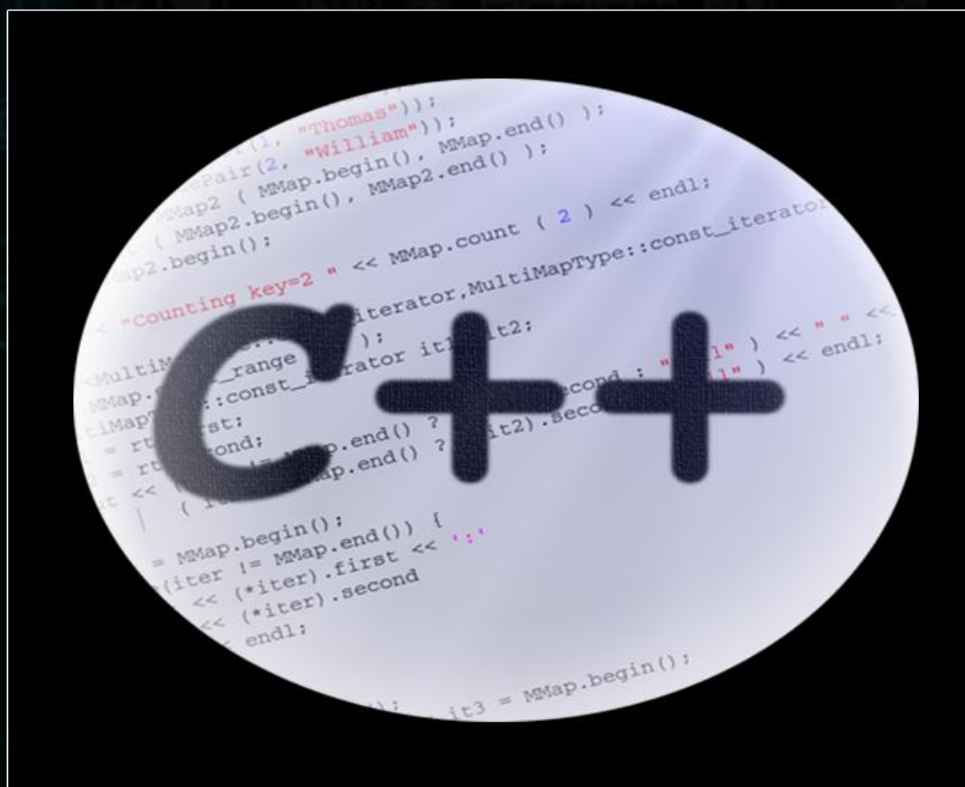


UNDERCODE

TALLER DE C++



TEMAS

INTRODUCCIÓN
INSTALACION
HOLA MUNDO
OPERADORES
VARIABLES
ACTIVIDADES

TUTOR

FOM3T

Primeramente, ¿qué es C++?

C++ es un lenguaje de programación que toma de base C, diseñado a mediados de los 80's por Bjarne Stroustrup.

La intención de su creación fue el extender al exitoso lenguaje de programación C con mecanismos que permitieran la manipulación de objetos. En ese sentido, desde el punto de vista de los lenguajes orientados a objetos, el C++ es un lenguaje híbrido.

¿Qué necesitas para programar en C++?

Un compilador de C++ (En este caso, usaremos Dev-C++), y muchas ganas de aprender.

Si estás en Windows, puedes usar:

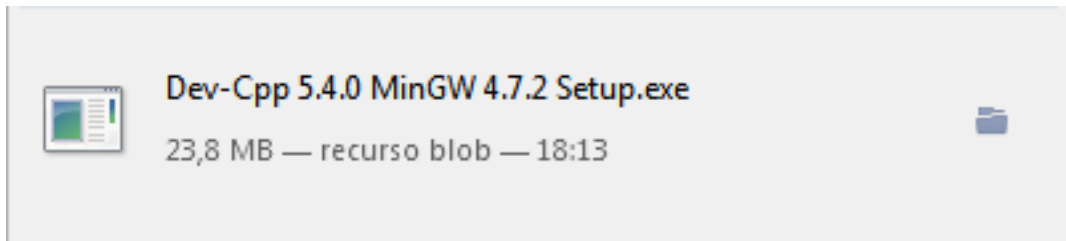
- Notepad ++ ([descargar](#))
- Code::Blocks ([descargar](#))
- Eclipse ([descargar](#))
- Visual C++ ([descargar](#))
- Dev-C++ ([descargar](#))

