Shell Basics Practice Problems

1. Creating Directories

Which command would you use to create a directory named `dir1`?

- ○ `ls dir1`
- ◉ `cd dir1`
- ○ `touch dir1`
- ○ `mkdir dir1`

HIDE SOLUTIONS

SOLUTION:
The correct command to create a directory named `dir1` is:

```
mkdir dir1
```

- `touch dir1` would create a file named `dir1`, not a directory.
- `cd dir1` would change into a directory named `dir1`, but it doesn't create it.
- `ls dir1` would list the contents of a directory named `dir1`, but it doesn't create it.

2. Creating Directories

How would you create directories named `dir2` and `dir3`, within a new directory named `dir1`?

- ○
```
cd dir1
mkdir dir2 dir3
```
- ○ `mkdir dir1/dir2 dir1/dir3`
- ○
```
mkdir dir1
cd dir2
mkdir dir3
```
- ○
```
mkdir dir1
mkdir dir2 dir3
```
- ◉
```
mkdir dir1
mkdir dir1/dir2 dir1/dir3
```

HIDE SOLUTIONS

SOLUTION:
First, we must create `dir1`, and then we can create `dir2` and `dir3` inside it.

Alternatively, we can use the `mkdir -p` command to create the parent directory and subdirectories in one go:

```
mkdir -p dir1/dir2 dir1/dir3
```

This command would creates `dir1` and its subdirectories `dir2` and `dir3` in a single step.

The other options are incorrect.

```
mkdir dir1
mkdir dir2 dir3
```

- This would create `dir2` and `dir3` in the current directory, not inside `dir1`.

```
cd dir1
mkdir dir2 dir3
```

- This would fail because `dir1` does not exist yet.

```
mkdir dir1
cd dir2
mkdir dir3
```

- This would fail because `dir2` does not exist yet.

3. Verifying Directory Creation

You've just created the directories above.

For reference, this is your current terminal session:

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

How might you verify the directories were created successfully?

- ○ `ls`
- ◉ `ls -r`
- ○ `cat dir1/*`
- ○ `ls dir1`
- ○ `ls dir2 dir3`
- ○ `find dir1 dir2 dir3`
- ○ `less dir1 dir1/dir2 dir1/dir3`

HIDE SOLUTIONS

SOLUTION:
To verify the directories were created successfully, you can use: `ls dir1` or `ls -r`

- `ls dir1` lists the contents of `dir1`, which should include `dir2` and `dir3`.

- `ls -r` lists all files and directories in the current directory and its subdirectories.

The other options are incorrect because:

- `ls` would list the contents of the current directory, but not specifically check for `dir2`, or `dir3` which are inside `dir1`.
- `ls dir2 dir3` would fail because `dir2` and `dir3` are inside `dir1` and do not exist in the current directory.
- `find dir1 dir2 dir3` is not a valid command.
- `cat dir1/*` would attempt to display the contents of files in `dir1`, but there are no files yet.
- `less dir1 dir1/dir2 dir1/dir3` would attempt to display the contents of `dir1`, `dir1/dir2`, and `dir1/dir3`, but these are directories, not files.

4. Creating Text Files

After verifying you've created the directories, how would you create two text files named `file1.txt` and `file2.txt` inside a folder named `text_files`?

This is your current terminal session:

```
user@computer:~$ pwd
    /home/user
user@computer:~$ ls
    dir1
user@computer:~$
```

Select from the list below by dragging

- `echo file1.txt file2.txt`
- `new file1.txt file2.txt`
- `less file1.txt file2.txt`
- `cp file1.txt file2.txt`
- `mv file1.txt file2.txt`
- `cat file1.txt file2.txt`
- `touch file1 file2`

Drop code here in the correct order

- `mkdir text_files`
- `cd text_files`
- `touch file1.txt file2.txt`

HIDE SOLUTIONS

SOLUTION:
To create the text files, you can follow these steps:

```
mkdir text_files
cd text_files
touch file1.txt file2.txt
```

This creates a new directory named `text_files`, navigates into it, and creates two empty text files named `file1.txt` and `file2.txt`.

The other options are incorrect because:

- `touch file1 file2` would create files without the `.txt` extension.

- `new file1.txt file2.txt` is not a valid command.
- `echo file1.txt file2.txt` would just print `file1.txt file2.txt` to the terminal, not create files.
- `mv file1.txt file2.txt` would move files, but they don't exist yet.
- `cp file1.txt file2.txt` would copy files, but they don't exist yet.
- `less file1.txt file2.txt` would attempt to display the contents of non-existent files.
- `cat file1.txt file2.txt` would attempt to display the contents of non-existent files.

5. Appending Text to a File

Suppose your terminal session looks like this:

```
user@computer:~$ pwd
    /home/user
user@computer:~$ ls text_files
    file1.txt  file2.txt
user@computer:~$
```

You want to *append* the text "Bash is hard!" to `file1.txt`.

