Operating Systems :: Project Pages :: Project 4

Project 4-Clone Utility

Educational Objectives:

- Create copy functionality using UNIX API
- Create remove functionality using UNIX API
- Set file attributes using UNIX API

Operational Objectives:

- Use UNIX API to recursively copy directories
- Use UNIX API to recursively remove files
- Set permissions using UNIX API

Deliverables: Two files: clone.c and makefile

Assessment Rubric:

```
Your code uses good programming practices                                  [0....30]
Your program correctly copies all files/directories
from the source directory to the destination directory                     [0....40]
Your program duplicates the permissions, owner and group
from the source files/directories to the destination files/directories     [0....30]
Your program removes all files/directories in the destination
directory that are not in the source directory                             [-10..10]
                                                                           ---------
Score:                                                                     [0...100]
```

Details

Your task is to create a utility that will clone the contents of a given directory on to another directory. Your utility will be called as follows:

clone.x <source> <dest>

where <source> represents a source directory and <dest> represents a destination directory. You can assume the source and destination directories are actual directories (or don't exist). You can't assume, however, whether you will be given an absolute path or relative path. Your utility should follow this basic plan:

(1) If the source directory doesn't exist, print an error.

(2) If the destination directory doesn't exist, create it.

(3) Copy all files / directories in the source directory to the destination directory (recursively). Print a message for each file / directory copied.

(4) Set permissions, owner, and group for each destination file / directory to that of the source file / directory. Print a message for each file / directory attribute changed.

(5) For 10 points extra credit, remove any files / directories in the destination directory not in the source directory (recursively). Print a message for each file / directory removed. But, wait, there is a twist! If you delete files you shouldn't (such as other student's submissions or grading scripts), you will instead lose up to 10 points!

Let's go over each of these.

(1) and (2) are self-explanatory.

For (3), copying files will require the use of low-level reading and writing functions. Note that you

can't simply call the cp command. In fact, you aren't allowed to make use of any preexisting utilities. You may only use whatever functions the UNIX API provides. Directories may contain more files and directories. Hence, you will need to copy recursively. It is possible to do it iteratively, but you need to do it recursively for this assignment. Also, do not "copy the copy." That is, if the source directory contains the destination directory, you want to ignore the destination directory when searching the source directory. Also, you want to copy over files no matter if they already exist in the destination directory. This ensures that the destination directory has the same files as the source directory without having to implement extra logic to compare files.

For (4), you need to make sure the destination files and directories match permissions, owner, and group with the source files and directories.

Cloning

Copying files from one directory to another may seem like a trivial task when using existing utilities like cp. But, it takes a bit more work when simply using the API. So, let's come up with a plan for the basic outline in the project description.

Step 1

Your first step is to decide if the source directory exists. You know how to do that by using the chdir() function. This is an easy step. Note that you may be given either a relative or absolute path here, but it's best to work with absolute paths inside your program. getcwd() may be useful for obtaining the absolute path. realpath() is an easier option that others prefer. If you use this, be sure to place #define _GNU_SOURCE at the top of your program.

Step 2

Your second step is to decide if the destination directory exists. If not, create it. You can use a similar idea to above, but there is also an mkdir() function that will create the directory if it doesn't exist. Just what you need. Again, another easy step. Note that you may be given either a relative or absolute path here, but it's best to work with absolute paths inside your program. getcwd() may be useful for obtaining the absolute path. realpath() is an easier option that others prefer. If you use this, be sure to place #define _GNU_SOURCE at the top of your program.

Step 3

Your third step is the start of the real work. You have the source and destination directories, but how do you actually copy the files and directories over? Furthermore, how do you actually get the files and directories that exist in the source? There is a function that exists called readdir(). Read over it carefully. The great news is that there is a pretty useful example on that page under the documentation. The example shows how to read all the files and directories inside of a given directory. By read, we mean get the names of them. It is recommended that you start with this code as a base. Now, from the first two assignments, you know how to detect whether something is a file or directory using the stat() function. In fact, you'll want to use the stat() function a lot as it gives you most of the info you need about a file or directory to complete this assignment. But, why do you care about distinguishing between a file and a directory? Let's talk about files first. When you copy a file from one location to another, you are actually duplicating the file data. That is, you are reading from the source file and writing to the destination file. How to do that? The read() and write() functions of course. Study those pages. Directories, however, are a bit different. For every directory you encounter in the source directory , you simply want to "make" that directory in the destination directory (i.e. mkdir()).

