**Initialization of variables.**

When the variables in the example above are declared, they have an undetermined value until they are assigned a value for the first time. But it is possible for a variable to have a specific value from the moment it is declared. This is called the *initialization* of the variable.

In C++, there are three ways to initialize variables. They are all equivalent and are reminiscent of the evolution of the language over the years:

The first one, known as *c-like initialization* (because it is inherited from the C language), consists of appending an equal sign followed by the value to which the variable is initialized:

*type identifier = initial\_value;*

For example, to declare a variable of type *int* called *x* and initialize it to a value of *zero* from the same moment it is declared, we can write:

|  |  |  |
| --- | --- | --- |
|  | int x = 0; |  |

A second method, known as *constructor initialization* (introduced by the C++ language), encloses the initial value between parentheses (()):

type identifier *(initial\_value)*;

For example:

|  |  |  |
| --- | --- | --- |
|  | int x (0); |  |

Finally, a third method, known as *uniform initialization*, similar to the above, but using curly braces ({}) instead of parentheses (this was introduced by the revision of the C++ standard, in 2011):  
*type identifier {initial\_value};*

For example:

|  |  |  |
| --- | --- | --- |
|  | int x {0}; |  |

All three ways of initializing variables are valid and equivalent in C++.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 | // initialization of variables  #include <iostream>  using namespace std;  int main ()  {  int a=5; // initial value: 5  int b(3); // initial value: 3  int c{2}; // initial value: 2  int result; // initial value undetermined  a = a + b;  result = a - c;  cout << result;  return 0;  } | 6 | [Edit & Run](https://cplusplus.com/doc/tutorial/variables/) |