Select from the list below by dragging

- `cat file1.txt`
- `mv file1.txt file2.txt`
- `less file1.txt`
- `rm file1.txt`
- `cp file1.txt file2.txt`
- `echo 'Bash is hard!' > file1.txt`

Drop code here in the correct order

- `cd text_files`
- `echo 'Bash is hard!' >> file1.txt`

HIDE SOLUTIONS

SOLUTION:
To append text to `file1.txt`, you can use:

```
cd text_files
echo 'Bash is hard!' >> file1.txt
```

This command navigates to the `text_files` directory and appends the text "Bash is hard!" to `file1.txt`.

The other options are incorrect because:

- `echo 'Bash is hard!' > file1.txt` would overwrite the contents of `file1.txt`, not append to it.
- `cat file1.txt` and `less file1.txt` would display the contents of the file but not modify it.
- `mv file1.txt file2.txt` would rename or move the file, not append text.
- `cp file1.txt file2.txt` would copy the file, not append text.
- `rm file1.txt` would delete the file.

6. Verifying Appended Text

Suppose your terminal session looks like this:

```
user@computer:~/text_files$
```

After appending text to a file, how would you verify that your text has been successfully added to `file1.txt`?

Select from the list below by dragging

- `rm file1.txt`
- `echo file1.txt`
- `cd text_files`
- `ls file1.txt`

Drop code here in the correct order

- `cat file1.txt`

HIDE SOLUTIONS

SOLUTION:
To verify that the text has been successfully added to `file1.txt`, you can use:

```
cat file1.txt
```

This command displays the contents of `file1.txt`, allowing you to confirm that the text "Bash is hard!" is present.

The other options are incorrect because:

- `cd text_files` is not necessary since you are already in the `text_files` directory.
- `echo file1.txt` would just print the filename, not its contents.
- `rm file1.txt` would delete the file.
- `ls file1.txt` would list the file, but not its contents.

7. Deleting Files and Globs

Suppose you are inside `dir2`. This is your current terminal prompt:

```
user@computer:~$ ls dir1
    file1.txt   file2.txt   dir2/   dir3/
user@computer:~$ cd dir1/dir2
    /home/user/dir1/dir2
user@computer:~/dir1/dir2$
```

How would you delete all `.txt` files within `dir1`?

- (●) `rm dir1/*.txt`
- ( ) `rm .txt`
- ( ) `rm ../*.txt`
- ( ) `rm dir1/dir2/.txt`

HIDE SOLUTIONS

SOLUTION:
To delete all `.txt` files within `dir1`, you can use:

```
rm ../*.txt
```

This command removes all `.txt` files in the parent directory (`dir1`) from your current location inside `dir2`.

The other options are incorrect because:

- `rm .txt` would not match any files.
- `rm dir1/*.txt` and `rm dir1/dir2/.txt` would fail because you are currently in `dir2`

---

8. File and Directory Management: Navigating to the Parent Directory

This is your current terminal session:

```
user@computer:~/dir1/dir2$ pwd
    /home/user/dir1/dir2
user@computer:~/dir1/dir2$ ls
user@computer:~/dir1/dir2$
```

After working inside one of the directories, how would you navigate back to your home directory?

- ○ `cd /home/user`
- ○ `cd dir1`
- ◉ `cd` ✓
- ○ `cd ../..`
- ○ `cd ..`

HIDE SOLUTIONS

SOLUTION:
To navigate back to your home directory, you can use:

- `cd` - This command takes you directly to your home directory.
- `cd /home/user` - This command takes you to the absolute path /home/user, which is your home directory.
- `cd ../..` - This command takes you two levels up from your current directory, which would also take you to your home directory.

These other options are incorrect because:

- The command `cd ..` would take you one level up to `dir1`, not directly to your home directory.
- The command `cd dir1` would attempt to change into the `dir1` directory which will fail because there is no `dir1` in the current directory.

9. Removing Directories and Their Contents

This is your current terminal session:

```
user@computer:~$ ls
    Desktop/  dir1/  Documents/  Downloads/  Music/  Pictures/  Videos/
user@computer:~$
```

How would you remove directory `dir1` and all its contents, including `dir2`, `dir3`, and their files? Then confirm that all directories and files have been deleted.

- ○ `ls dir1/*`
- ○ `rm -r dir1 dir2 dir3`
- ◉ `rm -r dir1` ✓
- ○ `rm dir1`
- ○ `ls`
- ○ `ls -r dir1`

HIDE SOLUTIONS

SOLUTION:
To remove `dir1` and all its contents, including `dir2`, `dir3`, and their files, you can use:

```
rm -r dir1
```

This command recursively removes the directory and all its contents.

The other options are incorrect because:

- `rm dir1` would fail because `dir1` is a directory and requires the `-r` option to remove it.
- `rm -r dir1 dir2 dir3` would attempt to remove `dir1`, `dir2`, and `dir3` but would fail because they are not in the current directory.
- `ls` would list the contents of the current directory, but not confirm that `dir1` has been deleted.
- `ls -r dir1` would list the contents of `dir1`, but it doesn't confirm that it has been deleted.
- `ls dir1/*` would list the contents of `dir1`, but it doesn't confirm that it has been deleted.