All of this seems simple enough right? But, the recursive nature is something you now have to take in to account. Any time you encounter a directory, you want to recursively search that directory. In other words, you want to repeat the process above until you have no more directories. Recursion in C basically amounts to having a function call itself. It's likely your function that performs the copying task above will take in a source directory and a destination directory. So, when you "re"call it, you'll just be giving it a new source directory and a new destination directory.

Finally, be very careful that you ignore the . and .. filenames or you'll run into an infinite

loop. Also, as mentioned above, don't try to "copy the copy." For example, say the source directory is called mystuff and the backup directory is called mystuff/backup. If you don't ignore the mystuff/backup directory when reading from your mystuff directory, your program will start by creating the mystuff/backup directory if it doesn't exist. But, since your source directory now has a backup directory (and we're copying all directories over), the mystuff/backup directory will get a backup directory (i.e. mystuff/backup/backup). This will also cause an infinite loop.

Step 4

Now that you have all the files and directories copied over, it's time to match the permissions, owner, and group. Note that you could do this as you copy each file over. Again, the stat() function is handy here. Remember that the stat() function gives you a structure with information in it. You want file permissions, owner id, and group ip. Study this to see that everything is there for you. However, the permissions may take a bit of manipulation (no pun intended). But, the basic idea is to gather the information from the source file or directory and apply it to the destination file or directory. Functions chmod() and chown() will be of use.

Step 5

Extra credit. But, remember to be careful so you don't actually lose points!

Ok, this should get you going in the right direction. You'll notice that there is less specific code given this time. The reason is that there is less code to write for this project than the previous ones. Once you figure it out, it should be smooth sailing. You should investigate the API more and figure out how to use the various functions for this project.

Sample Runs

Here are some sample runs that you can use to compare your output to. Note that your output doesn't have to match exactly, but it couldn't hurt.

First, let's see the directory structure to backup:

```
xxxxxx@linprog3.cs.fsu.edu:~/stuff>ls -l * */*
-rw------- 1 xxxxxx CS-Grads 5895 Mar 24 2011 clone.c
-rw------- 1 xxxxxx CS-Grads 7512 Mar 24 18:21 clone.o
-rwx------ 1 xxxxxx CS-Grads 12655 Mar 24 18:21 clone.x
-rw------- 1 xxxxxx CS-Grads 310 Mar 24 15:43 makefile

music:
total 12
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 15:41 cd1
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 15:41 cd2
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 15:41 cd3

music/cd1:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 15:41 song.txt

music/cd2:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 15:41 song.txt

music/cd3:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 15:41 song.txt
```

Now, let's see what happens when trying to clone an invalid directory:

```
xxxxxx@linprog3.cs.fsu.edu:~/stuff>clone.x mystuff backup
mystuff is not a valid source directory.
```

Now, let's see what happens when we try to clone a directory on to itself:

```
xxxxxx@linprog3.cs.fsu.edu:~/stuff>clone.x . .
```

