Exercises-Fileutils2

Part One

Description

In this exercise set you will practice the **mv** command. **mv** is used to rename and rearrange files and directories. Begin by creating a directory to work in beneath your home directory on *hills*. (This is ***your work area*** for this exercise). Here are the forms of the **mv** command:

Synopsis	***Meaning***
mv file1 file2	**file2** is deleted if it exists. Then **file1** is renamed **file2**
mv a [b c ...] dir **a,b,c,...** can be files or directories	move one or more existing files or directories into an existing directory. If a directory is moved, everything beneath it goes with it,.
mv dir1 dir2	if **dir2** does not exist, rename **dir1** to **dir2**. If **dir2** exists, this is the previous form, and **dir1** is moved into **dir2**
other options	**-i** - *interactive copy*. Asks for verification for each individual copy operation before proceeding **-v**: *verbose.* Reports each individual copy operation as it is done.

Procedure

In this part you create a directory structure by taking parts from an existing directory. Begin by getting a copy of the existing directory structure by copying (recursively) the directory located at **fileutils/test** beneath the class public data area on *hills* to *your work area*. You should begin by examining the **test** structure and perhaps making a drawing of it. Then make a new directory **test1** in *your work area* and connect to it. (Note: **test1** and your copy of **test** are assumed to be at the same level (i.e., they are both in the same directory)

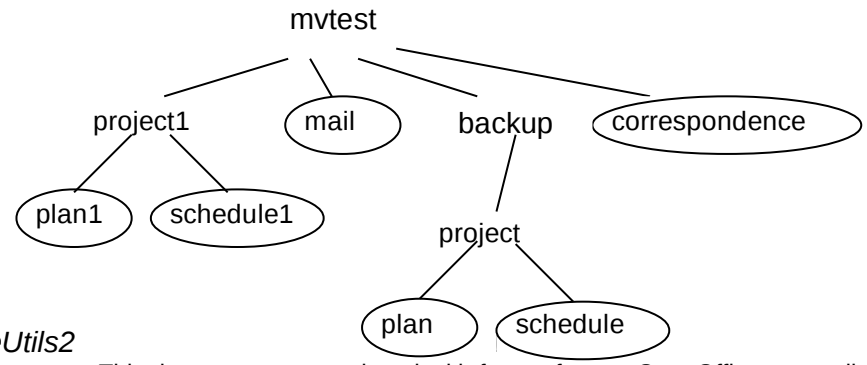
1. Place the files **sally** and **john** in **test1** (use the **mv** command)
2. Place the file **andrea** in **test1** and name it **sharon**
3. Place the **test/proj** directory in **test1**
4. Place the directory **current** in **test1**, naming it **recent**
5. Rename the **test1/proj** directory to **test1/project**
6. Place what is left of the **test/mail** directory beneath **test1/recen**t

List your **test1** directory recursively and see if it makes sense.

Part Two

In this part you will take a directory and rearrange it. Begin by recursively copying (use **cp**; ***do not use the* mv *command***) **fileutils/mvtest** from the class public data area on hills to *your work area*. List the directory structure and make a map of it.

Then rearrange it to the following structure using the minimum number of commands possible. During rearrangement, you are only allowed to use the **mv** and **rmdir** commands.

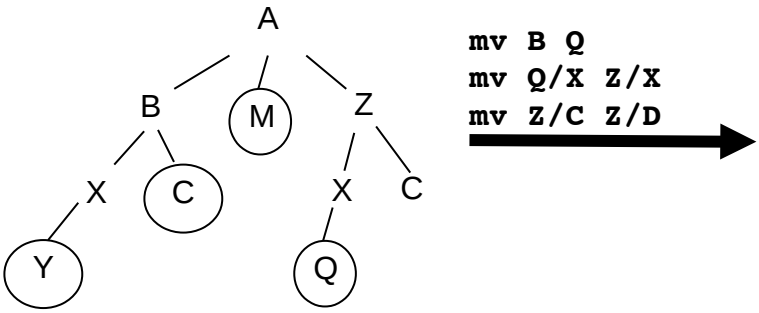


Part Three

In this exercise you will practice using mv on directory trees. In each exercise, apply the commands *in order* to the tree on the left and draw the resultant tree on the right. In all trees, you are connected to the top directory (which is A or B). To investigate your answers, you can copy the original trees from **`fileutils/part3`** (copy that directory recursively to your home directory. In it you will find **`A`** and **`B`**).

