ACM Computing Seminar C++ Guide

Matt Hancock

Outline

Introduction

Welcome to the ACM C++ guide, a manual for quickly learning C++ for mathematical and scientific computing applications. The goal of this guide is not to make you a C++ expert, but to quickly teach you enough of the C++ fundamentals and design patterns to help you off the ground. If you should like to go beyond this guide, a few references are listed below.

This guide is split into sections. The sections are ordered in such a way that topics in later sections often depend on topics from previous ones. You'll find code snippets peppered throughout the sections. At the end of each major section, you'll find exercises for practice.

A little about the language

Before you dive in, here is a little about the C++ programming language:

C++ is an extension of the C programming language. Both C and C++ are statically-typed and compiled languages, which means that the type of variables used in your source code must be declared explicitly and is checked at when the program is compiled

Getting started

The particular programming tools that you choose to use will likely be largely influenced by the operating system that you use. We will use free tools (often developed for GNU / Linux systems) in this guide. These tools are mostly available in other operating systems as well. For example, on Windows, you could use Cygwin, or install a dual boot with some Linux distribution (e.g., Ubuntu). On the other hand, MAC OSX, being a BSD-derived system, has many of the required tools already available (although, a command line utility, Brew, makes building and installing other tools very simple). In the following two sections, we'll talk about the two basic types of software that you'll need to begin writing C++ programs.

Text editors

The text editor that you choose to use should be be any program capable of editing plain text files. However, you may find that it's more productive to write in an editor that offers features such as syntax highlighting, code-completion, bracket-matching, or other features. Here are some popular free text editors:

- Atom is a recently open-sourced GUI editor which some have

Data types

As we discussed previously, you must explicitly declare the type of a variable. So, in this section, we'll talk about the main variable types you'll use, namely boolean, integer, floating point types. In the section on **??**, we'll discuss how to build our own custom data types.

Boolean

A boolean data type is either `true` or `false`. There are a number of operators between these types, illustrated in the code snippet below (note that lines starting with `//` are comments are ignored by the compiler):

```
1  bool a,b,c; // Declare the type of variables a, b, and
2  a = true;
3  b = false;
4
5  // !  is logical negation when applied to a single varia
6  c = !a; // c is false.
7
```

Control structures

Conditionals

Often a code block should only be executed if some condition is true. Below, we generate a random number between 0 and 1; print the number; and, print whether or not the number was greater than 0.5.

```
 1  #include <iostream>
 2  #include <stdlib.h>
 3  #include <time.h>
 4
 5  int main() {
 6    // Seed the random number generator based on the curre
 7    srand(time(NULL));
 8
 9    // rand() produces a random integer between 0 and RANI
10    double num = rand() / ((double) RAND_MAX);
11
12    std::cout << "num: " << num << "\n";
```

Functions

Object-oriented programming