Si estás en Linux, puedes usar:

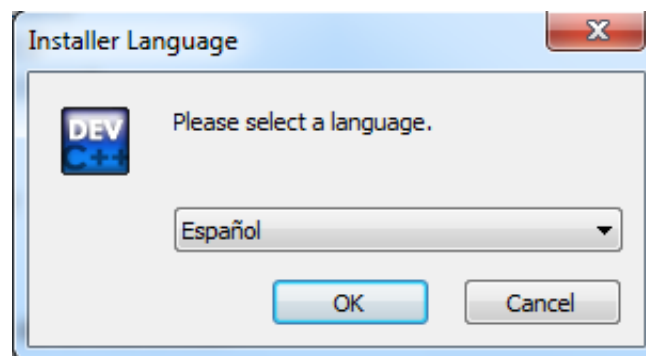
- Gedit ([descargar](#))
- Kate ([descargar](#))
- Code::Blocks ([descargar](#))
- Geany ([descargar](#))
- KDevelop ([descargar](#))
- Eclipse ([descargar](#))
- Notepad++ ([descargar](#))

¿Cómo instalar Dev-C++?

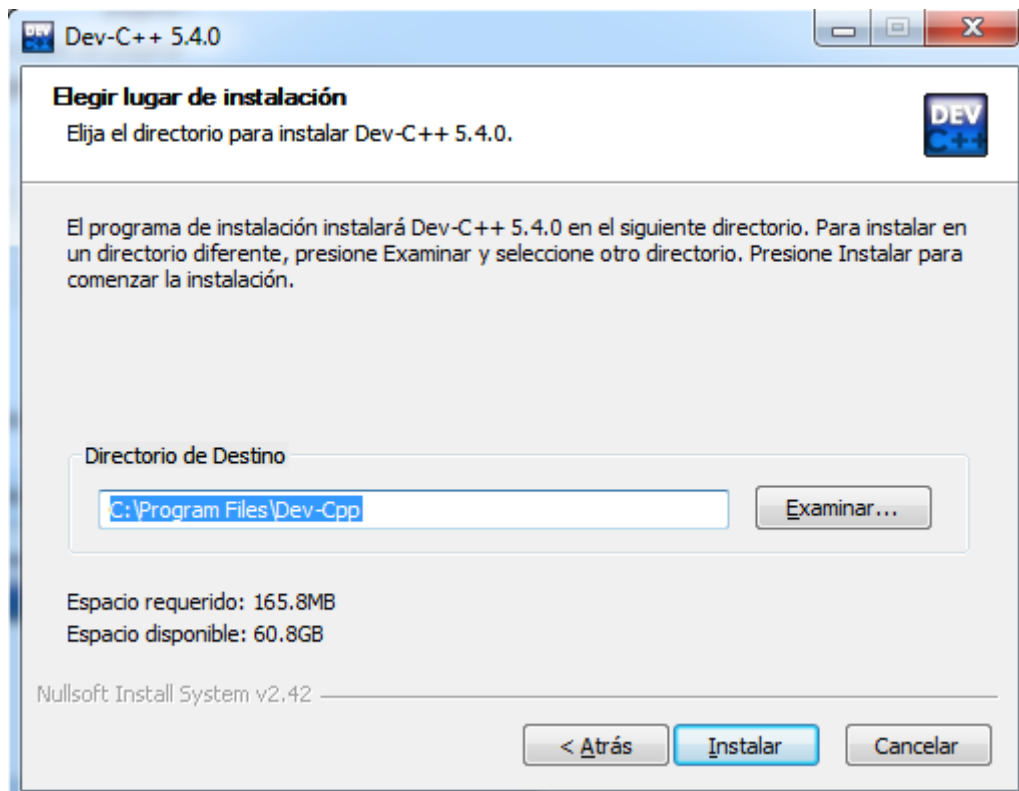
Descargamos el Dev-C++ de la web oficial.



