CS 100 Basics

In CS 100, we use the C programming language to write a number of basic programs. This includes in-class activities, laboratories, projects, and coding during exams. You will complete these programs on your laptop. This document walks you through the basics of configuring your laptop for CS 100. It includes:

1. Installing a C compiler on your laptop (if you do not have one)
2. Editing C source code files and manipulating files/directories
3. Using the cs-intro.ua.edu server for program testing
4. Submitting labs/projects/exams in CS 100

CS100 takes a pragmatic approach in teaching programming. The only way you truly learn a subject is by practice, lots and lots of practice. This is not a course where you can sit passively, listen to a lecture, and then be able to do what is needed. You cannot get better at programming by watching your instructor or fellow students write code. You only improve when you take the time to do it yourself. We will make sure you get lots of practice with writing code in CS 100.

Installing a Unix-style system on your laptop to compile and run C programs

I have a Mac.

- Figure out how to open a Terminal window (you may want to put Terminal Application on your dock).
- Test to see if you have ***gcc*** installed. Enter the following command into the terminal window: **`gcc`**
  - If you get a message saying *The gcc command requires the command line developer tools. Would you like to install them now?* then click the Install option.
- To check the installation was successful, type **`gcc`** in the terminal window and hit return. If you see an error message similar to **`clang: error: no input files`**, you are all set. This means the **gcc** command/compiler exists on your system and you need to enter a C program for compilation.

I have a PC running Windows. In this case, we recommend you install Cygwin, a Unix-like environment for Windows.

- Browse to http://www.cygwin.com and click on the *Install Cygwin* link near the top of the teal-colored box on the left-hand side of the page, then follow the instructions. Make sure you use the 64-bit version of the setup program if you have a 64-bit computer.
- Answer all the questions using the default answers. You are asked to select a location from which to download. Pick any location; some will be slow and some will be fast. Hopefully, you'll pick a fast location. At some point, you see a Select Packages screen. Click on plus signs next to:
  - **`Devel`** and click on the Skip column entry next to the **`gcc-core`**, **`gcc-g++`**, **`gdb`**, and **`make`** entries. (By default, the installation will skip these packages. By clicking the skip, it becomes unskip, and it will install the packages.)
  - **`Editors`** and click on the Skip labels next to the **`emacs`**, **`vim`**, and **`vim-common`** entries.
  - **`Net`** and click on the Skip label next to the **`openssh`** and **`rsync`** entry.
- Finally, click on the **`Next`** button in the lower right corner and continue accepting defaults.
- Open a Cygwin terminal (there should be an icon on your Desktop).
- Type **`gcc`** in a Cygwin terminal window. If you see **gcc: fatal error: no input files**, you are all set. This means the **gcc** command/compiler installed and you need to enter a C program for compilation.

CS 100 Editing and Manipulating Files

The environment that we use for software development (a terminal window) assumes that you have some basic understanding of the file system structure for your laptop and also know a few of the basic commands used to manipulate files (and directories) on your laptop. A cheat sheet of the basic terminal commands found in Mac's terminal window or a Cygwin window can be found in the appendices (the Unix/Linux Command Reference page).

Editing Files

The default editor that your instructor will use is **vim**. This editor has been around for many years, and is available on a wide variety of platforms. A good vim tutorial can be found at the site listed below. A vim cheat sheet can be found in the appendices too.

http://heather.cs.ucdavis.edu/~matloff/UnixAndC/Editors/ViIntro.html

Please realize that a choice of editor is entirely yours. While your instructors use vim, you are free to select another editor if you want. When you get into future courses, you will learn a more complete IDE (integrated development environment) for your coding projects.

One other editor to consider for CS 100 would be **Atom**, https://atom.io/. This editor is available for both the Windows and Mac environments. It is a simple environment that provides for clean editing of C source code.

Manipulating Files

One of the basics concepts you will need in CS 100 is an understanding of the file system on your laptop. You need to have a working knowledge of where the system stores files (by default) and how to move files around.

