Lab 1: Warmup

Goals

The purpose of this lab is to help you become acquainted with the various tools you will be using throughout this course. During this lab exercise you will:

- Acquire a CS account
- Become familiar with the UNIX/LINUX workstations
- Become familiar with a text editor of your choice
- Setup and familiarize yourself with the LC-3 simulator
- Familiarize yourself with your preferred C environment
- Meet the CSAC TA's and pass-off this lab

Setting Up and Using Your Account

Setting Up Your Account

The first thing you need to do to get started is to create a CS computer account. If you don't have one already send an email to Larry Hughes with the following:

1) First and Last Name

2) Your NOVELL/AD login name (it looks like abcd1234 = four characters + four numbers)

3) A valid uidaho.edu email address (vandals or employee)

4) If you are a CS or CE major.

Using this account, you will be able to complete all the labs using the CS computer facilities. However, you are strongly encouraged to setup a home or laptop computer with the necessary software.

If you are not already familiar, you should take some time to poke around and become familiar with Linux.

Working Remotely

Please read the guide on how to remotely login to computers and on how to transfer files (middle part of webpage). The student server is wormulon.cs.uidaho.edu. This knowledge will come in very handy for many of you, because it allows you to do your

work from virtually any computer. However, please keep in mind that all labs must be passed off in person.

Choosing a Text Editor

What's a Text File?

In these labs we will work with *text files* quite a bit. A text file is a file where the contents are just plaint text characters (letters, numbers, punctuation marks, returns, etc.). These files contain no formatting or header information, unlike more complex formats such as Microsoft Word documents. Programs in nearly all languages are written using this plain text format.

Text files types are distinguished by the *file extension*. File extensions are the last few letters of the filename after the period. For example a C code filename ends with the ".c" extension. Assembly language files in this class will end with the ".asm" extension.

By default, Microsoft Windows hides the extension. If you use Windows, it will be much easier to identify text files if you have Windows display the extensions.

To enable this feature in Windows XP, follow these steps:

1. Open "My Documents"
2. Under the "Tools" menu, click "Folder Options"
3. Click the "View" tab
4. Uncheck the box labeled "Hide extensions for known file types"
5. Click the "Apply" button.

In Windows Vista, follow these steps:

1. Open "Documents"
2. Click the layout button (the window icon with the small down arrow in the upper left corner of the window)
3. Click "Folder Options..."
4. Click the "View" tab
5. Uncheck the box "Hide extensions for known file types"
6. Click the "OK" button.

Emacs and Vim

Since we will be using text files so much, you should become familiar with a good text editor. The two most popular editors for serious programmers are Emacs and Vim. Both are available on just about every operating system and both are installed on the Linux and Unix machines in the computer labs. You can also download Windows versions of Emacs and Vim.

Both of these text editors are very different from normal editors, such as Word, in that they do not rely on the mouse. However, if they are learned well, you will be able to edit text files much more quickly and comfortably. They also support powerful features, such as syntax highlighting, search and replace, auto-completion, and much more.

So which is better, Emacs or Vim? This is an ongoing religious war with no clear winner. I will say that Emacs has some very powerful features that make programming in some languages much faster and it tends to be easier to learn. On the other hand, Vim is more comfortable for your hands and is more commonly used.

If you choose to use Emacs or Vim, **take the time to learn important commands**. If you find yourself using the mouse and arrow keys often in Emacs or Vim then you are **NOT** using the program correctly and you need to learn the proper commands.

Other Text Editors

There are several free Windows editors that are very good for programming, such as PSPad, Notepad++, Programmer's Notepad, ConTEXT, and others. These editors can be downloaded to your personal Windows Machines. On Linux, KWrite, GEdit, Nedit, and others are often available.

These are all traditional editors that use the mouse and keyboard, much like Microsoft Word. However, they also support programming features such as syntax highlighting, advanced search replace, and more. Since they rely heavily on the mouse, they tend to be slower to use and generally do not support sophisticated editing features.

Emacs Introduction

Here is a brief introduction to Emacs. If you think you would like to learn Emacs, follow the steps below.

- **Step 1:** Go to the CS lab and log onto a machine.
- **Step 2:** Read this brief introduction to Emacs (bottom of webpage).
- **Step 3:** Open up an xterm or shell window as explained above.
- **Step 4:** Start up Emacs by typing the word **emacs** at the prompt in the shell window. Once you have started Emacs, try some of the commands you read about in the brief tutorial.
- **Step 5:** Emacs offers its own tutorial. You can access this tutorial by typing **ctrl-h t** (type the **h** key while holding down the **Ctrl** key, then let up the control key and type the **t** key) in the Emacs window. The tutorial is very long, but it does contain a lot of useful information about Emacs. It is a great resource if you ever want to learn more. When you exit the tutorial, Emacs will ask you if you want to save it, you don't need to so just exit Emacs by typing **Ctrl-x Ctrl-c**.

