



LEADERSHIP
COMPUTING
FACILITY

ADMI Symposium 2026

HPC Crash Course

Togo Odbadrakh, Elijah MacCarthy



U.S. DEPARTMENT OF
ENERGY

ORNL IS MANAGED BY UT-BATTELLE LLC
FOR THE US DEPARTMENT OF ENERGY

FRONTIER



SCHEDULE

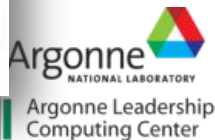
11:00	Intro to OLCF & HPC?
11:05	The Unix Operating System
11:25	The Vim Text Editor
11:45	Slurm Job Submission
11:50	Demo

Oak Ridge Leadership Computing Facility (OLCF)

- One of two Department of Energy LCF's
- Based in Oak Ridge, TN at the Oak Ridge National Laboratory (ORNL)
- Department of Energy-funded research
 - Neutron Science, High-Performance Computing, Advanced Materials, Biology and Environmental Science, Nuclear Science and Engineering, Isotopes, and National Security
- Largest, most modern center for unclassified computing in the US



We are one of the DOE's Office of Science computational user facilities



Argonne Leadership Computing Center

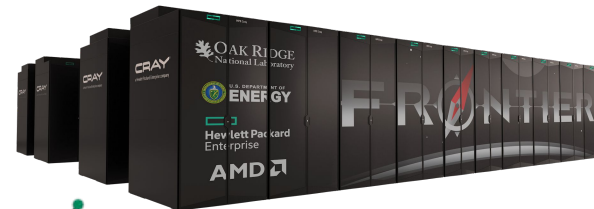


Oak Ridge Leadership Computing Center



NERSC

National Energy Research Scientific Computing Center



- DOE is leader in open high-performance computing
- Provide the world's most powerful computational tools for open science
- Access is free to researchers who publish
- Boost US competitiveness
- Attract the best and brightest researchers

One Big Job

“**exa**” = 1,000,000,000,000,000,000

TOP 500 The List.

HOME LISTS STATISTICS RESOURCES ABOUT MEDIA KIT

Home »Lists »Top500 »June 2024 »List

TOP500 LIST - JUNE 2024

R_{max} and **R_{peak}** values are in PFlop/s. For more details about other fields, check the TOP500 description.

R_{peak} values are calculated using the advertised clock rate of the CPU. For the efficiency of the systems you should take into account the Turbo CPU clock rate where it applies.

← 1-100 **101-200** 201-300 301-400 401-500 →

Rank	System	Cores	R _{max} (PFlop/s)	R _{peak} (PFlop/s)	Power (kW)
1	Frontier - HPE Cray EX235a, AMD Optimized 3rd Generation EPYC 64C 2GHz, AMD Instinct MI250X, Slingshot-11, HPE DOE/SC/Oak Ridge National Laboratory United States	8,699,904	1,206.00	1,714.81	22,786
2	Aurora - HPE Cray EX - Intel Exascale Compute Blade, Xeon CPU Max 9470 52C 2.4GHz, Intel Data Center GPU Max	9,264,128	1,012.00	1,980.01	38,698

25 YEARS ANNIVERSARY

TOP500 LIST

NEWSLETTER SIGN UP

1.2 exaflops

OLCF Frontier Overview

The system includes

- 9,408 nodes
- HPE Slingshot interconnect with 200 GbE interfaces
- 679 PB multi-tier Lustre filesystem “Orion”

System Performance

- Peak of 1.206 ExaFlop/s for modeling & simulation
- Peak of 9.95 ExaOps/s for data analytics and artificial intelligence

