CS 451/551 Systems Programming Startup Directions

This handout documents what you need to do to set up your computer environments for this course.

Setting Up Your Account on the CS Linux Machines

All students registered for this course should have been given an account on `remote.cs.binghamton.edu` and should have been sent an email with their access information if the account was not already active.

It is required that all submitted programming projects work under the environment available on `remote`.

It is best that you set things up so that your account environment is setup automatically for the course. Unfortunately, the method of doing so depends on the kind of login shell you are using as different shells use slightly different syntaxes. Usually, you can determine which kind of shell you are using, based on the prompt you receive after you login: if the prompt contains a `$` character, you are probably using a `sh`-based shell; if the prompt contains a `%` character, you are probably using a `csh`-based shell. One way to be sure which shell you are using is by looking at the output of the command `ps -p $$`.

Use your favorite editor to make the below changes in your shell startup files (if you do not have a favorite editor, the end of this document contains instructions specific to `emacs`).

If using a `csh`-type shell, then add the following line to your login startup file (usually `~/.login`):

```
source ~umrigar/cs551-17s/bin/student.login
```

and the following line to the startup file run by your shell when started as a non-login shell (usually, `~/.tcshrc` or `~/.cshrc`):

```
source ~umrigar/cs551-17s/bin/student.cshrc
```

If you are using a `sh`-based shell, you need to add the following line to your login startup file (one of `~/.bash_profile`, `~/.bash_login`, or `~/.profile` if using `bash`)

```
. ~umrigar/cs551-17s/bin/student.profile
```

and the following line to the startup file run by your shell when started as a non-login shell (`~/.bashrc` if using `bash`):

```
. ~umrigar/cs551-17s/bin/student.rc
```

Note the leading period in both cases.

When you login **after** adding the above lines, you should see a **read-only** copy of the `cs551-17s` course directory in your home directory (this is merely a symbolic link to the master course directory). This directory contains all the course material and mirrors the course web site.

You are free to ignore these instructions if you know what you are doing. A typical situation when you may want to ignore these instructions is when you are using the same account for other courses and the environment setup by these instructions interferes with the environment needed for the other courses.

The Organization of the `cs551-17s` Subdirectory

Typically, the `cs551-17s` directory will contain the following subdirectories:

`bin`
: Shell scripts, etc.

docs
: Documentation.

exams
: Exams and solutions.

hws
: Homeworks and solutions.

`include`
: If present, it will contain header files for libraries specific to this course.

`interludes`
: If present, this directory will contain interesting material marginally relevant or totally irrelevant to this course.

`lib`
: If present, it will contain libraries specifically installed for this course.

misc
: Miscellaneous information (including files like this)

programs
: Programs discussed during class.

projects
: Programming projects and solutions.

slides
: Source, hardcopy and html versions of classroom transparencies.

`src`
: If present, this directory will contain source code for tools and libraries specifically installed for this course.

Most of the above directory structure is available via the course home page at http://zdu.binghamton.edu/cs551-17s. The site is also mirror'd at http://cs.binghamton.edu/~umrigar/cs551-17s/. Most of the subdirectories are currently

empty, but will have material added as the course progresses.

Tracking Changes

Assuming you have setup your account as outlined above, each time you subsequently login to your account, you should be given a list of *interesting* source files which have changed since your last login.

Note also that the course site is updated via a shared git repository which can be accessed using ssh at the URL:

```
ssh://user@remote.cs.binghamton.edu/~umrigar/cs551-17s.git
```

where *user* is your login-id on `remote`.

Specifically, assuming you have a ssh client and git on your local machine, you can clone the above repository using `git clone` *url dir*, where *url* is the above URL and *dir* is the directory into which you want to clone the repository.

You can then track changes within *dir* using the `git pull` command.

[Note: if you have set up your remote.cs account as outlined above, you will have already have a read-only copy of the `cs551-17s` directory in your home directory.]