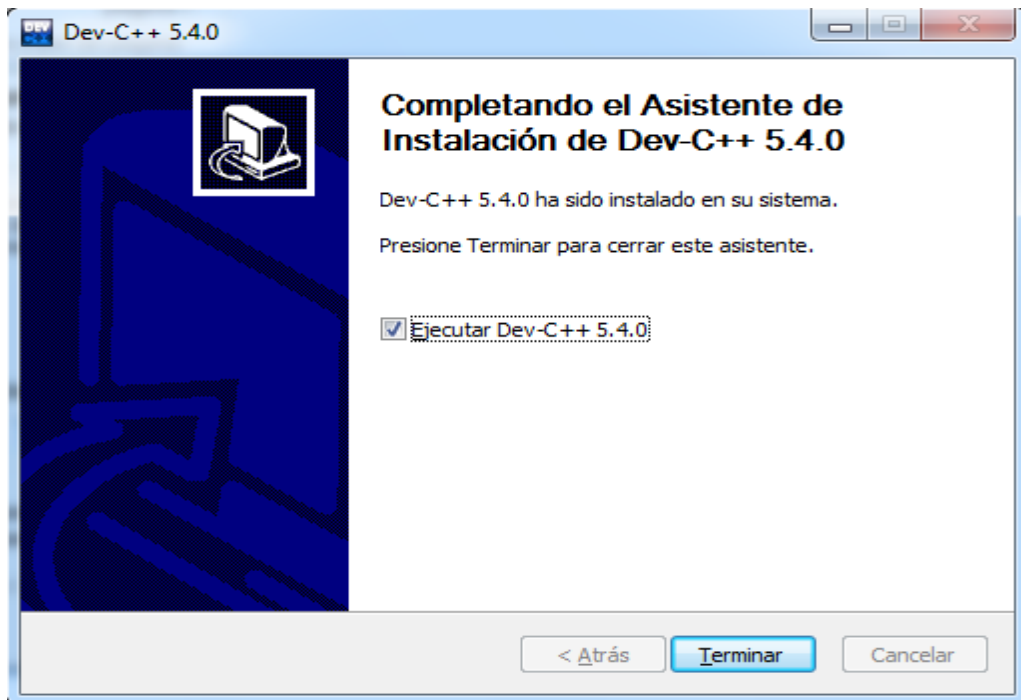
Abrimos el archivo, le damos click a ejecutar e indicamos el lenguaje.



Aceptamos el acuerdo de licencia, click en siguiente, y marcaremos el directorio de destino para el Dev-C++. (Necesitan un espacio libre de 165.8MB).



El último paso, terminar la instalación, agradecerle al asistente y ejecutar el Dev-C++.



¡Hola mundo!

Por fin haremos nuestro primer programa.

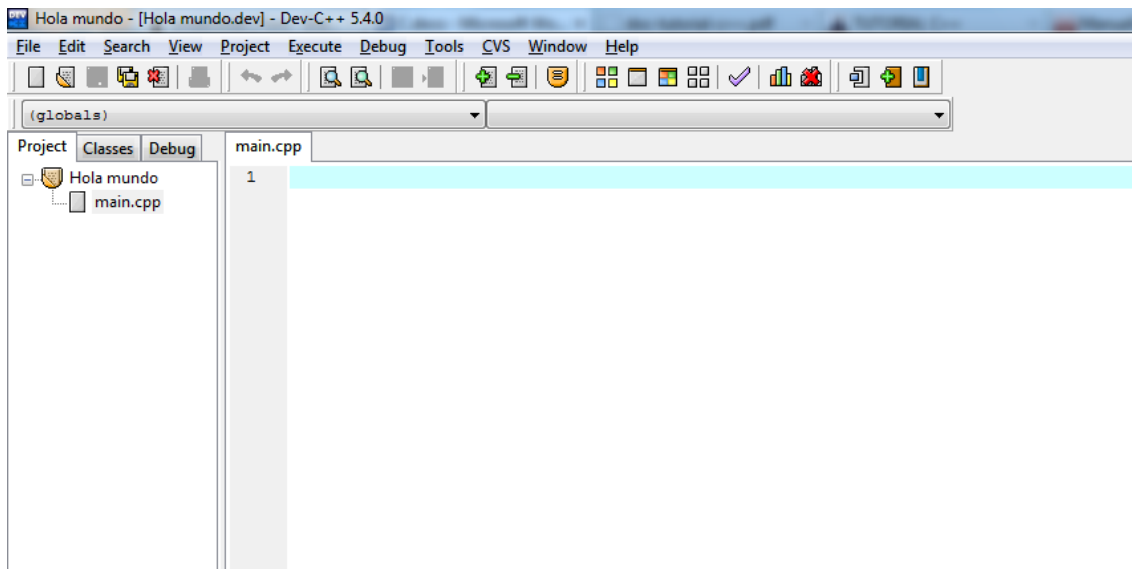
Abriremos Dev-C++ e iniciaremos un nuevo proyecto.