Each node has

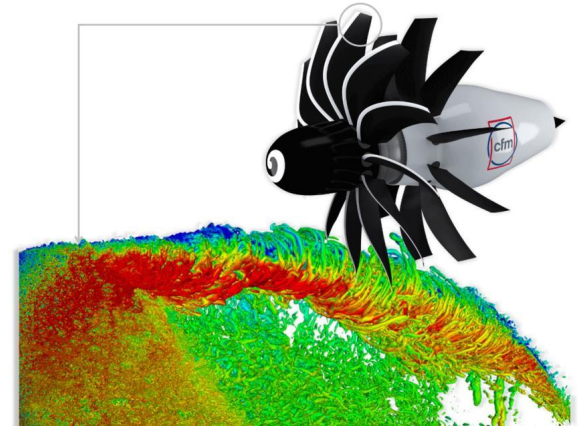
- 1 AMD Optimized 3rd Gen EPYC 64 core processor
- 4 AMD MI250X Instinct GPUs (8 GCDs)
- 640 GB of fast memory (128 GB HBM2 + 512 GB DDR4)
- 3.48 TB of NV memory (2 X 1.92TB NVMe SSDs)



First to Exascale!

Example Use Case

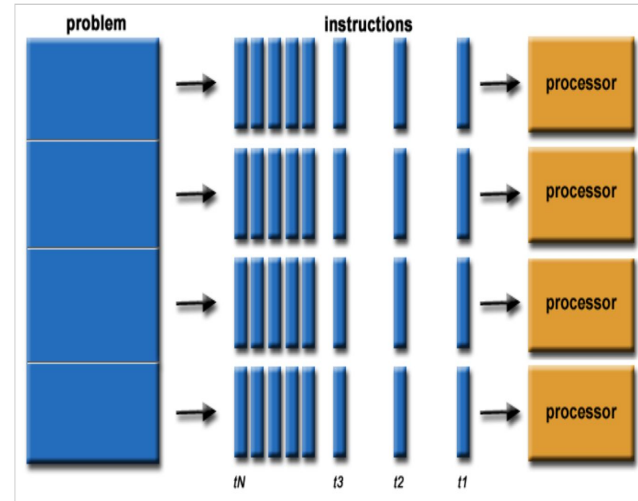
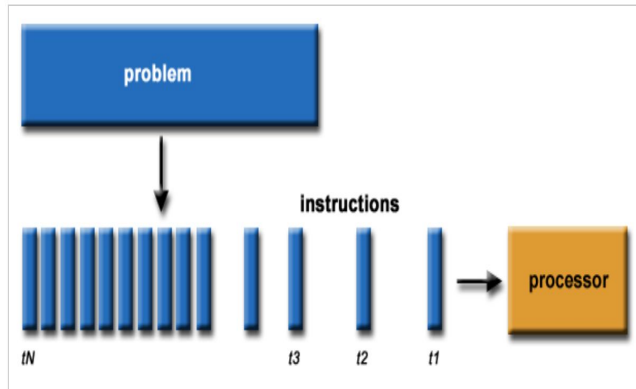
- GE Aerospace is designing next-gen commercial aircraft engines
- Frontier enables better evaluation of engine technologies at flight scale
- Software designed to model engine performance and noise levels
- Resulting runs simulated air movement for a full-scale open fan engine design with incredible detail



Credit: CFM, GE Research
https://www.ornl.gov/sites/default/files/styles/main_image_style/public/2023-08/GEAerospaceEngine.jpg

What is High Performance Computing?

- High Performance Computing (HPC) is about solving the world's largest engineering and science problems with supercomputers.
- That means doing work efficiently in parallel



Images courtesy of :

<https://hpc.llnl.gov/training/tutorials/introduction-parallel-computing-tutoria>

