CST8177 – Linux II

Processes
Todd Kelley
kelleyt@algonquincollege.com

Today's Topics

- testing scripts
- quoting
- from last lecture slides: crontab, anacron, at

Testing your scripts

- when you finish a script, you need to run it to verify correct operation
- you're expecting certain things from your script on certain runs
  - example: it expects arguments and you supply none – it should print an error message
  - example: you supply the wrong number of arguments – it should print an error message
- run your script with good input, and bad
  - check that operation is correct for good and bad
  - testing should "cover" all lines of code: every line of the script runs at least once during all your testing

Scripts and the checking program

- test your script before you run the assignment check program
- you need to be able to determine whether your script is behaving as you intended
- use -x and/or -v to "watch" it execute:
  - sh -x -u myscript.sh

Refresher on quoting

- http://teaching.idallen.com/cst8207/13w/notes/320_shell_variables.html
- http://teaching.idallen.com/cst8207/13w/notes/440_quotes.html

- You want variables to be inside double quotes, for two main reasons:
  1. globbing characters inside the variable will not be used to match filenames
  2. if the variable is empty, without double quotes it vanishes completely, and that's normally not what we want

Double Quoting Variables

- If a variable has a null value, as in

```
myvar=
# both of the following result in an error, because myvar is empty
if [ $myvar = something ] ; then echo yes; fi
# after variable expansion the above becomes
if [ = something ] ; then echo yes; fi
```

- If we put the same variable in double quotes:

```
myvar=
# both of the following do not result in an error (or any output)
if [ "$myvar" = something ] ; then echo yes ; fi
# after variable expansion the above becomes same as
if [ "" = something ] ; then echo yes ; fi
```