IT 240
Shell Scripting for Administrators

Chapter 13
Directory Operations

Stan J. Senesy
IT Program/CCS
New Jersey Institute of Technology

Globbing

- The natural behavior of the shell is to expand any filenames on the command line into matching filenames
- This works with the traditional shell wildcards - *, ?

Globbing

- If we pass *.txt as an argument to our script, the shell will expand it to all .txt files in the current directory and stores them in @ARGV
- This behavior may be used inside of a perl script as well with the glob operator

```
my @all_files = glob “*”;
```

Globbing

- Older programs may not use the glob operator as it's introduction is relatively recent
- An alternative syntax is:

```
my @all_files = <*>;
```

Directory Handles

- A directory handle acts as an alias to a file handle
- Instead of the typical *readline* operator, you use the *readdir* operator
- Once you're done with the directory handle, close it with the *closedir* operator

Removing Files

- The Unix command for file deletion is *rm*, but in perl we use the *unlink* operator

```
unlink “slate”, “bedrock”, “lava”;
```

- We can use the *glob* operator with unlink to delete a whole class of files
- The return value from *unlink* tells how many files have been successfully deleted

Renaming Files

- The *rename* function may be used to change the name of a file:

  rename “old_name”, “new_name:;

- It is important to check for the existence of a file before renaming as it will be overwritten if it exists

Making/Removing Directories

- The *mkdir* command is used to create a directory:

  makdir “fred”, 0755 or warn “Cannot make”;

- To delete a directory, use the *rmdir* command:

  rmdir “fred”;

- Remember, a directory must be empty before it may be deleted

More File Operators

- Changing file permissions:

  chmod 0755, “fred”;

- Changing ownership:

  chown “user”, “group”, “file”

- Changing timestamps:

  my $now = time;

  my $ago = $now - 24 * 60 * 60;

  utime $now, $ago, glob “*”;