To see what changes have been made, use the `git log -`*N* command in the `cs551-17s` directory to see the last *N* log messages; to see the last *N* log messages for *FILE* use `git log -`*N FILE*; the log messages will include a 40 character hex hash specifying the revision. To see what has changed between two specific revisions use `git diff -w -r` *HHHHH* `-r` *XXXXX FILE* where *HHHHH* and *XXXXX* are the first characters of the revisions you want to diff and *FILE* should specify the source file (usually `.umt`).

Local Development

We will be using open-source tools for this course and you are free to install them on your local machine. If your local

machine is sufficiently powerful with at least 3 GB of free disk space, 2 GB of free RAM and a recent x86-64 CPU which supports virtualization (VT-x/AMD-V), then you can run a environment on your local machine similar to that available on `remote` by running a vagrant-generated virtualbox VM on it.

To facilitate this, a Vagrantfile is provided which can be used to set up a debian 7.11 x86-64 virtual machine containing most of the tools needed for this course.

Specifically, you will need recent versions of both vagrant and virtualbox installed on your local machine. It is suggested that you install the latest version of vagrant from the vagrant site.

Copy the Vagrantfile into an empty directory and within that directory run the command `vagrant up`. This should download all necessary packages and create the virtual machine (it should take under 1 hour on a typical home internet connection).

You can login to your virtual machine using login-id `vagrant` with password `vagrant`. Use the right control key if you loose your mouse. Once logged in, you can start up a windowing environment using the command `startxfce4` (you are free to customize the machine and windowing environment as you wish). It is probably a good idea to immediately change the `vagrant` password. Note that the `vagrant` user has super-user access (using `sudo`); it may be a good idea to setup a unprivileged user for regular work.

You can run a full-screen X11 environment using the command `startxfce4`.

The use of vagrant is not supported; if you have problems, you are expected to debug them by yourself.

It is still your responsibility to ensure that all submitted programming projects work on remote.cs.

Instructions for Those Unfamiliar with a Unix Environment

If you do not already have a favorite editor for Unix systems, then `emacs` is strongly recommended, as it is much more than a editor and provides all the functions of a IDE. The instructions below refer to emacs.

In what follows, notation like `C-x` refers to holding down the control and `x` keys simultaneously, while `ESC x` refers to hitting the `ESC` key followed by the `x` key.

If emacs does not appear to be working properly, the most likely cause is that your terminal is configured incorrectly. Usually, you will be using a terminal emulator which will be emulating something like a `vt100` terminal. In that case, you would need to set the environmental variable `TERM` to `vt100`. Once again, the syntax depends on the shell family being used: use `TERM=vt100; export TERM` for `sh`-based shells and `setenv TERM vt100` for `csh`-based shells.

Remember that almost all emacs commands can be aborted using `C-g`.

Setting Up Your Account

Setting up your account typically involves adding a line to a couple of startup files. The following assumes that you are using `tcsh` as your shell and gives detailed instructions for adding a line to your `~/.login` file which is executed each time you login (vary the instructions according to your specific requirements; specifically, the added content will change if you are using a sh-based shell):

1. While in your home directory, type `emacs .login` to the Unix prompt. Emacs will be entered and will display the contents of the file.

2. Next use the cursor control down-arrow (or `C-n`) to move to the end of the file.

3. Insert the following line:

   ```
   source ~umrigar/cs551-17s/bin/student.login
   ```

   You must terminate this line using a newline --- i.e. you should get the cursor onto the next line.

4. Run the `save-buffer` command by using `C-x C-s`, and then exit emacs by using the `C-x C-c` (`save-buffers-kill-emacs`) command.

Next time you log in, a read-only directory called `cs551-17s` should appear in your home-directory. This directory will contain all the files you need for this course.

If you would like to learn emacs further, you can run the emacs tutorial by typing `C-h t` while inside emacs. You can read the online emacs manual by typing `C-h i`.