File >> New >> Project

Archivo >> Nuevo >> Proyecto

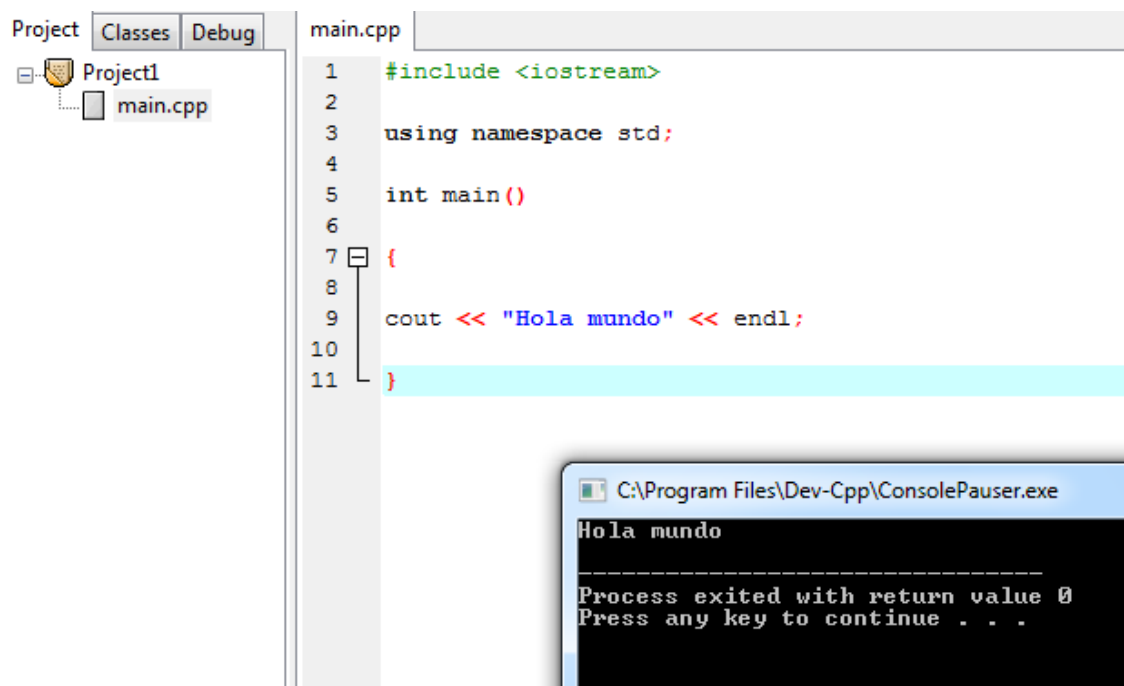
Seleccionaremos un "Empty Project", o bien un "Proyecto Vacío", que es una hoja vacía, para que nosotros insertemos el código.

Nombraremos al proyecto "Hola mundo", guardaremos, y después desplegaremos el menú de "Hola mundo" para ver la hoja vacía, guardaremos y la nombraremos "main.cpp". (Donde 'cpp' es la terminación de C++).

