Bash scripting exercises

some cool problem solving exercises with bash :

Weather Forecast

The *weather.sh* utility in the current working directory provides weather forecasts in your area.

When called, it outputs a 4-day forecast in the format "day: weather" separated by line breaks.

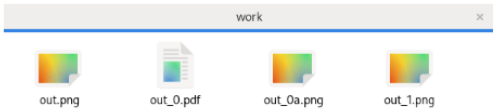
Write a bash script that outputs the number of days with a "Thunderstorm" forecast (case-sensitive).

Image Files

You ran a script that wrongly put its output image files in the 'work' directory inside the current directory.

Create a bash script that removes all the files in the 'work' directory that start with either 'out_0' or 'out_1' and end with '.png'.

For example, if the 'work' directory contains the following files:



After running your script, both *out.png* and *out_0.pdf* should exist in the 'work' directory but *out_0a.png* and *out_1.png* should no longer exist.

Elif

Select the statements about the output of the following script that are correct.

```
#!/bin/bash
value=$1
if [ $value -eq 100 ]
then
    echo "value is eq 100"
elif [ $value -gt 100 ]
then
    echo "value is gt 100"
else
    case $value in
    [1-3]*) echo "value is between 1 and 39 (inclusive)" ;;
    [4-5]*) echo "value is between 40 and 59 (inclusive)" ;;
    9[1-8]) echo "value is between 91 and 98 (inclusive)" ;;
    99) echo "value is $value" ;;
    *) echo "value is unknown" ;;
    esac
fi
```

ANSWER:

(Select all acceptable answers.)

- [ ] If the first argument of the script is 35 then 'value is unknown' echoed.
- [x] If the first argument of the script is 90 then 'value is unknown' echoed.
- [x] If the first argument of the script is 91 then 'value is between 91 and 98 (inclusive)' echoed.
- [ ] If the first argument of the script is 99 then 'value is \$value' echoed.
- [x] If the first argument of the script is 100 then 'value is eq 100' echoed.

Submit

Your score is 100%, perfect!

Environment

The PATH variable is set in /etc/environment:

```
PATH="/usr/local/sbin:/usr/local/bin:/usr/games:/usr/local/games"
```

And updated in .bash_profile:

```
export PATH=/usr/temp:$PATH:/usr/local/apps
```

Select the statements are correct.

(Select all acceptable answers.)

- [x] */usr/local/bin* will be searched before */usr/games*.
- [ ] */usr/local/apps* will be searched before */usr/local/sbin/*.
- [ ] If both */usr/local/apps* and */usr/local/bin* contain a binary with the same name, the version in */usr/local/apps* will be executed.
- [ ] If the current directory contains a binary with the same name as one in */usr/local/sbin* the version in the current directory will be executed.

ANSWER:

(Select all acceptable answers.)

- [x] */usr/local/bin* will be searched before */usr/games*.
- [ ] */usr/local/apps* will be searched before */usr/local/sbin/*.
- [ ] If both */usr/local/apps* and */usr/local/bin* contain a binary with the same name, the version in */usr/local/apps* will be executed.
- [ ] If the current directory contains a binary with the same name as one in */usr/local/sbin* the version in the current directory will be executed.
- [x] The *which* command can be used to find binaries at locations within the PATH.

Submit

Your score is 100%, perfect!

Cron

The output of *crontab -l* contains the following job definitions:

```
MAILTO=candidate@testdome.org,admin@testdome.org
0 0,12 1 */4 * df -h /

@hourly du -h / >> /var/log/disk_usage

0 6 1-12 * * /usr/bin/testdome_candidates
```

Select the statements about the job definitions and schedules that are correct.

(Select all acceptable answers.)

- [x] The df command is executed at midday and midnight on the first day of every 4th month.
- [ ] The df command is executed 12 times, each hour, between midday and midnight on the 1st of April.
- [x] The MAILTO variable tells cron to email the output of df -h / to candidate@testdome.org and admin@testdome.org.
- [x] Every hour of every day the current disk usage is appended to the /var/log/disk_usage log file.
- [ ] The /usr/bin/testdome_candidates binary is executed at 6:00am every day.

Submit

Your score is 100%, perfect!

Allow Login

You are the sysadmin of a Linux server that uses local passwords for authentication.

Two members of the team were assigned to work on the server: Ashley, who needs to change her username, and Peter, who already has a user account but forgot his password.

What steps should you do so Ashley and Peter are able to login to the server using their new credentials?

ANSWER :

(Select all acceptable answers.)

- [x] Get superuser access by logging in as *root* or using a command such as *sudo* to perform the changes.
- [x] Change Ashley's username with the *usermod* command.
- [ ] Manually create a new */home* folder for Ashley.
- [x] Use the *passwd* command to change Peter's password.
- [ ] Delete the old */etc/passwd* file so it is automatically regenerated with the default user passwords.
- [ ] Restart the system to enable the new password for Peter.

Submit

Your score is 100%, perfect!

Debug Programs Directory

Your company has a Linux computer with a directory named *debug_programs* intended for developers only.

The following needs to be done:

- *debug_programs* should only be accessible to users in the *developers* group.
- Alex should be removed from the *developers* group because he left the company.
- Confirm that the new developers Cynthia, Emma, and Justin belong to the *developers* group.

What should be done to satisfy the requirements above?

Answer :

(Select all acceptable answers.)

- [x] Use the *chown* and *chmod* utilities to change the permissions of the *debug_programs* directory and its contents.
- [x] Use the *usermod* utility to remove Alex from the group.
- [ ] Add a file named 'developers' with the developer usernames, and place it in */etc/group*.
- [ ] Add Alex to the *nobody* group.
- [ ] Check that Cynthia, Emma, and Justin are in the *developers* group with the *users* command.
- [x] Use *groups username* for each developer to confirm whether they belong to the *developers* group.

Submit

Your score is 100%, perfect!