

Lecture 5

Variables, Operators, Libraries, Input/output

Instructor: Ashley Gannon

ISC3313 Fall 2021



Variables



What is a variable?

Short Answer

It is an object that represents a certain class of data

For example if you saw the expression $x + 5$ in a math class you would immediately know that x was a variable that represented a number.

In order to perform any kind of algorithm you must have some sort of data and that data is stored in what we call a variable.

The class or group of data that the variable might contain is called its type.

It might be useful to portray data stored in our computers in the following fashion:

Variable Name	Type	Value
x	int	2
y	int	3



Types

A few built in types that are available for you to use:

- **int** (represents the math class integer with bounds $[-2147483648, 2147483647]$)*
- **float** (represents the math class Real with precision 10^{-7})*
- **char** (represents a single character of text ie. 'a')
- **string** (We've already seen this one in our Hello World example. It represents a list of characters.)

*Values may vary from one computer to another.



Variable Declaration and Assignment

To declare a variable we state the type then the name.

For example if we typed `int x;` we would create a variable named `x` of type `int`

Variable Name	Type	Value
<code>x</code>	<code>int</code>	<code>?</code>



Variable Declaration and Assignment

To declare a variable we state the type then the name.

For example if we typed `int x;` we would create a variable named `x` of type `int`

Variable Name	Type	Value
<code>x</code>	<code>int</code>	?

To assign a value to a variable, we use the `=` operator. For example if we now typed `x = 5;`, the the variable `x` would contain the value 5.

Variable Name	Type	Value
<code>x</code>	<code>int</code>	5



Variable Declaration and Assignment

To declare a variable we state the type then the name.

For example if we typed `int x;` we would create a variable named `x` of type `int`

Variable Name	Type	Value
<code>x</code>	<code>int</code>	<code>?</code>

To assign a value to a variable, we use the `=` operator. For example if we now typed `x = 5;`, the the variable `x` would contain the value `5`.

Variable Name	Type	Value
<code>x</code>	<code>int</code>	<code>5</code>

It is also possible to declare a variable and initialize it in the same line. For example: `int x = 5;`, would give us the same result.

Variable Name	Type	Value
<code>x</code>	<code>int</code>	<code>5</code>



Exercises

Try Compiling and running after every step, so that you can spot your mistakes a little easier.

- Create a file called `Types.cpp`
- Declare and initialize an integer variable named `x` to the value 5;
- Declare and initialize a float variable named `y` to the value 1.5;
- Delcare an int variable named `z`;
- Initialize `z` to the value `x/y`.
- Print the value of `z` with the command `printf(z) ;` Why doesn't it work?
- Now try printing the value of `z` with the command `printf("%f", z) ;`
- What's the value of `z`? Why do you think that is?
- What if we change the `f` to a `d`? `printf("%d", z) ;`



Miscellaneous Notes on Variables

- Variable names are case sensitive. Thus `patrick` is registered as a different variable than `Patrick`.
- When assigning char's you must put a single quote around the character.
`char letterA = 'a';`
- When assigning a string you must put a double quote around the group of characters, ie. "Hello World!".



Booleans

Boolean

A variable that covers the state space of true or false. Which in terms of ones and zeros is either 1 or 0. A small caveat of c++ is that it often treats everything that is not a zero as true. This can be useful or it can be annoying.

A declaration and initialization might look like:

```
bool performCalculation = false;
```



Operators

An operator is a simple algorithm that takes some input (usually two values) and returns another value as output. The mathematical operators are good examples and behave as expected.

Comparison operators

< less than	<= less than or equal to	== equal to
> greater than	>= greater than or equal to	!= not equal to

logical operators

&& Represents 'AND'

|| Represents 'OR'

! Represents 'NOT'



Conditions

Condition

A statement that evaluates to a boolean value.

Conditional statements: True or false?

1 $1 < 2$

2 $2 >= 3$

3 $1 != 2$

4 $1 == 2$

5 $1 = 2$



Conditions

Condition

A statement that evaluates to a boolean value.

Conditional statements: True or false?

1	$1 < 2$	True
2	$2 >= 3$	False
3	$1 != 2$	True
4	$1 == 2$	False
5	$1 = 2$	Invalid notation



Tokens

Token type	Description/Purpose	Examples
Keywords	Words with special meaning to the compiler	<code>int, double, for, auto</code>
Identifiers	Names of things that are not built into the language	<code>cout, std, x, myFunction</code>
Literals	Basic constant values whose value is specified directly in the source code	<code>"Hello, world!", 24.3, 0, 'c'</code>
Operators	Mathematical or logical operations	<code>+, -, &&, %, <<</code>
Punctuation/Separators	Punctuation defining the structure of a program	<code>{ } () , ;</code>
Whitespace	Spaces of various sorts; ignored by the compiler	Spaces, tabs, newlines, comments



Keywords

Some keywords are covered by the datatypes. Some other important ones to know for now are:

- `if`: executes a chunk of code if a given condition is true (can execute other code with `else` if condition is not true)
- `for`: allows the program to execute a number of statements in sequence
- `while`: allows the program to execute statements until a condition is satisfied

Much more on these later in the course!



Identifiers

Identifiers are things that come from outside the language itself, i.e. from user-written code. For example, the `cout` and `endl` identifiers:

```
cout << "Hello world!" << endl;
```

These actually come from a library.



Libraries

Libraries can be input to a program by means of a *header file*. This file has a special extension, `.h`. For example, the header file `iostream.h` gives us the `cout` and `endl` identifiers. Some other popular ones are

- `stdio.h`
- `iostream.h`
- `stdlib.h`
- `math.h`
- `string.h`

You can think of these as augmenting the C++ language. You can write your own header files! Put them into your program with `#include <filename.h>`



Input and Output

Instead of `printf()`;

```
cout << variable;
```

Used to print to the terminal. It can be used to print many different data types.

```
cin >> variable;
```

Takes input from the screen and puts it into the following variable.

These commands are available in the `iostream` package, so be sure to use

```
#include <iostream>
```

at the top of your program.



Class Exercise - Loops!

- 1 In your ClassActivity folder, write a program `HelloWorld5_for.cpp` code that prints "Hello World" 5 times using a for loop.
- 2 In your ClassActivity folder, write a program `HelloWorld5_while.cpp` code that prints "Hello World" 5 times using a while loop.

