

Name Surname

Matriculation Number  
email

Game of dockers

Documentation

**Declaration**

I declare, except where explicit reference is made to the contribution of others\*, that this assignment is the result of my own work and has not been submitted for any module or programme degree at the Edinburgh Napier University or any other institution. This is in accordance with Edinburgh Napier University’s Academic Integrity Regulations.

\*IMPORTANT: Contribution of others may include use of Artificial Intelligence (AI) tools (details of which can be found in the Guidelines for Students on AI & Writing Assistant Tools). Please declare here whether you have used such tools, and to what extent:

NO I have not used such tools

YES I have used such tools and I have provided details below and included sample prompts and responses in an appendix.

If you answered YES here, please, in around 100 words, describe how (and at which points) you have used such tools to support your completion of this assessment.

1. Coursework Planning

Here you can describe how you worked on your coursework and the timeline of your activities.

01/11/23:

Prior to commencing the coursework, I read through the specification thoroughly and noted any key points and features. I then revisited the lectures and practical’s from week 4 and went to YouTube to watch some videos about Round Robin technique and Dockers to ensure I fully understood what I was meant to do and how they worked before starting my scripts.

8/11/23:

Starting the script, I did some research on organising the files in order of Shortest job next for docker 1, Shortest job next for docker 2 and leaving the files intact for docker 3. I then transferred the coursework files from moodle through my email to Ubuntu on VSOC.

10/11/23:

My priority was to get the docker containers up and running, and use some test files to see what was accessible from host etc. I ended up deciding that I would create the docker containers without running them, copy the files in from my dockerFile directories, then run the dockers to sort the files.

Once the files were sorted within the docker, I could then stop the dockers running and still be able to pull the files from the docker container to an output text file. As for my approach for the round robin, I am thinking of using a loop for each docker, then for each file within the docker, using a counter, excluding the number of files that is equal to the counter, and every 2 files moving onto the next docker container.

The user interface would be easier than the docker file handling, as I remembered much from the practical’s about text handling.

1. Implementation Details

Here you can talk about your implementation and possible tricks or arrangements that you made (e.g. usage of an extra library), which might not be communicated through your comments .

21/11/23

I created the dockers and copied the files in earlier in the month. The reason that I use ‘sudo docker create’ rather than ‘sudo docker run’ was because I needed to copy the files in from the host, and I was having trouble at the time copying the files in whilst dockers were running. I then moved onto the sort; I tried sorting the files in the docker:/home directory , but as I gained understanding of the ls -lS sort through practice and testing, I realised I would have to somehow store the variables in their sorted order, then handle the files accordingly; storing the files in an array, looping through that array, appending a count to the filename so that they stay sorted, and moving them to the new directory. Here I started my use of functions.

25/11/23

I created functions to create the dockers, copy files into them, and sort the files to clean up my script and make it easier for me to continue with the implementation. After implementing these functions, I could then call them in a loop for each docker container. Here I implemented the user interface using case statements, wanting to get as many tasks out of the way before I commenced the round robin function, which I knew would be the most time consuming.

26/11/23-10/12/23

Through trial and error, I ended up using an infinite loop that is only broken out of when the termination condition is met; this condition being that the files array for each docker container is empty. If even one array is not empty, it will loop again and try to process files from each array to the output file; this ensures that no files are left behind as each docker contains a different amount of files. If there are any filenames in the array, it writes the file contents to the output file and removes the file from the docker.