On a Mac:

- The default (initial) file location is **`/Users/`*`userid`***
- Move files between any directories using either UNIX commands (**`cp`**, **`mv`**, …) or via Finder

On a Windows-based machine using Cygwin:

- The default (initial) file location is **`c:/cygwin64/home/`*`userid`***
- Move files between any directories using either UNIX commands (**`cp`**, **`mv`**, …) or via Windows Explorer.

CS 100 – Testing your programs with the cs-intro.ua.edu Server

There are three specific things you need to understand about the cs-intro.ua.edu server for CS 100.

1. Why we have a separate server for grading CS 100 programs
2. How to move your work back and forth between your laptop and this server
3. How to use cs-intro to check your programs

A Dedicated Server for Grading in CS 100

While one would like to think that a C compiler behaves exactly the same on all operating systems, this is not the case. When different software vendors write a compiler for a Mac or Windows system or Linux, there are some areas where the ANSI standard does not dictate exactly how to accomplish a given task. As a result, a program that runs on one system might not function the same on another system. This gets into the software engineering issue of portability of code. A common example of this is some compilers are designed so that integer variables have a default value of zero, while other compilers do not initialize variables (their initial value is unspecified).

In CS 100, we recognize that different students are developing in different environments. We want you to be able to use your own laptop and work on assignments whenever/wherever you have the opportunity. However, we also need a common platform for grading these assignments.

As a result, all programming assignments in CS 100 will be graded using the server cs-intro.ua.edu. All students in the course have access to this server, so you can test your programs on the **cs-intro** server before submission.

Please note that if you are working from off-campus, you must first establish a VPN connection to UA. VPN information be found at http://oit.ua.edu/oit/services/virtual-private-network-information/

Moving Files To and From the CS 100 Server

On your local machine, change to the directory where you file(s) exist and type

```
rsync filename userid@cs-intro.ua.edu:filename
```

After issuing this command, the file specified should also exist on the cs-intro.ua.edu server. You can use **ssh** to log into the cs-intro server, compile your program, and test it on the server.

If you already know how to use ftp, you can use sftp in one terminal to move your programs between your machine and the cs-intro server, and use ssh in another terminal to compile and execute your programs.

Accessing the CS 100 Server

All students have access to **cs-intro.ua.edu**. From the command line prompt, in either a Cygwin window on a Windows system or a terminal window on a Mac, type **ssh username@cs-intro.ua.edu** where username is your myBama userid. Note that, when entering your myBama password, you will not see the keystrokes echoed on the display. Once you are logged in, you should see a standard prompt and Unix environment. You can issue any of the basic commands that you use in your terminal window on your laptop, including **vim** to edit files. Note that the Atom editor will not work on the cs-intro server. To log off of the **cs-intro** server, simply type **`exit`**.

Submitting Labs/Projects/Exams in CS 100

For each lab, project, and exam in CS 100, we ask you to create a directory and do your work in that directory. For example, we ask you to create a **CS100** directory in your home directory for the course and then create sub-directories within it for each of your labs and projects. For each directory name, please use lowercase for all the letters and leave no spaces between letters and digits.

```
.../cs100
.../cs100/project1
.../cs100/project2
.../cs100/lab1
.../cs100/lab2
```

For a given task, once you have completed your testing and are ready to submit your work, compress the corresponding directory into a **`.zip`** file on your local system and submit that single (compressed) file. To do this:

First, on your local machine, compress your directory into a single (compressed) file. To do this:

- PC: Using Windows Explorer, right click on the specific directory and select "Send To" and then "Compressed (zipped) folder."
- Mac: Using Finder, use a secondary click on the specific directory and then select "Compress *foldername*"

Second, once you have a compressed file of a particular directory containing your source code, submit that file to Blackboard.

Two quick observations regarding your submissions:

1. We are only interested in the source code file (the **`xxx.c`** file). If there are other files in your directory, then we simply ignore those files. When grading, we re-compile your code and run it on a set of test cases.
2. We expect everyone to submit a **`.zip`** file. We will not accept submissions that use another format such as the **`.rar`** format.