Parallel Algorithms

Just you and one washer and one dryer



Optimizing just you and one washer and one dryer

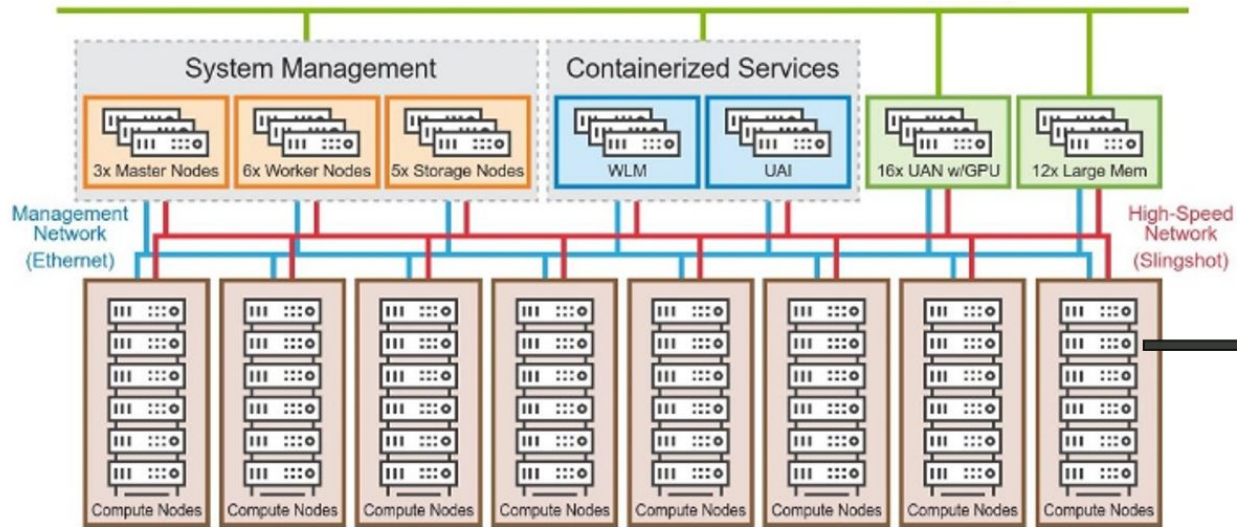


You your housemate, 2 washers, and 2 dryers



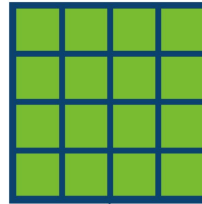
What is High Performance Computing?

- High Performance Computing (HPC) is about solving the world's largest engineering and science problems with supercomputers.
- That means doing work efficiently in parallel

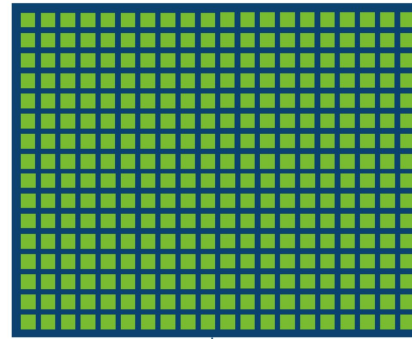


Processors

- Modern HPC nodes combine Central Processing Units CPUs with accelerators such as Graphical Processing Units GPUs.
 - CPUs generally have several compute cores that can run 10s of tasks at a time
 - GPUS have streaming multiprocessor that can run 1000s of tasks at a time.



