión de cada uno.

1. Cuando realizamos operaciones entre enteros, el resultado que mostrará será entero (trunca el resultado).

Ejemplo:

10 / 6 Dará como resultado 1 (y no 1.666666…)

En cambio, si uno de los dos operandos es real el resultado sí será real (decimal)

Ejemplo:

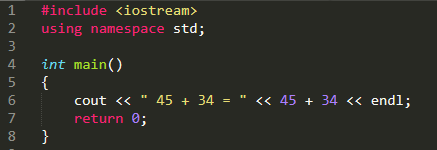
10.0 / 6 Dará como resultado 1.6666666…

10 / 6.0 Dará como resultado 1.6666666…

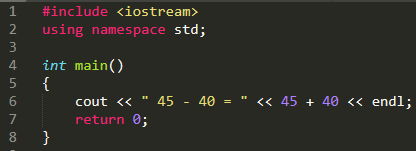
10.0 / 6.0 Dará como resultado 1.6666666…

**Marco Práctico:**

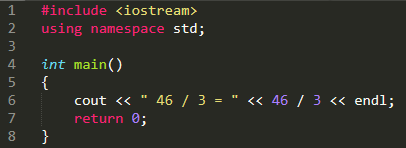
1. Resolver y Realizar en C++:
2. La suma de 45 + 34 mostrar x pantalla el resultado:



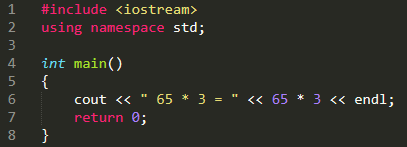
1. La resta de 45 -40 mostrar le resultado x pantalla



1. La división de 46 y 3

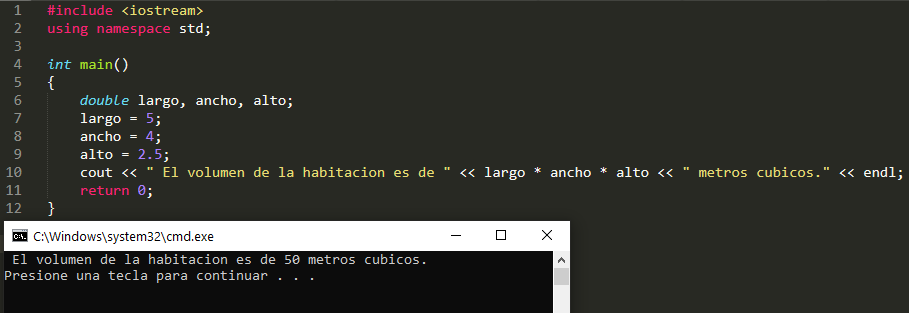


1. Un programa que multiplique 65 \* 3 con la salida en pantalla

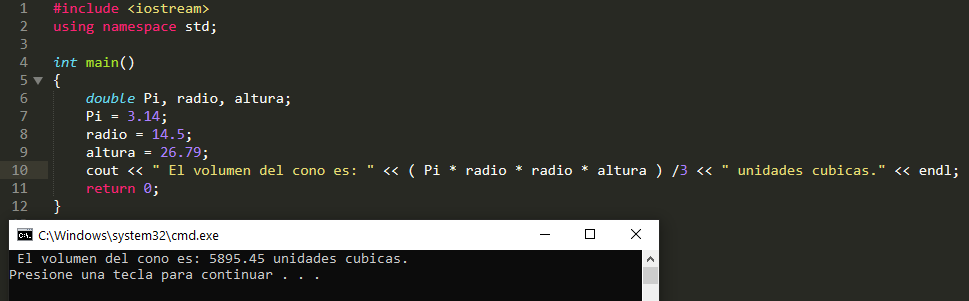


2. Resolver usando Variables y operadores, mostrar el resultado x pantalla.

1. Calcula el volumen, de una habitación que tiene 5 m de largo, 4 m de ancho y 2.5 m de alto.



1. Escribe un programa en C++ que nos diga cuál es el volumen de un cono con un radio de la base de 14,5 y una altura de 26,79. La fórmula que debes usar es: (Pi x (radio)2 / altura) /3, recordar Pi=3.14.



1. ¿Modificar el programa anterior para que usen variables Dobles, que diferencia nota? ¿Por qué?

