CS121: Things you already know, but now in C++

Dylan A. Shell

January 22, 2018

Most of the things you're already familiar with have similar forms in C++. The following is not exhaustive, but covers just the essentials to get off the mark.

Basic operators

Arithmetic: `+ - * / %` addition, subtraction, multiplication, division, remainder/mod.

(All but the last work for integer and floating point types.)

Logical: `&& || !` and, or, not.

Relational/comparison: `< <= > >= == !=` less than, less-than-or-equal-to, *etc*, equal-to, not equal-to.

Increment: `++x --x x++ x--` prefix-(increment/decrement), postfix-(increment/decrement).

Reassignment: `+= -= *= /=` is like `++` but parameterized, e.g., `x += 5;` increments by 5.

Documenting code

Comments: `/* multi-line\n block comment */` or `// remainder of the line`

Basic I/O

Printing: `cout << "Hello " << name << endl;` write to standard output.

Input: `cin >> x;` read standard input.

Variables

Numbers: `int i = -4;` for integers; and `float f = 3.14;` for floating point numbers.

Booleans: `bool b;` can be equal to `true` or `false` only, the logical operators can be applied to them.

Alphanumeric characters: `char c;` for ASCII characters. Note the single quote marks: `c = '8';`

(Useful escape sequences: `\n` = newline, `\t` = tab, `\b` = backspace, `\\` = backslash, `\0` = null.)

Branching

Single: `if (c < 0) one_stmt();` or `if (a >= 12) { stmt_0(); stmt_1(); ··· }`

Double: `if (x != 13) do_a_thing(); else do_some_other_thing(x);`

(It is unusual not to have have braces for blocks.)

More: `if (y > 1) one_thing(); else if (y < -4) second_thing(y); else final_thing(y*2);`

(This is the `if-else` rule applied twice, not a new construct.)

Looping

All of the following have single statement variations without braces also.

Iteration: `for (int i = 0; i < 10; i++) { stmt_0(); stmt_1(); ··· }`

(The variable needn't be declared in the first statement, if it has been declared earlier.)

While: `while (!exit_time) { stmt_0(); stmt_1(); ··· }`

Do/While: `do { stmt_0(); stmt_1(); ··· } while (still_working);`

(Importantly, the `while` and `do/while` differ with regard to when the loop condition is checked.)

Organization

Blocks: `{ int i = 0; /* local scope */ }` whitespace is ignored (but indenting is helpful).

Statements `x = 5.0; ;` including the vacuous one, which can be a source of errors.

Preprocessor directives

Insert file's contents: `#include <abc>` standard C++ library header file.

Insert file's contents: `#include <abc.h>` standard C library header file.

Insert file's contents: `#include "abc.h"` custom C/C++ library header file.

Replace text: `#define MAXSTR 100` substitutes this string (flag `-E` will reveal this process).

Naming

Namespaces: `using namespace std;` saves us typing `std::cout << std::endl;`

Pitfalls

Uninitalized variables: `int i; ··· i = i+10;`

Extraneous semi-colon: `for (i = 0; i < 10; i++); ···`

or `while (t > 10); ···`

or `if (t > 10); ···`