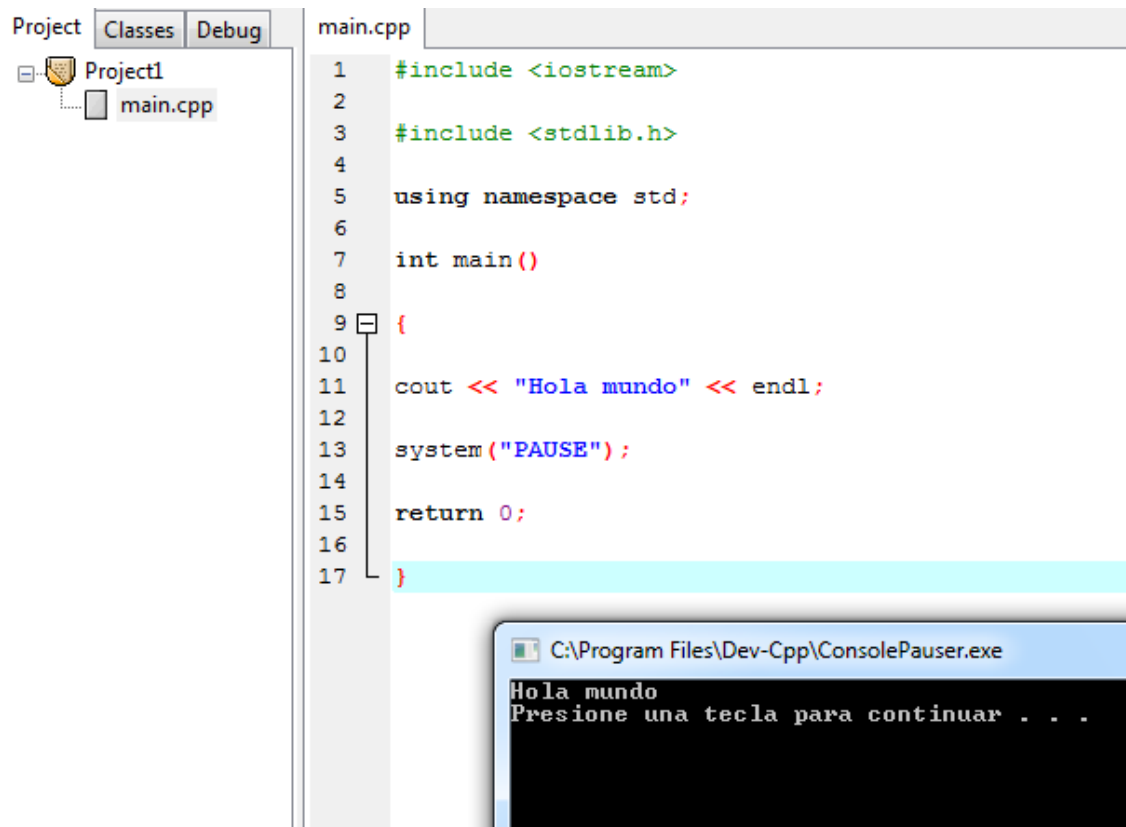


Con F9 compilas, con F10 ejecutas el programa y con F11 ejecutas y compilas.

La estructura de un “Hola mundo”, sería así;



O bien.



Recomiendo más este segundo para iniciados por la facilidad de sintaxis.

Variables

Estos son los principales rasgos de C++.

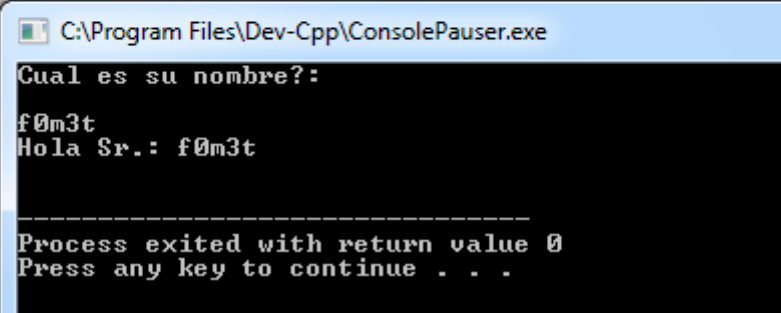
<i>Datos enteros</i>	char	signed char	unsigned char
	signed short int	signed int	signed long int
	unsigned short int	unsigned int	unsigned long int
<i>Datos reales</i>	float	double	long double

Donde 'char' es un carácter, 'int' es un número entero, mientras que 'float' es un número de punto, 'signed' se refiere a un número positivo o negativo, mientras que 'unsigned' a un número no negativo, 'double', 'long', y 'short', tienen el mismo modo de uso.