CPU



GPU

The Unix Operating System

The Unix Operating System

- HPC Clusters operate on the Unix operating system

Born 1969

At Bell Labs



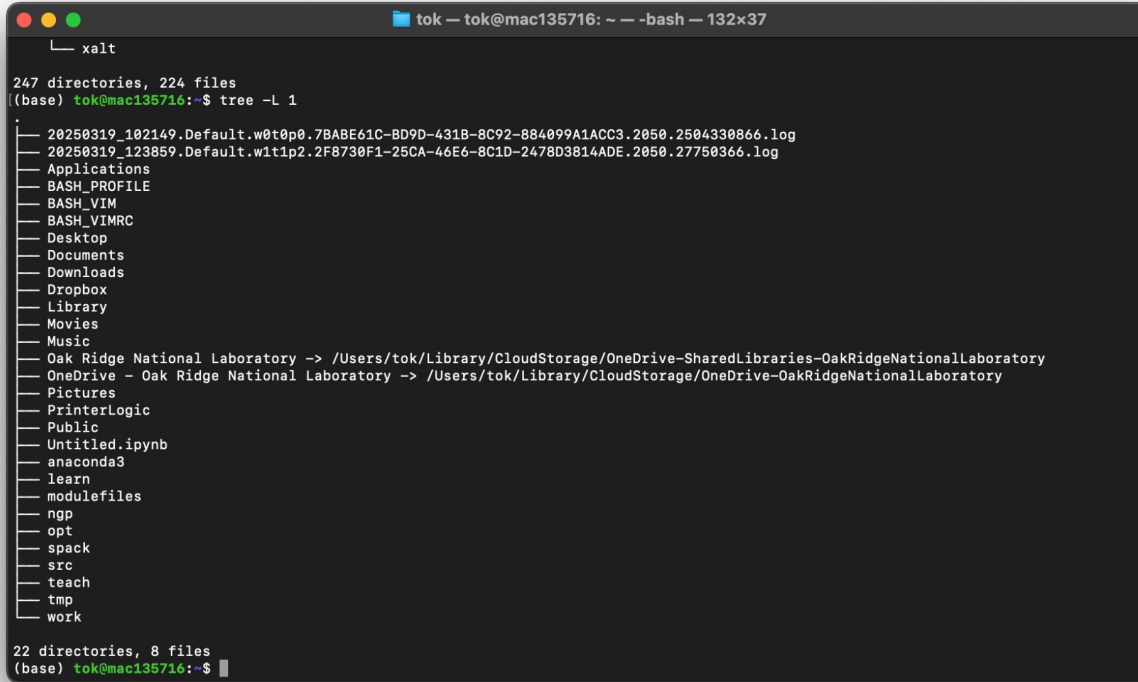
The Unix Operating System

- The Unix OS operates via the command line interface

```
demo — tok@mac135716: ~/work/admi2025/demo — -bash — 102x27
-rw-r--r--@ 1 tok ORNL\Domain Users 32239 Feb 25 09:55 Domestic_Travel Questionnaire .docx
-rw-----@ 1 tok ORNL\Domain Users 37254 Mar 10 09:13 Pledge_1225_from_Association_of_Computer_Sc
ience_Departments_at_Minority_Institutions.pdf
-rw-r--r--@ 1 tok ORNL\Domain Users 35127 Mar 11 15:25 Receipt_1225.pdf
-rw-r--r--@ 1 tok ORNL\Domain Users 7033207 Mar 19 16:41 admi-2025-hpc-crash-course.pptx
(base) tok@mac135716:~/work/admi2025$ mkdir demo
(base) tok@mac135716:~/work/admi2025$ cd demo/
(base) tok@mac135716:~/work/admi2025/demo$ ll /
total 10
drwxrwxr-x 67 root admin 2144 Mar 24 11:36 Applications
drwxr-xr-x 77 root wheel 2464 Mar 12 10:40 Library
lrwxr-xr-x 1 root wheel 28 Mar 12 10:41 Network -> /System/Volumes/Data/Network
drwxr-xr-x@ 10 root wheel 320 Mar 6 05:06 System
drwxr-xr-x 6 root admin 192 Mar 12 10:40 Users
drwxr-xr-x 5 root wheel 160 Mar 26 11:41 Volumes
drwxr-xr-x@ 39 root wheel 1248 Mar 6 05:06 bin
drwxr-xr-x 2 root wheel 64 Apr 1 2023 cores
dr-xr-xr-x 4 root wheel 4696 Mar 12 10:39 dev
lrwxr-xr-x@ 1 root wheel 11 Mar 6 05:06 etc -> private/etc
lrwxr-xr-x 1 root wheel 25 Mar 12 10:41 home -> /System/Volumes/Data/home
drwxr-xr-x 6 root wheel 192 Oct 28 16:33 opt
drwxr-xr-x 6 root wheel 192 Mar 12 10:41 private
drwxr-xr-x@ 77 root wheel 2464 Mar 6 05:06 sbin
lrwxr-xr-x@ 1 root wheel 11 Mar 6 05:06 tmp -> private/tmp
drwxr-xr-x@ 11 root wheel 352 Mar 6 05:06 usr
lrwxr-xr-x@ 1 root wheel 11 Mar 6 05:06 var -> private/var
(base) tok@mac135716:~/work/admi2025/demo$
```

