Ayudantía 1

Tipos de Datos

Benjamín Aceituno

Programación S04

Variables

Las variables en C++ son contenedores que permiten almacenar y manipular datos en un programa.

Estas mismas pueden ser declaradas y asignadas dentro del programa para ser ocupadas por el usuario.

Existen distintos tipos de variables en C++, como por ejemplo

int: **Almacena números enteros sin parte decimal.**

float: **Almacena números con parte decimal.**

double: **Al igual que el float, almacena números con parte decimal, pero siendo más exacto.**

long: **Almacena números enteros pero de mayor rango que int.**

char: **Almacena un solo caracter.**

string: **Almacena una cadena de caracteres, es decir, texto.**

Tipo de Dato	Rango de Valores
int	-2,147,483,648 a 2,147,483,647
long	-9,223,372,036,854,775,808 a 9,223,372,036,854,775,807
double	2.3×10^308 a 1.7×10^308
char	-128 a 127 (con signo), 0 a 255 (sin signo)
string	Depende la cantidad de memoria del sistema

Encuentre el error

El siguiente programa en C++ no funciona debido a disintos errores que tiene el código, intente identifcar cuáles son estos.

```cpp
#include <iostream>
using namespace std;

int main(){
    int A;
    int B = A;
    int c = 5;
    string D = Hola Mundo;

    A = 8;
    C = A + 4;

    cout << "A: " << A << endl;
    cout << "B: " << B << endl;
    cout << "C: " << C<< endl;
    cout << "Saludos! " << D << endl;
    return 0;
}
```

Código corregido

```cpp
#include <iostream>
using namespace std;

int main(){

    int A;
    int B;
    int C = 5; // Usar C mayuscula, para poder operar a lo largo del código
    string D = "Hola Mundo"; // Corregido, es necesario usar las comillas

    A = 8; // Inicializamos el valor A
    B = A; // Le damos el valor de A a B ya que A está inicializado
    C = A + 4;

    cout << "A: " << A << endl;
    cout << "B: " << B << endl;
    cout << "C: " << C<< endl;
    cout << "Saludos! " << D << endl;
    return 0;
}
```

Operaciones y Asignaciones

A continuación veremos distintos operadores y asignaciones que se pueden realizar a variables de tipo entero C++.

Donde podemos contemplar distintas operaciones aritméticas convencionales, y asignaciones de nuevos valores para las variables.

```cpp
#include <iostream>
using namespace std;

int main() {

    int a = 10;
    int b = 3;
    int suma = a + b; // 10 + 3 = 13
    int resta = a - b; // 10 - 3 = 7
    int division = a / b; // 10 / 3 = 3
    int modulo = a % b; // Dividendo - (Cociente * Divisor) -> 10-9 = 1

    a += 5;  // Esto es lo mismo que decir a = a + 5 (15)
    b *= 2;  // Esto es lo mismo que decir b = b * 2 (6)

    cout << "Suma: " << suma << endl;
    cout << "Resta: " << resta << endl;
    cout << "División: " << division << endl;
    cout << "Resto: " << resto << endl;
    cout << "Nuevo valor de a: " << a << endl;
    cout << "Nuevo valor de b: " << b << endl;

    return 0;
}
```

Módulo

El módulo en C++ se denota con el signo % y se utiliza de la misma forma que la división, pero da como resultado el resto de la división.

Cómo se calcula el resto:

```
10 % 3 = 1;
10/3 = 3 ->
3*3 = 9 ->
10 - 9 = 1
```

Ejercicio módulos

En C++ si aplicamos un módulo de 10 a un número podemos obtener el último dígito de esta misma, por ejemplo

```
int a = 456;
int ultimoDigito = a % 10; (6)
```

Implemente un programa en C++ que sea capaz de entregar el número “123” en su forma inversa, usando módulos solamente.

Solución

```cpp
#include <iostream>
using namespace std;
int main() {

    int a = 123;

    int ultimoDigito = a % 10;         // Último dígito: 12|3| % 10 = 3
    int digitoMedio = (a / 10) % 10;  // Dígito del medio: (1|2|3 / 10) -> (12), 12 % 10 = 2
    int primerDigito = (a / 100) % 10;  // Primer dígito: (|1|23 / 100) -> (1), 1 % 10 = 1;

    // Imprimir los dígitos concatenados como el número invertido
    cout << ultimoDigito << digitoMedio << primerDigito << endl;

    return 0;
}
```

Operaciones con int y long

Nosotros podemos hacer distintas operaciones variables int y long, pero debemos tener ojo de no pasarnos del límite que tiene cada variable, como sabemos, int tiene un rango mucho menos que long

```cpp
#include <iostream>
using namespace std;

int main() {
    int num1 = 20;
    long num2 = 100000; //Numero largo
    long num3 = 10000000000000000; //Numero muy largo que excede el rango de los int

    int suma = num1 + num2;
    long resta = num2 - num1;
    int multiplicacion = num1 * num2;
    long division = num2 / num1;

    int multiplicacionFueradeRango = num1 * num3;

    cout << "Suma con int: " << suma << endl;
    cout << "Resta con long: " << resta << endl;
    cout << "Multiplicación con int: " << multiplicacion << endl;
    cout << "División con long: " << division << endl;
    cout << "Multiplicación fuera de rango: " << multiplicacionFueradeRango << endl;

    return 0;
}
```

```
Suma con int: 100020
Resta con long: 99980
Multiplicación con int: 2000000
División con long: 5000
Multiplicación fuera de rango: -1156317184
```

Operaciones con float y double

Al igual que con int y long nosotros podemos hacer operaciones también con float y double, pero también tenemos que tener cuidado con las caracteristicas y rangos de cada variable.