Caracteres

Tipo **char**: La variable tipo char contiene un único carácter y se almacena en un byte de 8 bits. Ejemplo de código char;

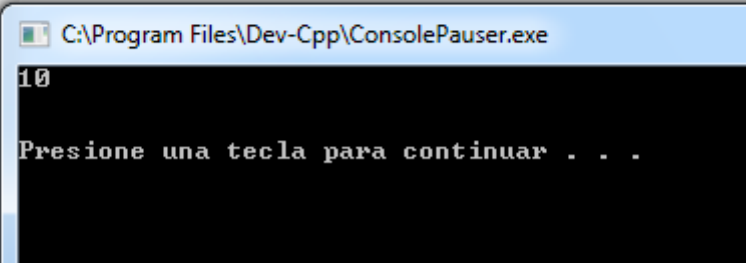
```
1 #include<stdio.h>
2
3 int main()
4 {
5     char a[20];
6     printf("Cual es su nombre?: \n\n");
7     scanf("%s", &a);
8     printf("Hola Sr.: %s\n\n",a);
9 }
```



Tipo **inf**: La variable tipo inf se almacena en 4 bytes de 32 bits, también conocida como variable entera.

Ejemplo de código de variable **entera**:

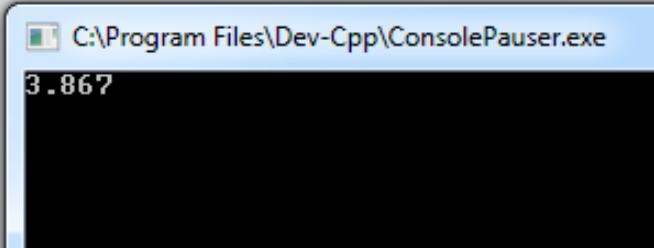
```
1 #include <iostream>
2 #include <stdlib.h>
3
4 int num = 10;
5
6 using namespace std;
7
8 int main()
9 {
10     cout << num << "\n\n" << endl;
11     system("PAUSE");
12     return 0;
13 }
```



Tipo float:

Las variables de tipo float contienen números de 'punto' o de 'coma flotante', que se le conoce cotidianamente como un número decimal. (3.1416)

```
1 #include <iostream>
2 #include <conio.h>
3
4 using namespace std;
5
6 int main()
7 {
8     float num = 2.567;
9     float num2 = 1.3;
10    cout << num+num2 << endl;
11    getch();
12    return 0;
13 }
```



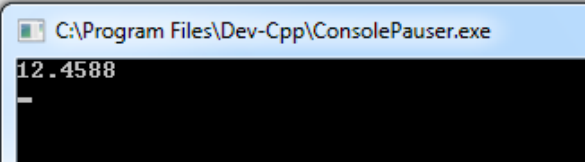
The screenshot shows a console window titled "C:\Program Files\Dev-Cpp\ConsolePauser.exe" with the output "3.867".

Tipo **double**:

Esta variable puede ser conocida como un 'float de doble precisión', como ya sabemos, el float no es de precisión, el double se almacena de la misma manera, pero este es mucho más preciso.

Ejemplo de código double;

```
1 #include <iostream>
2 #include <conio.h>
3
4 using namespace std;
5
6 int main()
7 {
8     double num = 12.458796767698769876;
9     cout << num << endl;
10    getch();
11    return 0;
12 }
```



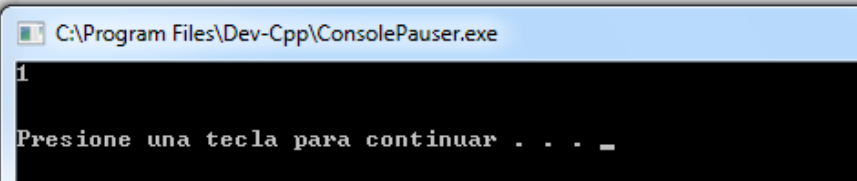
The screenshot shows a console window titled "C:\Program Files\Dev-Cpp\ConsolePauser.exe" with the output "12.4588".