The Unix Operating System

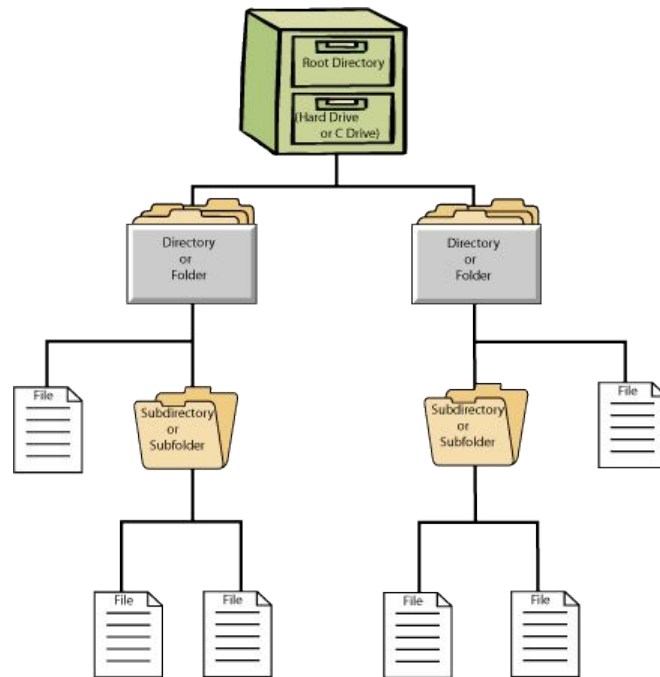
- The Unix OS is file-based



```
tok — tok@mac135716: ~ -- -bash — 132x37
└─ xalt
247 directories, 224 files
(base) tok@mac135716:~$ tree -L 1
├── 20250319_102149.Default.w0t0p0.7BABE61C-BD9D-431B-8C92-884099A1ACC3.2050.2504330866.log
├── 20250319_123859.Default.w1t1p2.2F8730F1-25CA-46E6-8C1D-2478D3814ADE.2050.27750366.log
├── Applications
├── BASH_PROFILE
├── BASH_VIM
├── BASH_VIMRC
├── Desktop
├── Documents
├── Downloads
├── Dropbox
├── Library
├── Movies
├── Music
├── Oak Ridge National Laboratory -> /Users/tok/Library/CloudStorage/OneDrive-SharedLibraries-OakRidgeNationalLaboratory
├── OneDrive - Oak Ridge National Laboratory -> /Users/tok/Library/CloudStorage/OneDrive-OakRidgeNationalLaboratory
├── Pictures
├── PrinterLogic
├── Public
├── Untitled.ipynb
├── anaconda3
├── learn
├── modulefiles
├── ngp
├── opt
├── spack
├── src
├── teach
├── tmp
└── work
22 directories, 8 files
(base) tok@mac135716:~$
```

The Filesystem 1

- **File** – A collection of data
 - Plain text, Input/output, a program (executable), an image, etc.
- **Directory** – A logical structure to help organize files (think “folder”)
- **Filesystem** – A collection of files and directories
- As a user, you land in your “home” directory when you log in, and can create directories and files there, or move elsewhere to do that.

