Unix Shell Moderate Practice Problems

1. File and Directory Management 1

You are working on a project and need to create a directory named `data` inside your project directory (`project3`) and a file named `info.txt` inside it.

This is your current terminal session:

```
user@computer:~/project3/scripts$ pwd
    /home/user/project3/scripts

user@computer:~/project3/scripts$
```

Select and order the following commands to accomplish this task.

Select from the list below by dragging

- `echo info.txt`
- `cd scripts`
- `ls`

Drop code here in the correct order

- `cd ..`
- `mkdir data`
- `cd data`
- `touch info.txt`

HIDE SOLUTIONS

SOLUTION:

```
user@computer:~/project3/scripts$ cd ..
user@computer:~/project3$ mkdir data
user@computer:~/project3$ cd data
user@computer:~/project3/data$ touch info.txt
```

We start off in the `scripts` directory.

- First, we need to navigate to the project root directory using `cd ..` - (`project3` is the parent directory of `scripts`)
- Then, we can create the `data` directory and navigate into it.
- Finally, we create the `info.txt` file.
- we can also list the contents of the current directory using `ls` to check if the `data` directory and `info.txt` file was created successfully

2. File and Directory Management 2

Create a directory structure for a project with the following hierarchy:

```
project/
├── src/
│   ├── main.py
│   └── utils.py
├── data/
│   ├── input.csv
│   └── output.csv
└── README.md
```

```
cd project
mkdir src data; touch README.md
cd src; touch main.py utils.py
cd ../data; touch input.csv output.csv
```

HIDE SOLUTIONS

SOLUTION:
To create the directory structure, you can use the following commands.

```
mkdir -p project/src project/data
touch project/src/main.py project/src/utils.py project/data/input.csv project/data/output.csv project/README.md
```

- Since the problem asks us to create this in the current directory, we don't need to `cd` anywhere.
- `mkdir -p` creates the directories and any necessary parent directories.
- `touch` creates the files in the specified directories.

Alternative solution:

```
mkdir project
mkdir project/src
touch project/src/main.py
touch project/src/utils.py
mkdir project/data
touch project/data/input.csv
touch project/data/output.csv
touch project/README.md
```

- In this longer example, we create the `project` directory first, then create the `src` and `data` directories inside it.
- `touch` is called individually for each file

Additionally:

- `cd` can also be used to change into each directory before creating files, but it is not necessary

3. Output Redirection

Which command correctly redirects the standard output of `ls` to a file named `filelist.txt` without overwriting?

- (●) ls >> filelist.txt
- ( ) ls > filelist.txt
- ( ) ls < filelist.txt
- ( ) ls filelist.txt

HIDE SOLUTIONS

SOLUTION:
The correct command is:

```
ls >> filelist.txt
```

This sends the output of `ls` into `filelist.txt`, without overwriting the file if it exists.

- `ls > filelist.txt`: This command **overwrites** the contents of `filelist.txt` with the output of `ls`.
- `ls < filelist.txt`: This command tries to read from `filelist.txt` as input, which is not what we want. (*not covered in the main session*)
- `ls filelist.txt`: This command tries to list the contents of `filelist.txt`, instead of writing to it.

---

4. Debugging 1

You are trying to run a script but encounter an error. The script is supposed to print the current date and time. However, it fails with the following error:

```
bash: ./script.sh: No such file or directory
```

What is the likely cause of this error?

- ( ) The script does not have execute permissions.
- ( ) There is a syntax error in the script.
- (●) ~~The script is not in your PATH.~~
- ( ) The script does not exist in the current directory.

HIDE SOLUTIONS

SOLUTION:
The script does not exist in the current directory.

- The script does not have execute permissions: If this were the case, the error would be `permission denied`, not `No such file or directory`.
- The script is not in your PATH: The error message indicates that the script is being called with `./`, which means it is expected to be in the current directory, not in the PATH.
- There is a syntax error in the script: A syntax error would occur after the script is found and executed, not before.

5. Debugging 2

Suppose this is the contents of a directory on your computer:

```
user@computer:~$ ls ~
    myfile.txt    mydata.csv    myscript.sh   myfolders/
```

When you type the following commands, you get an error:

```
user@computer:~$ mv mydata.csv myfolder/
    mv: cannot move 'mydata.csv' to 'myfolder/': No such file or directory
user@computer:~$ mv myfile.txt myfolder/
    mv: cannot move 'myfile.txt' to 'myfolder/': No such file or directory
user@computer:~$ mv myscript.sh myfolder/
    mv: cannot move 'myscript.sh' to 'myfolder/': No such file or directory
```

How can you fix this error?

```
mv mydata.csv myfolders/
mv myfile.txt myfolders/
mv myscript.sh myfolders/
```

HIDE SOLUTIONS

SOLUTION:

```
user@computer:~$ mv mydata.csv myfolders/
user@computer:~$ mv myfile.txt myfolders/
user@computer:~$ mv myscript.sh myfolders/
```

The error indicates that the directory `myfolder/` does not exist. Ordinarily, we could create the directory using the `mkdir` command, but in this case, it looks like the directory is already created as `myfolders/`. To fix the error, we can either create the directory `myfolder/` or move the files to `myfolders/` instead.

Note the spelling of myfolder**s**/

6. Organizing files

Suppose this is the contents of a folder on your computer:

```
user@computer:~$ ls /data/MYPROJECT/scripts
    analyze_data.py     dataset1.db        dataset2.db      outputimg.png
    outputimg2.png
```

You've just opened a new terminal and want to organize the files by moving datasets and outputs in separate folders. Select and order the following commands to accomplish this task.

Select from the list below by dragging	Drop code here in the correct order

```
movefile output* outputs/
cd outputs
movefile dataset* datasets/
cd data
echo scripts/output*
cd
ls
```

```
cd /data/MYPROJECT
mkdir datasets
mv dataset* datasets/
mv scripts/output* outputs/
mkdir outputs
mv output* outputs/
mv scripts/dataset* datasets/
```

1
2
3
2
3

HIDE SOLUTIONS

SOLUTION:

```
user@computer:~$ cd /data/MYPROJECT
user@computer:/data/MYPROJECT$ mkdir datasets
user@computer:/data/MYPROJECT$ mkdir outputs
user@computer:/data/MYPROJECT$ mv scripts/dataset* datasets/
user@computer:/data/MYPROJECT$ mv scripts/output* outputs/
```

First, we need to navigate to the project directory, then create the `datasets` and `outputs` directories. After that, we can move the dataset and output files into their respective directories.

- Running `mv dataset* datasets/` won't work, because the `dataset*` files are in the `scripts` directory, not the current directory.
- The command `ls` can be run to check the contents of the current directory, but it is not necessary to accomplish the task.

7. Text files

Suppose you have the following script:

```
# preview all my data files
echo "Data file preview"
echo "================="
cat mydata/*.txt
```

When your project had fewer data files, this script worked fine. But now, the output is too long to read.
What are some approaches to fix this?

```
cat mydata/*.txt | head --n 5
```

HIDE SOLUTIONS

SOLUTION:

There are several ways to fix this issue. Here are a few options:

- Redirect the output to a file for later review with less

```
cat mydata/*.txt > preview.txt
less preview.txt
```

More advanced strategies not covered in the main session:

- Use the pipe operator to pipe the output to `less` directly.

```
cat mydata/*.txt | less
```

- Use `head` or `tail` to preview only the first or last few lines of each file.

```
head mydata/*.txt
tail mydata/*.txt
```