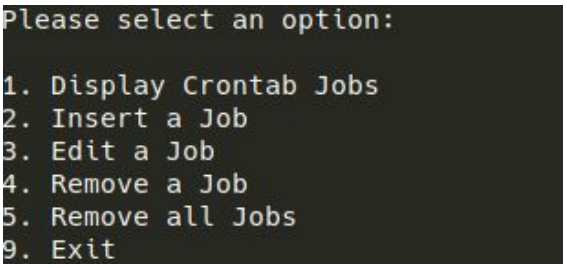
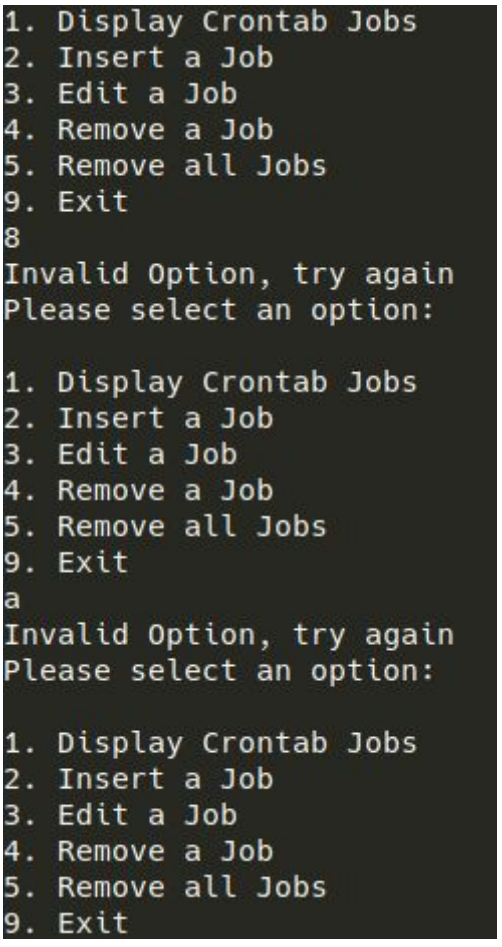
Operating Systems Crontab Coursework

This is the documentation for the operating systems coursework, the members are Harry Addlesee, Kian Fullarton and Kevin Kimuyu. To begin the script the user is required to run the ‘./mycrontab.sh’ command and then the script begins by displaying all of the options that the user can select from and then reads the user's keypress to move to the requested part of the code.



The options that the user can select from are shown in the screenshot above. If the user attempts to enter a number that is not one of the options, or a random character, the script will print an error message and then reprint the options that can be selected. The screenshot below shows what happens if you try and enter an invalid option.



Displaying All Jobs

Option 1 is for displaying all of the user's current crontab jobs. This is done by simply running the 'crontab -l' command. If option 1 is selected when the user has no current crontab jobs then a message is displayed informing them that they have no current jobs. In the screenshot below, option 1 is selected and then the options are reprinted after displaying the current crontab jobs.

```
1
Option 1 selected
31 * 21 * 1 /home/harry/gitRepos/Coursewok/job-scripts/test3.sh
8 12 19 5 2 /home/harry/gitRepos/Coursewok/job-scripts/test2.sh
* 20 21 11 * /home/harry/gitRepos/Coursewok/job-scripts/test1.sh

Please select an option:

1. Display Crontab Jobs
2. Insert a Job
3. Edit a Job
4. Remove a Job
5. Remove all Jobs
9. Exit
```

Inserting a Job

Option 2 is for inserting a new crontab job. The user is asked a few questions relating to when they would like their new crontab job to run. For each of these questions the user is given a range of numbers that can be entered or they can choose to use an asterisk (*) to have the crontab job run at every instance, for example the crontab job will run every hour of a day rather than just at one hour of the day.

```
echo x"$DofW" | grep '*' > /dev/null;
```

The screenshot above is the line of code that is used to to check if an asterisk (*) is entered. The '|' sends the output of 'echo' to grep which performs the check and then the output of grep is then sent to 'dev/null' to avoid confusing the user with extra lines being outputted. If an asterisk (*) is found then the number check will be bypassed but if no asterisk (*) is found then the script will continue to the number check which is shown below.

```
[[ $DofW =~ ^[0-9]+$ ]]
```

The line above is the regex we used to ensure that the script does not allow anything other than a number to be input while creating a new crontab job. The '^' makes sure that the user's input begins with a number, the '+' allows for integers with multiple digits to input and the '$' checks that nothing is entered after the number.

```
((DofW >= 0 && DofW <= 7))
```