You see that nothing happens, which is good. Ok, let's do an actual clone:

```
xxxxxx@linprog3.cs.fsu.edu:~/stuff>clone.x . stuffbackup
Creating directory /home/grads/xxxxxx/stuff/stuffbackup
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup: 79038, 299
Copying /home/grads/xxxxxx/stuff/makefile to /home/grads/xxxxxx/stuff/stuffbackup/makefile
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/makefile: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/makefile: 79038, 299
Creating directory /home/grads/xxxxxx/stuff/stuffbackup/music
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music: 79038, 299
Creating directory /home/grads/xxxxxx/stuff/stuffbackup/music/cd1
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd1/song.txt to
/home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt: 79038, 299
Creating directory /home/grads/xxxxxx/stuff/stuffbackup/music/cd3
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd3/song.txt to
/home/grads/xxxxxxr/stuff/stuffbackup/music/cd3/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3/song.txt: 79038, 299
Creating directory /home/grads/xxxxxx/stuff/stuffbackup/music/cd2
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd2/song.txt to
/home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.c to /home/grads/xxxxxx/stuff/stuffbackup/clone.c
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.c: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.c: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.o to /home/grads/xxxxxx/stuff/stuffbackup/clone.o
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.o: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.o: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.x to /home/grads/xxxxxx/stuff/stuffbackup/clone.x
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.x: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.x: 79038, 299
```

Now, let's take a look at the cloned directory:

```
xxxxxx@linprog3.cs.fsu.edu:~/stuff>cd stuffbackup
xxxxxx@linprog3.cs.fsu.edu:~/stuff/stuffbackup>ls -l * */*
-rw------- 1 xxxxxx CS-Grads 5895 Mar 24 18:22 clone.c
-rw------- 1 xxxxxx CS-Grads 7512 Mar 24 18:22 clone.o
-rwx------ 1 xxxxxx CS-Grads 12655 Mar 24 18:22 clone.x
-rw------- 1 xxxxxx CS-Grads 310 Mar 24 18:22 makefile

music:
total 12
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 18:22 cd1
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 18:22 cd2
drwx------ 2 xxxxxx CS-Grads 4096 Mar 24 18:22 cd3

music/cd1:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 18:22 song.txt
```

```
music/cd2:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 18:22 song.txt

music/cd3:
total 4
-rw------- 1 xxxxxx CS-Grads 29 Mar 24 18:22 song.txt
```

You see that it matches the original. You also see that there is no "copy of a copy" issue.

Now, let's see what happens if there is something in the destination directory not in the source directory:

```
xxxxxx@linprog2.cs.fsu.edu:~/stuff>clone.x . stuffbackup
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup: 79038, 299
Copying /home/grads/xxxxxx/stuff/makefile to /home/grads/xxxxxx/stuff/stuffbackup/makefile
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/makefile: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/makefile: 79038, 299
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music: 79038, 299
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd1/song.txt to
/home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd1/song.txt: 79038, 299
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd3/song.txt to
/home/grads/xxxxxx/stuff/stuffbackup/music/cd3/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd3/song.txt: 79038, 299
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2: 79038, 299
Copying /home/grads/xxxxxx/stuff/music/cd2/song.txt to
/home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/music/cd2/song.txt: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.c to /home/grads/xxxxxx/stuff/stuffbackup/clone.c
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.c: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.c: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.o to /home/grads/xxxxxx/stuff/stuffbackup/clone.o
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.o: 600
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.o: 79038, 299
Copying /home/grads/xxxxxx/stuff/clone.x to /home/grads/xxxxxx/stuff/stuffbackup/clone.x
Setting permissions for /home/grads/xxxxxx/stuff/stuffbackup/clone.x: 700
Setting user and group for /home/grads/xxxxxx/stuff/stuffbackup/clone.x: 79038, 299
Removing /home/grads/xxxxxx/stuff/stuffbackup/video/video1.txt
Removing /home/grads/xxxxxx/stuff/stuffbackup/video/video2.txt
Removing /home/grads/xxxxxx/stuff/stuffbackup/video/video3.txt
Removing /home/grads/xxxxxx/stuff/stuffbackup/video
```

Grading

Your code uses good programming practices (30%).

Your program correctly copies all files / directories from the source directory to the destination directory (40%).

Your program duplicates the permissions, owner, and group from the source files / directories to the destination files / directories (30%).

Your program removes all files / directories in the destination directory that are not in source directory (+ / - 10%).

Submitting

Submit two files: clone.c containing your source code, and makefile which builds an executable named clone.x. Do this with ~cop4610p/submitscripts/proj4submit.sh.