- **If you get stuck:** Until you become familiar with Emacs , it is possible to get the editor in a mode that you don't understand and things just don't work right. If this happens, just type **Ctrl-g** (type the **g** key while holding down the **Ctrl** key) a few times. You can type **Ctrl-g** at any point; when you type this, Emacs will beep at you and should return to its normal state. Also, don't be afraid to ask someone for help. Ask a TA or even the student next to you if you get stuck. We are all in this education process together.

Here is an EMACS reference card to help you remember the various commands. You may print it for your reference. Many Emacs tutorials are also available online.

Vim Introduction

If you think you would like to learn Vim I suggest you find a good tutorial online.

Skills Test

After you have chosen an editor, take the time to learn to use it. At a minimum for this lab, you should master the following skills:

- Starting the editor
- Quitting the editor
- Creating a new text file
- Opening an existing text file
- Saving changes to a text file
- Editing text

Be prepared to demonstrate these skills, in person, to a TA.

LC-3 Simulator Setup

As a student, you have a lot of flexibility in where you work and under what Operating System. You may setup the LC-3 Simulator under Linux or Windows at home (It is already setup in the CS Lab). However, you must be able to demonstrate that you can run it in the Linux environment.

- Linux (Download .tgz file with simulator from here)
  - To assemble code just type lc3as filename (don't need to put .asm extension). This creates filename.obj which can then be called by the simulator.
  - To convert code from hexadecimal just type lc3convert filename (don't need to put .hex extension). This creates filename.obj which can then be called by the simulator.
  - To run the simulator (non gui), type lc3sim to start the simulator. Then type file lc3os (you will have to assemble this the first time – you can find a copy lc3os.asm). After the file is loaded type continue. Then type file

filename (this is the name of the file you want to simulate – you will always load the OS first). After the file is loaded type continue. You should then see the output (if any) of your program.
    - To run the simulator (gui) is the same process as the non-gui but there is a button to load the .obj files and a button for continue.
- Windows (Download .exe file with editor and simulator from here)
    - To assemble or convert a file open lc3edit.exe and a window will appear. Load the appropriate file (filename.asm or filename.hex) and click translate/Assemble or translate/Convert Base 16. A filename.obj will be created. Open Simulate.exe and load the filename.obj (no need to load the lc3os). Then choose Execute/Run and the output (if any) should show up in the console window.
- Apple (Best option is to use X11 or similar and remotely login to wormulon, use ssh –Y user1234@wormulon.cs.uidaho.edu)

On any system you can remotely access the LC-3 simulator and assembly/C converter already installed on the CS student server. To remotely access just login using ssh as outlined here. The LC-3 simulator and assembly/C converter is found in /usr/local/lc3. To add this to your path so the files can be called without putting in the full path:

For tcsh/csh:

Add this to your ~/.cshrc:

```
set path = ( $path /usr/local/lc3)
```

For bash:

Add this to your ~/.bash_profile:

```
PATH=$PATH:/usr/local/lc3

export PATH
```

Skills Test

Make sure you are comfortable using the simulator by performing the following tasks.

1. Download, assemble, and simulate warm1.asm.
2. Download, convert, and simulate warm2.hex.

To download these files, remember to right-click the links, then choose "**Save Link As...**" (or "Save Target As...") to save them. The output of the simulation should be saved and turned in as part of the lab report.

C Language

You will want to familiarize yourself with a C compiler for the system on which you will do your labs. You should be able to use the same compiler you used for the cs120 course. There are many tutorials and help available online (g++ is a compiler you can use and ./a.out will run the executable you create by default).

Skills Test

Make sure you are comfortable using your compiler by downloading, compiling, and executing warm3.c.

CSAC TA Passoff

Pass off with a CSAC TA by demonstrating the above skills:

- Create a CS computer account
- Demonstrate the text editor skills above
- Successfully run the LC-3 simulator in Linux
- Assemble and execute the sample LC-3 program warm.asm
- Convert and simulate warm.mco
- Compile and execute the C program warm.c

Make sure the CSAC TA sings you off on the provided list. If no list is available have the CSAC TA send an email to the grader (guo4949@vandals.uidaho.edu).

Lab Report

The lab report for this course should contain at least the following (and anything that will be helpful for you to remember for future labs):

1) Your name
2) Your email address
3) Your CS account login name
4) Which text editor you decided to use
5) From where you plan to run the LC-3 Simulator (home PC, laptop, lab)
6) The ssh command to remotely login and access the LC-3 simulator
7) The sftp command to remotely transfer files to your CS account
8) The output of the LC-3 simulator for warm.acm and warm.mco
9) The output of the compiled warm.c code
10) The name of the CSAC TA who passed you off