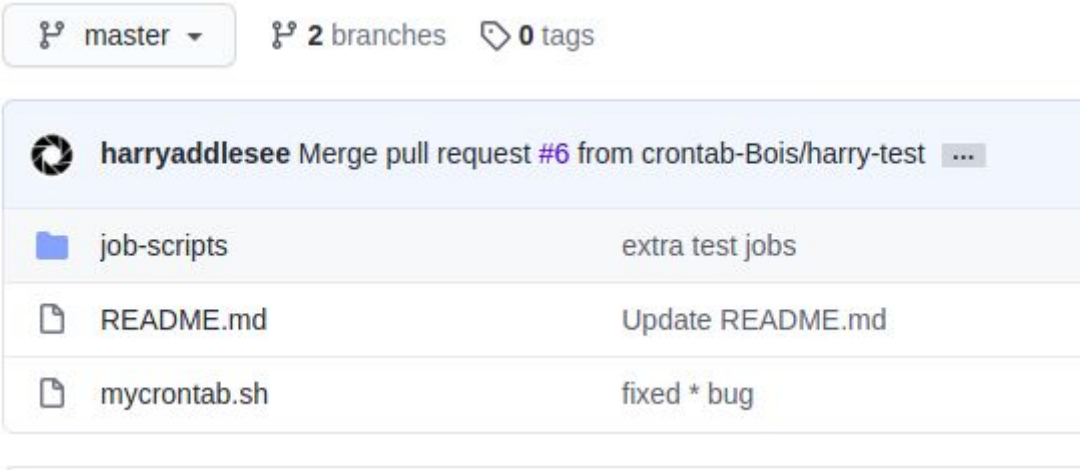
After the user's input is confirmed to be a number then the above line of code is executed to check if the entered number is within the specified range.

The screenshot below shows the process of option 2 and what happens if the user tries to enter a number out of the range or a random character.

```
2
Option 2 selected
When would you like your job to run?
Choose a day of the week (0-7) (Sunday = 0 or 7)
Enter an '*' if you would like all options selected:
2

Choose a month (1-12)
Enter an '*' if you would like all options selected:
*

Choose a day of the month (1-31):
Enter an '*' if you would like to select all options:
36

Please enter a valid number (1-31):
13

Choose an hour of the day (0-23):
Enter an '*' if you would like to select all options:
w

Please enter a number (0-23):
12

Choose a minute of the Hour (0-59:)
Enter an '*' if you would like to select all options:
35

Please type the name of the script you would like to run. Make sure that the script is saved
in your job-scripts folder. For example ./job-scripts/test1.sh you would type test1.sh
test1.sh
```

As shown above, a different error will be displayed for when the user attempts to enter a number out of range compared to when they try to enter an invalid character. The script will ask for a “valid number” if a number out of range is entered and will be asked to “enter a number” when an invalid character is entered, this makes it extremely clear to the user as to what error they have made.

At the end the user is asked to enter the name of the script that they would like to run in their crontab job. The code uses the ‘pwd’ command to input the path to the script. This means that our script will always work no matter where the user clones our GitHub repository on their machine. In our GitHub repository we have created a folder for users to enter their scripts to make it easier as it will only require them to input the script name rather than the path through their files. Our GitHub repository is shown in the screenshot below.



There are more complex operations that can be used while creating a new crontab job but as the coursework was specified for a new user, we decided to only give them the more basic options. An advanced user could use the crontab editor directly instead of using our user friendly script.

Editing a Job

Option 3 is for editing a job. The user is asked for the name of the job that they would like to edit and then is asked to enter the new timings for that job. The screenshot below shows the current jobs and then shows the process of editing one of those jobs (test1.sh).