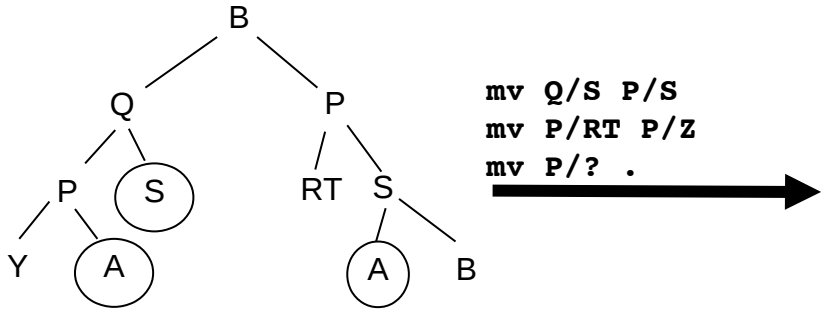
A last tree practices a combination of cp and mv on the same tree.

1.



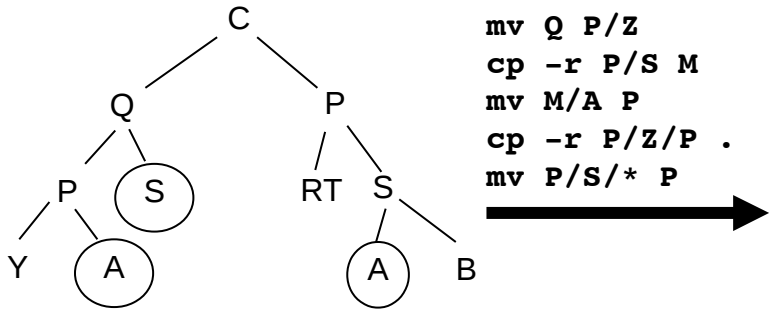

```
mv B Q
mv Q/X Z/X
mv Z/C Z/D
```

2.




```
mv Q/S P/S
mv P/RT P/Z
mv P/? .
```

3.




```
mv Q P/Z
cp -r P/S M
mv M/A P
cp -r P/Z/P .
mv P/S/* P
```

Answers

Part One

Begin with the following commands to create your empty **test1** and your copy of **test** to work with:

```
cp -r /pub/cs/gboyd/cs160a/fileutils/test .
mkdir test1
cd test1
```

1. **mv ../test/sally ../test/john . OR mv ../test/{sally,john} .**
2. **mv ../test/andrea sharon**
3. **mv ../test/proj .**
4. **mv ../test/mail/current recent**
5. **mv proj project**
6. **mv ../test/mail recent**

Your **test1** directory should look like this after you are finished:

```
$ ls -RF test1
john      project/  recent/   sally     sharon

test1/project:
progress   work

test1/recent:
in     mail/  out

test1/recent/mail:
proj/

test1/recent/mail/proj:
old
```

Part Two

Begin by making a copy of the mvtest directory and connecting to it:

```
cp -r /pub/cs/gboyd/cs160a/fileutils/mvtest .
cd mvtest
```

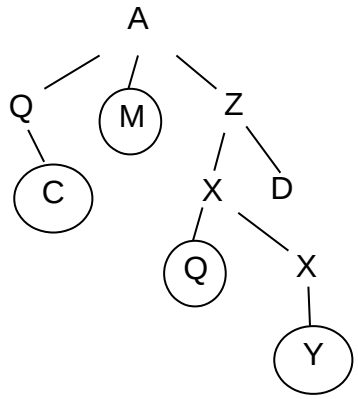
Then issue the following commands to rearrange the tree:

```
mv backup/* .
mv projects/project backup
mv projects/project1 .
rmdir projects
```

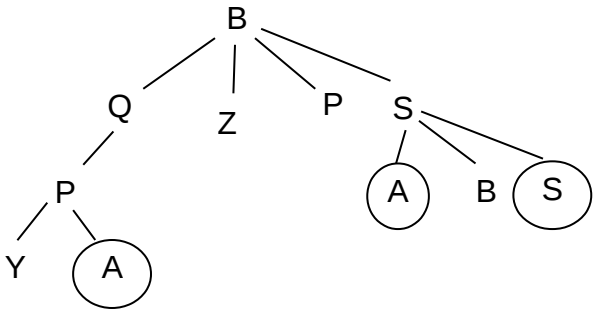
Part Three

(next page)

1.



2,



3. P/S/A overwrites P/A