Variables in C++

In this lesson, you will see how to declare and initialize variables in C++.

We'll cover the following

Variable declaration

A variable declaration means that we want the compiler to reserve a space for a data with the given name and type.

The basic syntax for declaring a variable in C++ is:

```
variable_datatype  variable_name ;
```

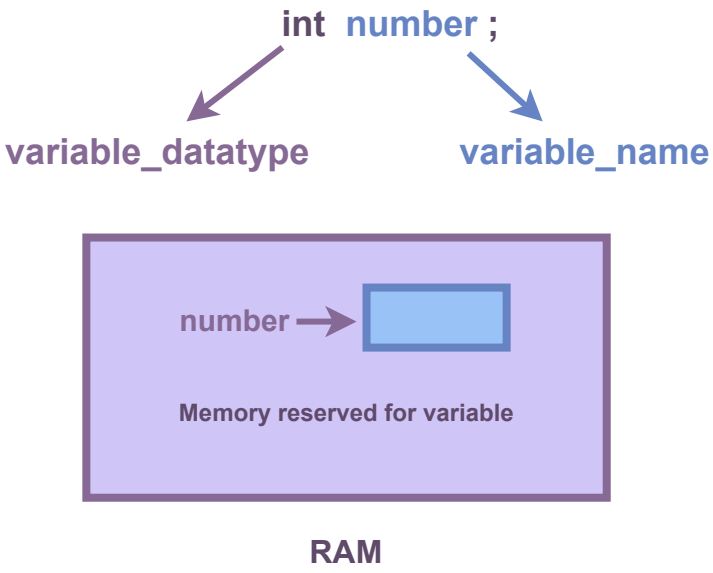
> 📝 **Note:** Don't worry about the data types yet. We will cover these in detail in the next chapter. For this chapter, we will just have to work with `int`. `int` is used to store an integer value in a variable. A variable declared with an `int` data type cannot store floating-point values.

To declare a variable that can store an integer value, we will write the following line:

```
#include <iostream>

using namespace std;

int main() {
  int number;

}
```

Variable declaration

In the above line, we have declared a variable with the name `number`, and it can store data of type `int`. Here, the `number` is an identifier.



Variable declaration

We can declare more than one variable in a single line.

```
int number1, number2, number3;
```

The above line declares three variables `number1`, `number2`, and `number3`.

Variable initialization

Variable initialization means to actually store value in the reserved space.

The basic syntax for initializing a variable in C++ is given below:

```
variable_datatype  variable_name = variable_value ;
```

In C++, we will write the following lines for initializing the variable of integer type:

```
#include <iostream>

using namespace std;

int main() {
  int number;
  number = 100;
```

```
}
```

Variable declaration and initialization in two steps

In the above code, we have defined the variable `number` that can store integer values, and it is assigned an initial value of 100.