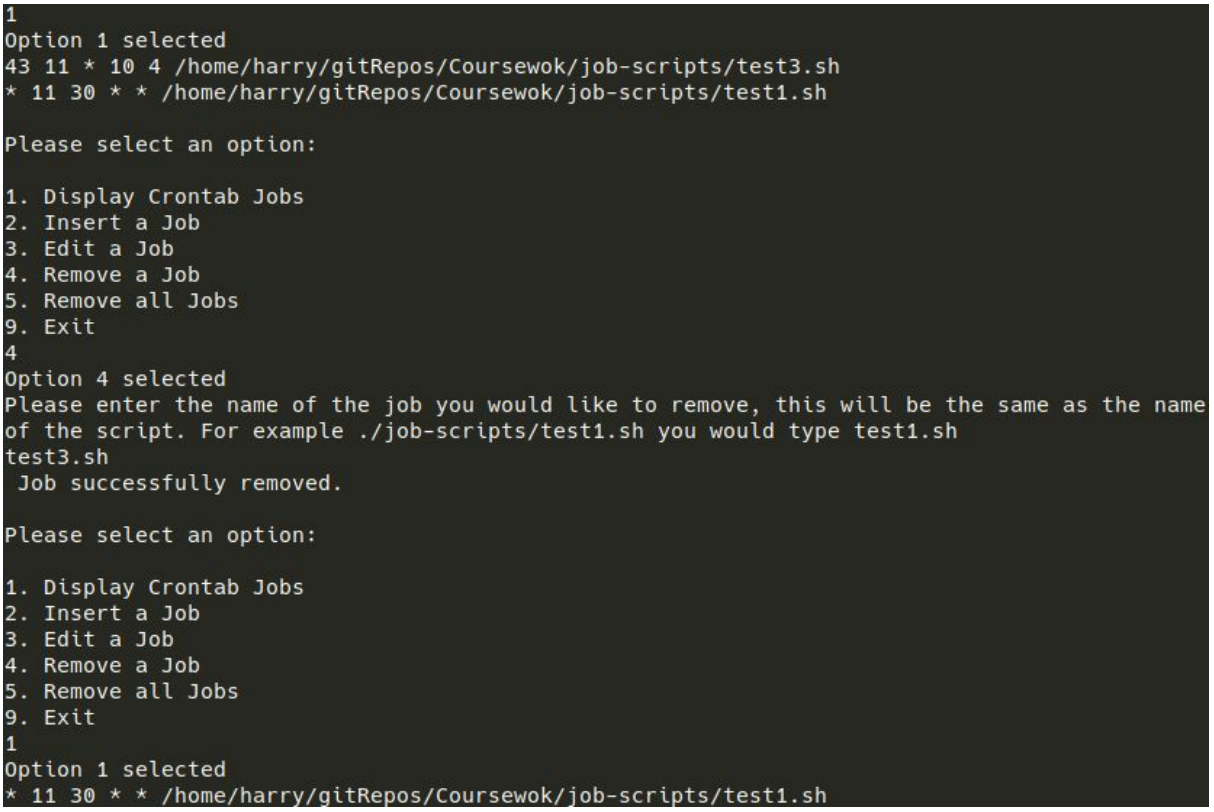
```
1
Option 1 selected
35 12 13 * 2 /home/harry/gitRepos/Coursewok/job-scripts/test1.sh
43 11 * 10 4 /home/harry/gitRepos/Coursewok/job-scripts/test3.sh

Please select an option:

1. Display Crontab Jobs
2. Insert a Job
3. Edit a Job
4. Remove a Job
5. Remove all Jobs
9. Exit
3
Option 3 selected
Please enter the name of the script you would like to edit, this will be the same as the name
 of the script. For example ./job-scripts/test1.sh you would type test1.sh
test1.sh
Please enter the new details of when you would like your job to run
Choose a day of the week (0-7) (Sunday = 0 or 7)
Enter an '*' if you would like all options selected:
*

Choose a month (1-12)
Enter an '*' if you would like all options selected:
*

Choose a day of the month (1-31):
Enter an '*' if you would like to select all options:
30

Choose an hour of the day (0-23):
Enter an '*' if you would like to select all options:
11

Choose a minute of the Hour (0-59:)
Enter an '*' if you would like to select all options:
*


Please select an option:

1. Display Crontab Jobs
2. Insert a Job
3. Edit a Job
4. Remove a Job
5. Remove all Jobs
9. Exit
1
Option 1 selected
43 11 * 10 4 /home/harry/gitRepos/Coursewok/job-scripts/test3.sh
* 11 30 * * /home/harry/gitRepos/Coursewok/job-scripts/test1.sh
```

As you can see in the above screenshot the job (test1.sh) has been updated with the new details. Our method allows for multiple edits to be made at the same time and if a user attempts to edit a non-existent crontab job, then it is created.

Removing a Job

Option 4 is for deleting a single job. When the user selects option 4 they are asked for the name of the job that they would like to remove and then a message is displayed when the job is successfully deleted. The screenshot below shows the

process of option 4 and that test3.sh has been removed from the current crontab jobs section.



Removing all Jobs

Option 5 is for removing all of the crontab jobs. We use the ‘crontab -ir’ command to give the user a warning before they delete all of their crontab jobs and make the user confirm their selection by entering a ‘y’.This process is shown in the screenshot below.

