

## Assignment1 – Dario Placencio

<https://github.com/placenciohid/repo759>

1. I went through a) through c) and understand how to time code, how to submit my assignments with git, and what the recommended workflow is when it comes to working on my assignment
2. Write one line of bash code for each of the following sub-tasks (assume that all the files and directories mentioned exist). The purpose of this task is to get you familiar a bit with the Linux command line.

- a. Change the current directory to a subdirectory called somedir

```
cd somedir
```

- b. Print out to the terminal the contents of a file called sometext.txt. The file exists in the current directory.

```
cat sometext.txt
```

- c. Print out to the terminal the last 5 lines of a plain text file called sometext.txt. The file exists in the current directory.

```
tail -n 5 sometext.txt
```

- d. Print out to the terminal the last 5 lines of each file that ends in the extension .txt and lives in the current directory

```
tail -n 5 *.txt
```

- e. Write a for loop which prints each integer from 0 to 6 (including 0 and 6).

```
for i in {0..6}; do echo $i; done
```

3. The purpose of this task is to get you familiar using Euler. On Euler, using the module command, answer the following questions.

- a. Are there any modules loaded (module list) when you log in on Euler ?

No there are no modules loaded after logging in

```
[placenciohid@euler-login-1 ~]$ module list
```

No modules loaded

- b. What version (version number) of gcc is available to you without loading any modules?

```
[placenciohid@euler-login-1 ~]$ gcc --version
```

gcc (GCC) 14.1.1 20240522 (Red Hat 14.1.1-4)

Copyright (C) 2024 Free Software Foundation, Inc.

This is free software; see the source for copying conditions. There is NO

warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

- c. List all cuda modules available on Euler.

```
[placenciohid@euler-login-1 ~]$ module avail cuda
```

```
----- /opt/apps/lmod/modulefiles -----
```

```
gromacs/cuda-12.2-mpich/2023.3  nvidia/cuda/11.3.1  nvidia/cuda/12.1.0
nvidia/nvhpc-hpcx-cuda12/23.11
gromacs/cuda-12.2/2023.3      nvidia/cuda/11.6.0  nvidia/cuda/12.2.0
nvidia/nvhpc-hpcx-cuda12/24.5 (D)
nvidia/cuda/10.2.2           nvidia/cuda/11.8.0  nvidia/cuda/12.5.0      (D)
nvidia/cuda/11.0.3           nvidia/cuda/12.0.0  nvidia/nvhpc-hpcx-cuda11/24.5
```

Where:

D: Default Module

If the avail list is too long consider trying:

"module --default avail" or "ml -d av" to just list the default modules.

"module overview" or "ml ov" to display the number of modules for each name.

Use "module spider" to find all possible modules and extensions.

Use "module keyword key1 key2 ..." to search for all possible modules matching any of the "keys".

- d. List one other piece of software that has a module on Euler and write one sentence about what it does. (If you aren't familiar with any of the other software, google one up and write a sentence about it.)

Julia 1.6.1 is available, I used this program/language on my class of introduction to optimization, and it was really good for working on optimization problem, but overall, a great tool for high speed and efficiency in computing tasks.

4. Write a bash script called task4.sh with a Slurm header which asks for

- 2 CPU cores
- A job name of FirstSlurm
- An output file called FirstSlurm.out
- An error file called FirstSlurm.err

and runs a single command to print the hostname of the machine (compute node) running the job. This job must be submitted by running sbatch task4.sh on Euler's head node.

5. Research some useful Slurm tools (one sentence responses):

- a. In what directory does a Slurm job on Euler begin execution? You may run some jobs in different directories to check this.

In the same directory from which the Slurm job was submitted.

- b. Explain what SLURM JOB ID is in the environment of a running Slurm job.  
It represents a unique identifier for the job.
  - c. How would you track the status of job(s) run by yourself? Assume that the job(s) have not been completed yet.  
The command `squeue -u <username>` displays a track of the status of the jobs.
  - d. How would you cancel a job submitted by yourself? Assume that the job is still in the queue.  
Using the command `scancel <jobid>`, I actually had to do this on this exercise cause I forgot a line of code at the beginning.
  - e. Explain what the following script header line specifies: `#SBATCH --gres=gpu:1`  
A request for 1 gpu for the job.
  - f. (Optional) Explain what the following script header line specifies: `#SBATCH --array=0-9`  
Creation of an array 10 job with 10 sub-jobs, indexed 0 to 9.
6. Write a C++ program called `task6.cpp` that:
- a. Takes a command line argument `N`. (If you are confused about command line arguments, it may be helpful for you to read this)
  - b. Prints out each integer from 0 to `N` (including 0 and `N`) in ascending order with the `printf` function.
  - c. Prints out each integer from `N` to 0 (including `N` and 0) in descending order with `std::cout`.  
For each printing process, the integers should be separated by spaces on a single line ending in a newline.
  - d. Compile command: `g++ task6.cpp -Wall -O3 -std=c++17 -o task6`
  - e. Run command (do not run this on Euler's head node; do it through Slurm): `./task6 N`  
- Expected output (followed by a newline):  

```
0 1 2 3 ... N
N ... 3 2 1 0
```

- Example expected output for `N = 6` (followed by a newline):  

```
0 1 2 3 4 5 6
6 5 4 3 2 1 0
```

I have basically no knowledge of C++ before this class, so I have used this prompt to help me construct the code on a LLM:

Help me build the basic skeleton in C++ for a code that takes a line argument N, print out each integer from 0 to N in ascending order and descending order, separated by spaces on a single line ending with a new line.



Here is a basic skeleton in C++ that accomplishes the task:

```
C++

#include <iostream>

int main(int argc, char* argv[]) {
    if (argc != 2) {
        std::cerr << "Usage: " << argv[0] << " <N>" << std::endl;
        return 1;
    }

    int N = std::stoi(argv[1]);

    // Print numbers in ascending order
    for (int i = 0; i <= N; ++i) {
        std::cout << i << " ";
    }

    // Print numbers in descending order
    for (int i = N; i >= 0; --i) {
        std::cout << i << " ";
    }

    std::cout << std::endl;

    return 0;
}
```