Podemos entender al float y el double como la contraparte decimal de int y long

```cpp
#include <iostream>
using namespace std;

int main() {
    float num1 = 20.3;
    float num2 = 132.33;
    double num3 = 1.7e307;   // Número muy grande

    // Operaciones
    float suma = num1 + num2;
    double resta = num2 - num1;
    float multiplicacion = num1 * num2;
    double division = num3 / num2;

    float multiplicacionFueradeRango = num1 * num3;

    cout << "Suma con float: " << suma << endl;
    cout << "Resta con double: " << resta << endl;
    cout << "Multiplicación con float: " << multiplicacion << endl;
    cout << "División con double: " << division << endl;
    cout << "Multiplicación fuera de rango: " << multiplicacionFueradeRango << endl;

    return 0;
}
```

```
Suma con float: 152.63
Resta con double: 112.03
Multiplicación con float: 2686.3
División con double: 1.28467e+305
Multiplicación fuera de rango: inf
```

Redeondeo y truncamiento

Podemos truncar variables float haciendo la conversión de float a int y redondearlos con la función round()

```cpp
#include <iostream>
#include <cmath> // Para redondear
using namespace std;

int main() {
    float x = 5.7;
    int y = static_cast<int>(x);  // Conversión de float a int (truncamiento)

    cout << "Valor original de x: " << x << endl;
    cout << "Valor truncado de x a int: " << y << endl;

    // Redondeo
    int redondeado = round(x);
    cout << "Valor redondeado de x: " << redondeado << endl;

    return 0;
}
```

```
Valor original de x: 5.7
Valor truncado de x a int: 5
Valor redondeado de x: 6
```

Simplificación de operadores para contador

Podemos simplificar operaciones para por ejemplo, usar un contador y aumentar o disminuir en uno el valor de un contador

```cpp
#include <iostream>
using namespace std;

int main() {
    int contador = 10;
    contador++;  // Esto es lo mismo que decir
                 // contador = contador + 1 (contador = 11)

    contador--;  // Esto es lo mismo que decir
                 // contador = contador - 1 (contador = 10)

    contador += 5;  // Esto es lo mismo que decir
                    //contador = contador + 5 (contador = 15)
    contador -= 3;  // Esto es lo mismo que decir
                    // contador = contador - 3 (contador = 12)

    cout << "Valor final de contador: " << contador << endl;

    return 0;
}
```

```
Valor final de contador: 12
```

Suma de strings

Podemos sumar strings para concatenar o unir de cierto modo distintas palabras que nosotros queramos, uniendo los strings al sumarlos en nuestro programa.

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    string saludo = "Hola!";
    string saludo2 = "Como estas?";

    // Concatenación de strings
    string saludoCompleto = saludo + " " + saludo2;

    cout << "Saludo completo: " << saludoCompleto << endl;

    return 0;
}
```

```
Saludo completo: Hola! Como estas?
```

Conversión de string e int

Podemos convertir string a int como un int a un string usando sstream para que adquieran distintas propiedades, donde el int adquiere propiedades de un string y el string de un int.

```cpp
#include <iostream>
#include <sstream> // Necesario para stringstream
using namespace std;

int main() {
    // Conversión de texto a número
    string textoNumero = "123";
    int numero;

    stringstream ss(textoNumero);
    ss >> numero;

    cout << "El texto convertido a numero es: " << numero << endl;
    cout << "Suma de texto convertido: " << numero + 10 << endl;
    // Conversión de número a texto

    int otroNumero = 456;
    string texto;

    stringstream ss2;
    ss2 << otroNumero;
    texto = ss2.str();

    cout << "El texto convertido es: " << texto << endl;
    cout << "Concatenacion: " << texto + textoNumero << endl;

    return 0;
}
```

```
El texto convertido a numero es: 123
Suma de texto convertido: 133
El texto convertido es: 456
Concatenacion: 456123
```

Porcentajes y proporción

Podemos expresar porcentajes y proporciones en C++ usando las distintas operaciones aritmeticas que nos proporciona

```cpp
#include <iostream>
using namespace std;

int main() {
    float total = 300.0;
    float parte1 = 75.0;
    float parte2 = 150.0;

    // Cálculo de proporciones en porcentaje
    float porcentaje1 = (parte1 / total) * 100.0;
    float porcentaje2 = (parte2 / total) * 100.0;

    cout << "Parte 1 representa el " << porcentaje1 << "% del total." << endl;
    cout << "Parte 2 representa el " << porcentaje2 << "% del total." << endl;

    // Cálculo del cambio porcentual entre dos valores

    float valorInicial = 120.0;
    float valorFinal = 180.0;
    float cambioPorcentual = ((valorFinal - valorInicial) / valorInicial) * 100.0;

    cout << "El valor ha cambiado en un " << cambioPorcentual << "%." << endl;

    return 0;
}
```

```
Parte 1 representa el 25% del total.
Parte 2 representa el 50% del total.
El valor ha cambiado en un 50%.
```

Ejercicio

Se le pide realizar un código en C++ donde deba ingresar su nombre y apellido para calcular su promedio de notas, considere que tiene 3 solemnes y 3 controles, los controles equivalen a un 40% y las solemnes a un 60%.

Además, el resultado debe tener un solo decimal, para que pueda ser ingresado correctamente al sistema de notas.

Ejemplo de input: Estimado Benjamín Aceituno, su promedio de notas es 6.9

Contacto

+569 86031881

benjamin.aceituno@mail.udp.cl

benja.mp4

https://github.com/benjamp4