Tipo **booleano**:

Esta variable sólo tiene 2 valores (true or false == 1 ó 0), esta variable sirve para calificar situaciones de lógica y en algunos casos para tomar una respuesta.

Ejemplo de código booleano:


```
1  #include <iostream>
2  #include <stdlib.h>
3
4  using namespace std;
5
6  bool variable = true;
7
8  int main()
9  {
10     cout << variable << "\n\n" << endl;
11     system("PAUSE");
12     return 0;
13 }
```



Operadores

Hay que tener en cuenta, ¿qué es un operador?

Un operador es cada uno de los símbolos que indican las operaciones a realizar.

<Número> + <número2>

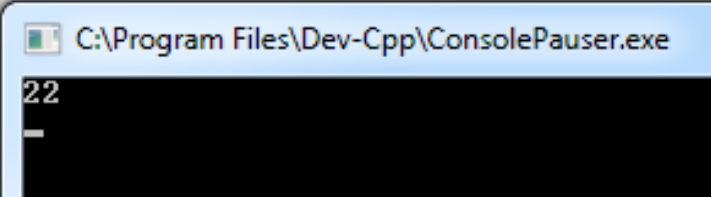
<Número> - <número2>

<Número> * <número2>

<Número> / <número2>

Estructura de una suma básica.

```
1  #include <iostream>
2  #include <conio.h>
3
4  using namespace std;
5
6  int numero = 12;
7  int numero2 = 10;
8
9  int suma;
10
11 int main()
12 {
13     suma = numero + numero2;
14     cout << suma << endl;
15     getch();
16     return 0;
17 }
```



The image shows a code editor window with a C++ program. The code calculates the sum of two integers, 12 and 10, and outputs the result, 22. Below the code, a console window titled 'C:\Program Files\Dev-Cpp\ConsolePauser.exe' displays the output '22'.

Ya todos conocemos los operadores básicos (+, -, *, /)

Y estas operaciones llevan una jerarquía de ejecución:

- Paréntesis
- Potencias y raíces
- Multiplicaciones y divisiones
- Sumas y restas
- Dos o más de la misma jerarquía u orden, se resuelve de izq. a derecha.

Y existen operadores unitarios, son un poco excepcionales ya que sólo trabajan sobre variables:

<Variable> ++ (post-incremento)
++ <Variable> (pre-incremento)
<Variable>-- (post-decremento)
--<variable> (pre-decremento)

Veamos un código donde explicaremos esta parte:

```

1  #include <iostream>
2
3  using std::cout;
4  using std::endl;
5
6  //inicio de main
7
8  int main ()
9  {
10 int c; //declaración de variable
11 //ejemplo de post-incremento
12 c = 10; //se le asigna un valor de 10 a c
13 cout << c << endl; //muestra 10
14 cout << c++ << endl; //muestra 10 y después hace el post-incremento
15 cout << c << endl << endl; // muestra 11
16
17 //ejemplo de pre-incremento
18 c=10; //mismo valor a c
19 cout << c << endl; //muestra 10
20 cout << c++ << endl; //hace el pre-incremento y después muestra 11
21 cout << c << endl; // muestra 11
22
23 return 0; //fin del programa
24
25
26 } //fin de main

```

Actividades

Con lo que has aprendido hasta ahora, podrás ejecutar estos sencillos pasos en base a reforzar tus conocimientos.

1. ¿Quién fue el creador de C++?
2. ¿Para qué se usa 'char'?
3. Hablando de precisión, ¿'float' o 'double'?
4. Escribe un programa que muestre un saludo.
5. Escribe un programa que sume dos números.
6. Escribe un programa que sume dos números y reste el resultado con un tercer valor.