Configuring bash to your liking

- During your time in CS, you’ll be spending a lot of time working in/with bash, it helps to configure it to your liking
- There are a number of bash (and other) config files in your home directory, including .bashrc, .bash_aliases, and .history
- We’ll take a quick look at tweaking these (cautionary note: make a backup of each before you mangle it, just in case!)

Which file is which

- The .bashrc file is run whenever you login, so is a good place to put configuration options
- The .history file contains a numbered list of the commands you’ve run recently
- The .bash_aliases file contains your own aliases for specific commands
- The .bash_logout file is run whenever you logout, a good place for any “cleanup” commands you wish to automate

.bashrc

- .bashrc, contains default settings for many key environment variables (we’ll discuss some shortly)
- It also checks for a .bash_aliases file, and runs it if found
- If you create a script of bash functions that you want ‘sourced’, at the bottom of your .bashrc you can insert something like

```
If [ -f mybashstuff.sh ] ; then
    source mybashstuff.sh
fi
```

Environment variables

- Bash uses a number of established variables to keep track of various settings
  - PATH – which directories to search for programs
  - PS1 – what your command prompt looks like
  - HISTSIZE – how many history commands to store
  - EDITOR – which editor to use as your default
  - PAGER – which program to use to display files by default
  - CC – which compiler to use by default
  - ... there are many many many others you can set

Modifying environment variables

- They can be set like other bash variables, e.g. to change your default editor to vim use
- EDITOR=/usr/bin/vim;export EDITOR
- The export portion ensures your setting is available to all the child processes of your current shell
- If you simply type in “env” it will show all your current environment variable settings

Modifying PATH

- When you type in a command (e.g. ls), bash looks in a number of directories to find that program so it can be run
- The PATH variable determines which directories and in which order
- You can add directories to the path, you’ll see each directory listed as an absolute path and that they’re separated by colons (e.g. maybe you create your own bin directory full of useful programs you’ve written, so you add that directory to your path)
- WARNING COMING ON NEXT SLIDE .....

PATH risks

- Adding another user’s directory to your path can be risky, especially early in the path – if they put a buggy or nefarious program in that directory (and, for instance, name it ls) then you can potentially wind up running it the next time you use that command
- For that reason, if you add . (shorthand for the current directory) to your path ALWAYS ALWAYS ALWAYS make sure it is the very last thing in PATH

.bash_aliases

- This file is typically used to declare shorthand you wish to use for various commands
- Suppose you frequently cd to a deeply nested directory, e.g. davestu frequently cd’s to csci265/examples/bash so you wish to create a short name for that command
- The syntax is alias shorthand=’full command’, e.g.

```
alias bash265=’cd /home/student/davestu/csci265/examples/bash’
```

- Now to run the command davestu simply has to type *bash265*

Other config files

- Many programs store config files in your home directory, either in a file (e.g. .vimrc) or in a directory (e.g. .ssh)
- The settings and layout of those files depend very much on the specific program
- Customizing your files can make life much more pleasant, but remember to make backups first just in case you break something