

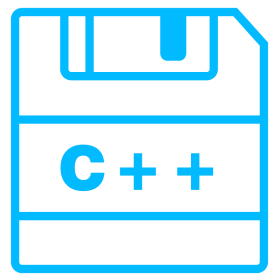
SEARCH

RESOURCES

CONCEPTS

1. Intro
2. CODE: Write and Run Your First C...
3. Compiled Languages vs Scripted L...
4. C++ Output and Language Basics
5. CODE: Send Output to the Console
6. How to Store Data
7. Bjarne Introduces C++ Types
8. Primitive Variable Types
9. What is a Vector?
10. C++ Vectors
11. C++ Comments
12. Using Auto
13. CODE: Store a Grid in Your Progr...
14. Getting Ready for Printing
15. Working with Vectors
16. For Loops
17. Functions
18. CODE: Print the Board
19. If Statements and While Loops
20. Reading from a File
21. CODE: Read the Board from a File
22. Processing Strings
23. Adding Data to a Vector
24. CODE: Parse Lines from the File
25. CODE: Use the ParseLine Function
26. Formatting the Printed Board
27. CODE: Formatting the Printed Bo...
28. CODE: Store the Board using the ...
29. Great Work!

Using Auto



You have now seen how to store basic types and vectors containing those types. As you practiced declaring variables, in each case you indicated the type of the variable. It is possible for C++ to do automatic type inference, using the `auto` keyword.

Have a look at the notebook below to see how this works.

Using auto

In your previous code, the type for each variable was explicitly declared. In general, this is not necessary, and the compiler can determine the type based on the value being assigned. To have the type automatically determined, use the `auto` keyword. You can test this by executing the cell below:

```
In [ ]: #include <iostream>
#include <vector>
using std::vector;
using std::cout;

int main() {
    auto i = 5;
    auto v5 = {1, 2, 3};
    cout << "Variables declared and initialized without explicitly stating type!" <<
}
```

Run Code

See Explanation

Loading terminal (id_kyiq0u0), please wait...

It is helpful to manually declare the type of a variable if you want the variable type to be clear for reader of your code, or if you want to be explicit about the number precision being used. C++ has several number types with different levels of precision, and this precision might not be clear from the value being assigned.

Practice

Practice using `auto` to declare and initialize a vector `v` with the value `{7, 8, 9, 10}`. If you have trouble, [click here](#) for help.

```
In [ ]: #include <iostream>
#include <vector>
using std::vector;
using std::cout;
```

Menu

Expand

On to an Exercise

Now that you have seen some exposure to variables and containers, test your knowledge in the next exercise! Before you go, be sure to have a careful look at the 2D vector example back in *Storing Vectors*, as you'll need this for the exercise.