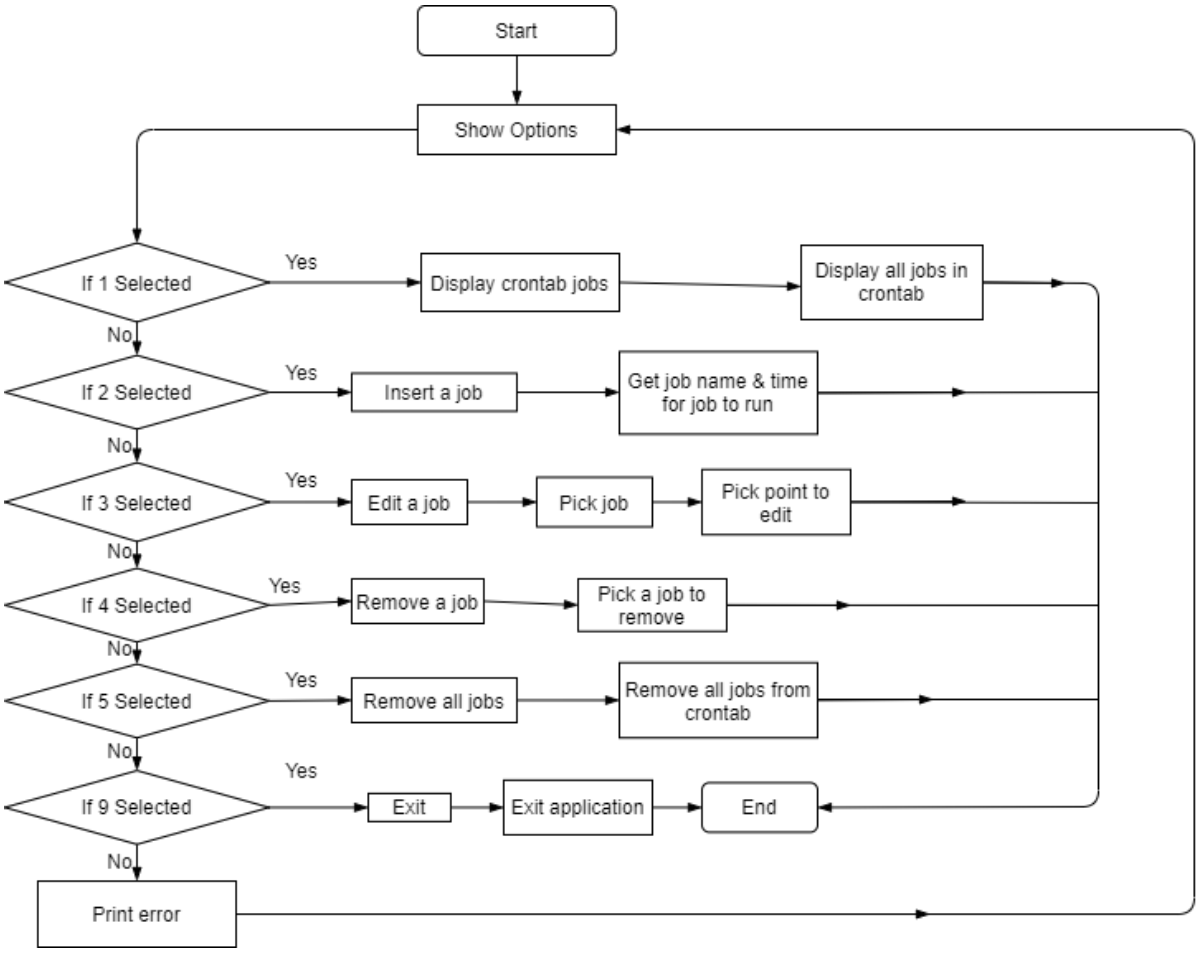
```
1
Option 1 selected
* 11 30 * * /home/harry/gitRepos/Coursewok/job-scripts/test1.sh
* * 16 11 3 /home/harry/gitRepos/Coursewok/job-scripts/test2.sh

Please select an option:

1. Display Crontab Jobs
2. Insert a Job
3. Edit a Job
4. Remove a Job
5. Remove all Jobs
9. Exit
5
Option 5 selected
This will delete all pre-existing cron jobs, press 'y' to confirm
crontab: really delete harry's crontab? y
All jobs removed successfully.

Please select an option:

1. Display Crontab Jobs
2. Insert a Job
3. Edit a Job
4. Remove a Job
5. Remove all Jobs
9. Exit
1
Option 1 selected
no crontab for harry
```

As seen above, all of the users' crontab jobs are removed and this is confirmed by selecting option 1 as it will tell the user that they have no crontab jobs.

Exiting

Finally when option 9 is selected the session is ended.

The flowchart below shows the process of the script in a more visual way.



Harry developed the script and wrote the documentation with contribution from Kian who created the flowchart above. All of our code can be found here: https://github.com/crontab-Bois/Coursewok