Intro to HPC Exercise

Lab Exercise: Introduction to HPC

The assumption is that you have already tested your Amazon Web Service Elastic Compute Cloud (EC2) virtual machines chosen for the LCI hands on exercises. The virtual machines in EC2 are running Red Hat Linux 7.3 as the base operating system.

Actions and commands that you should perform or enter are in the computer **boldface** font & are in Blue.

After each Unix command you type at the prompt (explained below), press the key.

If you have any difficulty with this exercise, raise your hand and one of the instructors will be able to help you. If you are unable to get any help kindly come and see me or as a last resort send an e-mail to: irfan@ucar.edu and I will find someone to help you.

If you are having trouble entering the source code examples using an editor, you can also download the soft-copy of the exercise instructions & code from GitHub Repo at:

https://github.com/WyoARCC/build-a-cluster

This will allow you to cut/paste the sample source example into your source file.

HOW TO USE THIS DOCUMENT

1. READ.

2. Follow the instructions.

3. After you read and followed the instructions and still have issues, then feel free to ask questions.

We URGE that you read each numbered item all the way through. If you already have significant experience with Unix-like operating systems such as Linux, then this exercise will probably go very quickly; if you're new to Unix-like or Linux-Like operating systems, then it'll probably take a bit longer.

1. At the command prompt (This will change after first login):

   ```
   [ec2-user@ip-10-0-0-4 ~]$
   ```

2. Check your home directory by issuing the command pwd (print working directory):

   ```
   pwd
   ```

3. Some hints and tips on UNIX/Linux.
   a. Filenames are case sensitive, meaning that it matters whether you use upper case (capital) or lowercase (small) for each letter in a filename.
   b. Pieces of a filename (or actually of the directory that a file is in) are separated by slashes.
   c. The symbol ~ (known as a tilde, pronounced "TILL-duh") denotes your home directory (another way to denote your home directory is ~yourusername). On most keyboards, the tilde ~ is in the upper left, just above the Tab key.
4. Create a subdirectory of your home directory named LCI, like so:

   ```
   mkdir ~/LCI
   ```

   This command means: "Create a directory named LCI as a subdirectory inside my home directory." You WON'T have to do this for future logins. Confirm that you have successfully created your LCI directory by listing the contents of the current working directory:

   ```
   ls  (ls -l)
   ```

   "List the names of the files and subdirectories in my current working directory." Use "man ls" to learn about all the different flags/options of this command. The "ls -l" gives you a long listing output of the ls command.

5. Set the permissions on your LCI directory so that only you can access it:

   ```
   chmod u=rwx,go= LCI
   ```

   This command means "change the mode (list of permissions) on my subdirectory named LCI so that I (the user) can read files in it, write files in it, and go into

(execute) it, but nobody else can." Your LCI directory is now accessible only to you. The only other people who can access it are the system administrators (sysadmins for short) and users who have root privileges on the system.

6. Make sure you have the exercises copied from the master directory to your own local LCI directory. Confirm that you're in your home directory:

   ```
   pwd
   ```

   Check that you have a LCI subdirectory inside your home directory:

   ```
   ls
   ```

   Go into your LCI subdirectory:

   ```
   cd LCI
   ```

   This command means: "Change the working directory to LCI, which is a subdirectory of my current working directory." Confirm that you're in your LCI subdirectory:

   ```
   pwd
   ```

   See what files or subdirectories (if any) are in the current working directory:

   ```
   ls
   ```

   You may get no output, just the Unix prompt; if so, that indicates that your current working directory has no files or subdirectories in it. To learn more about a particular Unix command, enter: man commandname

   ```
   man chmod
   ```

   The example above will give you the online manual page for the chmod command. The output of man goes through another command, more, which shows one screenful at a time. To get the next screenful, press the spacebar; to get the next line, press the Enter key. To quit the more command, press the Q key.

Create a file called filea in your sub-directory called LCI.

```
pwd
cd ./LCI/intro
vi filea
```

Press “i” to enter vi’s “input” mode. Enter any text you want. Now save the file (press escape key and then :wq)

```
cp ~/LCI/intro/filea ~/LCI/intro/fileb
ls
```

This command means: “Make a copy of the **filea** and name the second **fileb** & put the file **fileb** in in the subdirectory intro which is inside the directory called LCI.”
Confirm that the Intro subdirectory was copied into your LCI/intro directory:

```
ls
```

7. Go into your Intro subdirectory:

   ```
   cd Intro
   ```

   Confirm that you’re in your Intro subdirectory:

   ```
   pwd
   ```

   See what files or subdirectories (if any) are in the current working directory (Intro):

   ```
   ls
   ```

8. Enter the C and FORTRAN source code and compile the source and get it ready so you can run it. For your convenience the source code is also available below.

   If you are having issues entering the source code into a file you can also download them from GitHub.

   https://github.com/WyoARCC/build-a-cluster

You should be able to simply type/enter the source code yourself into a file using your favorite editor. Red Hat Linux provides the following editors by default (vi, vim, emacs).

Here is a sample source for “Hello World” in C.

```
<-------snip C Hello World (intro_hpc.c)----------------->
/* Hello World program in C */

#include <stdio.h>
main()
{
printf("Hello World");
Return EXIT_SUCCESS;
}
<--------------------snip C Hello World-------------------->
```

Here is a sample source for “Hello World” in FORTRAN.

```
<--------snip Fortran Hello World (intro_hpc_1.f90------>
!*******************************************************
! Simple Hello World is FORTRAN.
!
 program hello
         print *, "Hello World!"
     end program hello
<--------------snip Fortran Hello World------------------>
```

Or try this one and observe the difference between the outputs of the two.

```
<--------snip Fortran Hello World (intro_hpc_2.f90------>
!*******************************************************
! Simple Hello World is FORTRAN.
!
 program Hello
         print *, "Hello World!"
     end program Hello
<--------------snip Fortran Hello World------------------>
```

Check your current working directory.

```
pwd
```

If you are not in the intro directory, then:

```
cd ~/LCI/intro
```

```
pwd
```

```
vi intro_hpc_1.f90
```

Enter the code for the 1st Fortran Hello World. See above for details.

```
gfortran -o hello.f1 intro_hpc_1.f90
```

```
vi intro_hpc_2.f90
```

```
gfortran -o hello.f2 intro_hpc_2.f90
```

Enter the code for the 2nd Fortran Hello World. See above for details.

```
vi intro_hpc.c
```

Enter the code for the C Hello World. See above for details.

```
gcc -o hello intro_hpc.c
```

9. Running/executing your “Hello World” job.

```
cd ~/intro
```

```
./hello.f1
```

```
./hello.f2
```

```
./hello
```

Did you notice anything different in your programs.

```
←-------snip C Hello World (intro_hpc.c)----------------->
/* Hello World program in C */

#include <stdio.h>
main()
{
printf("Hello World");  /* No automatic newline */
Return EXIT_SUCCESS;
}
←--------------------snip C Hello World---------------------->

←-------snip C Hello World (intro_hpc.c)----------------->
/* Hello World program in C */

#include <stdio.h>
main()
{
printf("Hello World\n");  /* This will add a newline */
Return EXIT_SUCCESS;
}
←--------------------snip C Hello World---------------------->
```