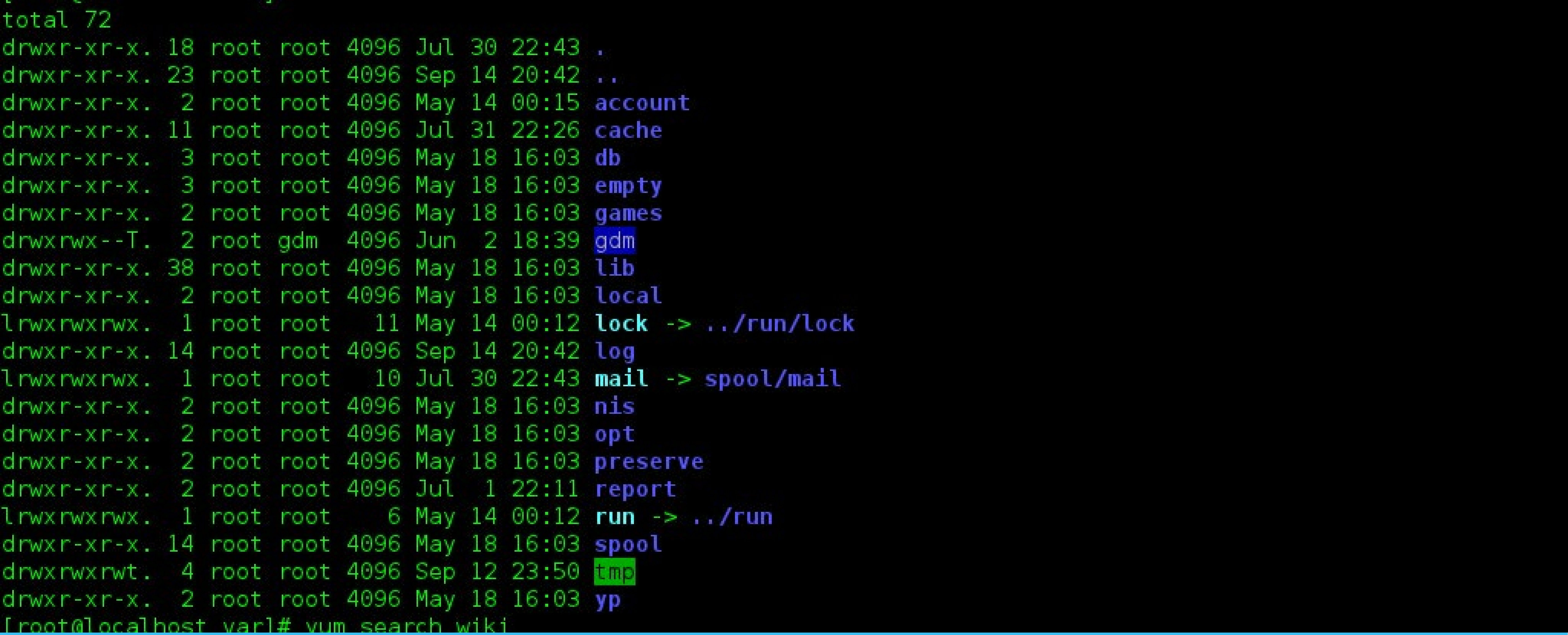

Command Line Arguments Lab

Clouds, Grids and Virtualisation

UNIVERSITY of GREENWICH

Command line arguments

To make the *main* function able to procesd command line arguments it must be declared as:

```
int main(int argc, char *argv[])
{
...
}
```

where *argc* variable will always reflect the number of parameters passed from the command line to the main function whilst the *argv* array of strings contains those parameters list.

Command line arguments

Typically:

argv[0] - contains path to the program that was executed,

argv[1] - contains string reflecting the first command line parameter,

...

argv[argc] - contains string reflecting the last command line parameter.

For example, for application *arguments.out* executed from */home/<username>* as:

```
$ ./arguments.out -l file_name
```

within the *main* funciton:

argc will be equal 2,

argv[0] - will contain string "./a.out",

argv[1] - will contain string "-l",

argv[2] - will contain string "file_name"

Slurm Script

#!/bin/bash	
#SBATCH --job-name=Arguments	- Name of the job
#SBATCH --output=./arguments.txt	- Name of command line output file
#SBATCH --cpus-per-task=1	- Number of CPUs per executable (serial code so 1)
#SBATCH --ntasks=1	- Number of executables running
#SBATCH --ntasks-per-node=1	- Number of executables running on each node
#SBATCH --nodes=1	- Number of nodes
#SBATCH --partition=COMP1680-dev	- Which queue to use
gcc arguments.c -o arugments.out	- Compile the code
./arguments.out 2 3 3.141	- Run the code with the command line arguments 2,3,3.141

To run the script use: sbatch <scriptname>

Part 1

1. Download arguments.c and arguments.sh from Moodle
2. Upload them to the HPC
3. Look at the code, make sure you understand what it does
4. Compile and run on the HPc using the slurm script
    1. Use sbatch arguments.sh
    2. The output will display in a text file called arguments.txt
    3. Change the arguments in the slurm script, how does the ouput change?
    4. You can also run it using salloc -n 1 -N 1 –p COMP1680-dev ./arguments.out 2 3 3.14

Part 2

Consider the following pseudocode

```
read n from command line
sum = 0
for i = 1 to n do
    a[i] = i*i
    print i + " " + a[i]
    sum = sum + a[i]
end do
print "The sum of " + n + "squares is" + sum
```

Modify the arguments.c program to perform the same operations as the pseudocode, you'll also need to modify your slurm script

Save the code, upload, compile and run on the hpc.

Add timings to the code

How quickly does it run for 100 squares?

Why does it run quicker without printing each loop?

Plot a graph of run time against n for 100 – 1000 squares, incrementing by 100, with and without printing each loop.
- hint you can get the slurm script to run the program multiple times with different arguments this is one of the uses of batch scripts!