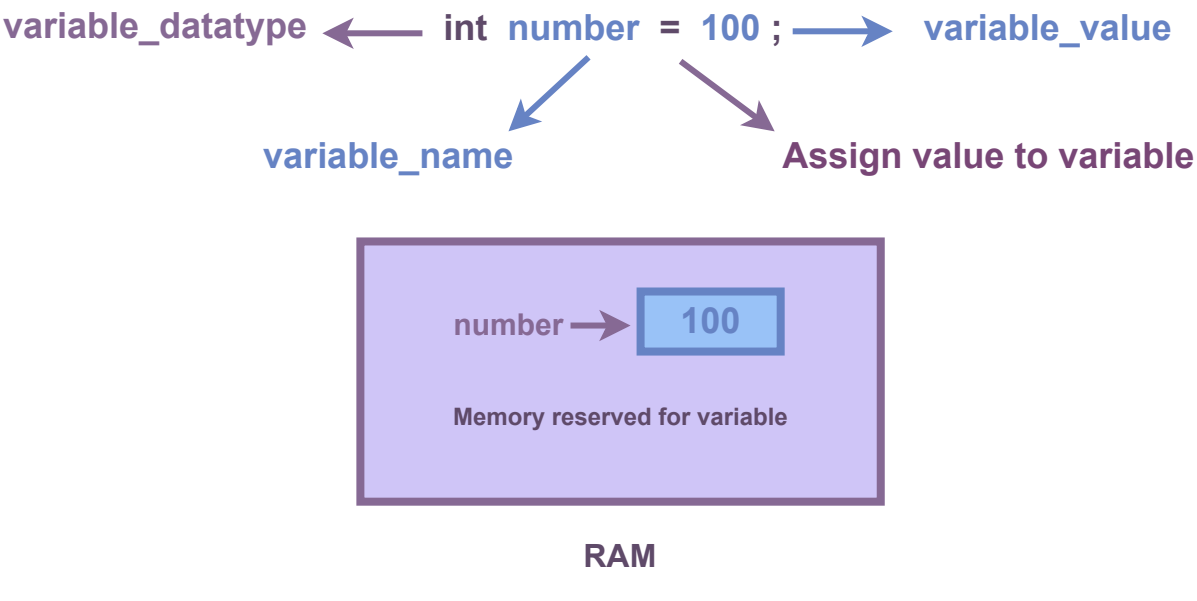
> **Do you know?** C++ is a **statically-typed** language. In a statically-typed language, a variable is declared with its type before its first use.

Variable declaration and initialization in one step

At this point, you're probably wondering, can't I just declare a variable and assign it a value in one go? The answer is yes! We can do this in the following way:

```
#include <iostream>

using namespace std;

int main() {
  int number = 100;
}
```

Variable declaration and initialization in one step



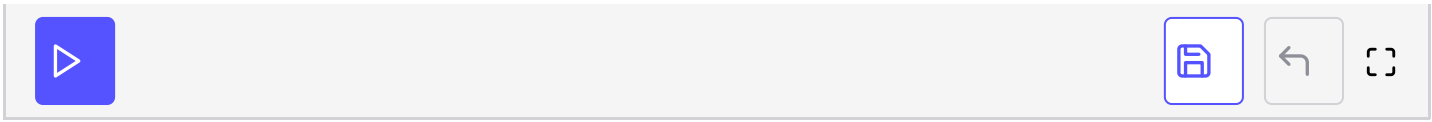
Variable declaration and Initialization

Example program

We can use a variable to keep track of the current amount in our bank account. Suppose you have $100 in your bank account. After some time, your friend transfers $20 to it. Now, the current amount is $120. Let's write a code in C++ that can keep track of your account balance.

Run the code below and see the output!

```
#include <iostream>

using namespace std;

int main() {
  // Declares a variable current_amount
  int current_amount;
  // Initialize a variable current_amount to 100
  current_amount = 100;
  // Prints the value of current_amount
  cout << "Your current amount is: " << current_amount << endl;
  // Updates the value of current_amount
  current_amount = 120;
  // Prints the updated value of current_amount
  cout << "Your current amount is: " << current_amount << endl;
}
```



Updating the value of a variable

Line No. 7: Declares a variable `current_amount` that will store the integer value

Line No. 9: Initially, there is $100 in a bank account. Therefore, we store **100** in variable `current_amount`

Line No. 11: Displays the value of `current_amount`

> To print the value of a variable, use `cout` followed by the insertion operator `<<` and variable name.

Line No. 13: When your friend transfers $20 to your account, the `current_amount` becomes $120. Therefore, we update the value of the `current_amount` to `120`, `changing` the value of a variable during the program execution.

Line No. 15: Displays the updated value of `current_amount`

Quiz

Q

In which of the following ways we can declare and initialize a variable `var` with a value of **1000**?

(You can select multiple correct answers)

Retake Quiz

Interesting so far? Let’s move on to the next lesson, where we will discuss identifiers in C++.