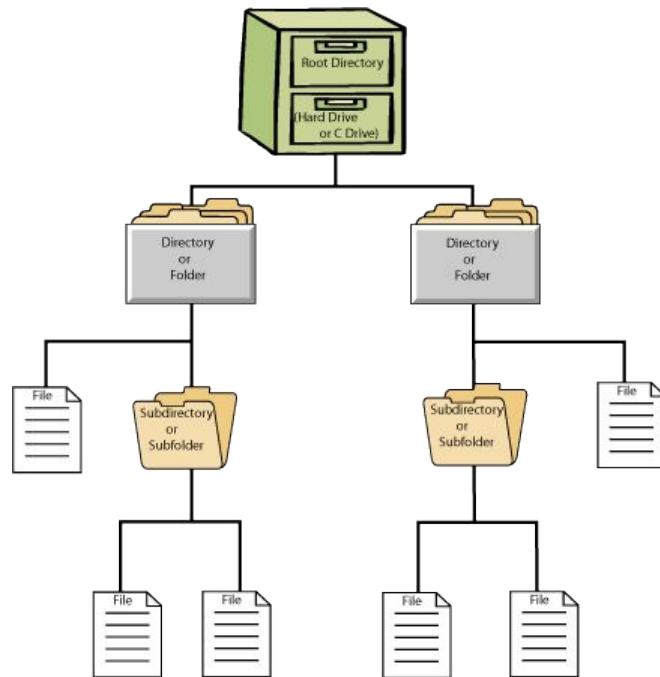


The Filesystem 2

- Think of the filesystem as a tree. Instead of branches, there are “directories. Instead of leaves, there are “files”
- It starts at /, which is called the “root” directory
- The slash (/) is the directory separator; thus when we move into subdirectories it’s used to separate things (i.e. /ccsopen/home/your_username)

If you are ever “lost”, use the pwd command. It prints your location.

```
$ pwd  
/ccsopen/home/your_username
```



Basic Commands

- Commands are usually abbreviations of words (or a series of words)
 - cp for “Copy”, rm for “Remove”, cd for “change directory”
- Commands tend to follow the following syntax:

```
[command] [option flags] [file/directory/object to act upon]
```

- Almost all commands take various options to control what they do.
- Some help is available via an online manual
 - The `man` command
 - Example: Want info about `ls`, type the following:

Basic File/Directory Commands

Command	Description
pwd	Print w orking (i.e. current) d irectory
man	Display the M anual for a command
whoami	Display the current user's username
mkdir	Create a directory (M a K e D IRectory)
rmdir	Delete a directory (R e M ove D IRectory)
cd	C hange (into a) d irectory
ls	L ist files
cp	C opy a file
mv	M ove a file (also used to rename a file)
rm	R emove (delete) a file
cat	Display the contents (con cat enate) a (hopefully text) file
less	Display the contents of a text file in a viewing mode
find	F ind files by filename
grep	Search for text within files
diff	Identify d ifferences in the contents of files

Directories

Use `pwd`, `cd`, `mkdir`, and `rmdir` commands to navigate the filesystem and manipulate directories

```
$ pwd
/home/user1

$ mkdir dir1

$ ls
dir1

$ cd dir1

$ pwd
/home/user1/dir1

$ cd ..

$ rmdir dir1
```

Typing just `cd` will always take you back to home no matter where you are.

Directories must be empty in order to delete them with `rmdir`

Special Directories

- Every directory contains two special directory entries: "." and ".."
- . is a reference to the current directory
- .. is a reference to the parent directory (so we can do things like `cd ..`)
- ~ can be a reference to home directories
 - ~/ is yours
 - ~user1/ is user1's
- You'll see how these are useful later

Manipulating Files 1

- The `rm` command is used to delete a file.
- The `mv` command is used to move and rename files
- There are multiple ways to create and view text files. In the challenge, we will look at various ways to use `cat` and `less` commands to do this.
- Utilities such as `less` and `cat` are intended only for text files. The system will not stop you from running them on a non-text file
 - If you do, you'll get a screenful of unintelligible characters
 - You might get a recognizable prompt (you might not)
 - There's no shame in closing that session's window & re-connecting

Manipulating Files 2

- The `cp` command is used to copy a file, and the `mv` command is used to move and rename files.

```
$ ls
dir1      filea    fileb

$ cat filea
This is a file that
contains three lines
of text.

$ cp filea filea1

$ ls
dir1      filea    filea1   fileb

$ mv filea1 filec

$ ls
dir1      filea    fileb    filec
```

Manipulating Files 3

- The `rm` command is used to delete a file. `cat` prints contents of a file to the screen. `less` displays the contents of a file in a viewing mode. (Press “q” to exit).

```
$ cat filec
This is a file that
contains three lines
of text.

$ less filec

$ mv filec dir1

$ ls dir1
filec

$ rm dir1/filec

$ ls
dir1      filea    fileb
```

Many uses of cat

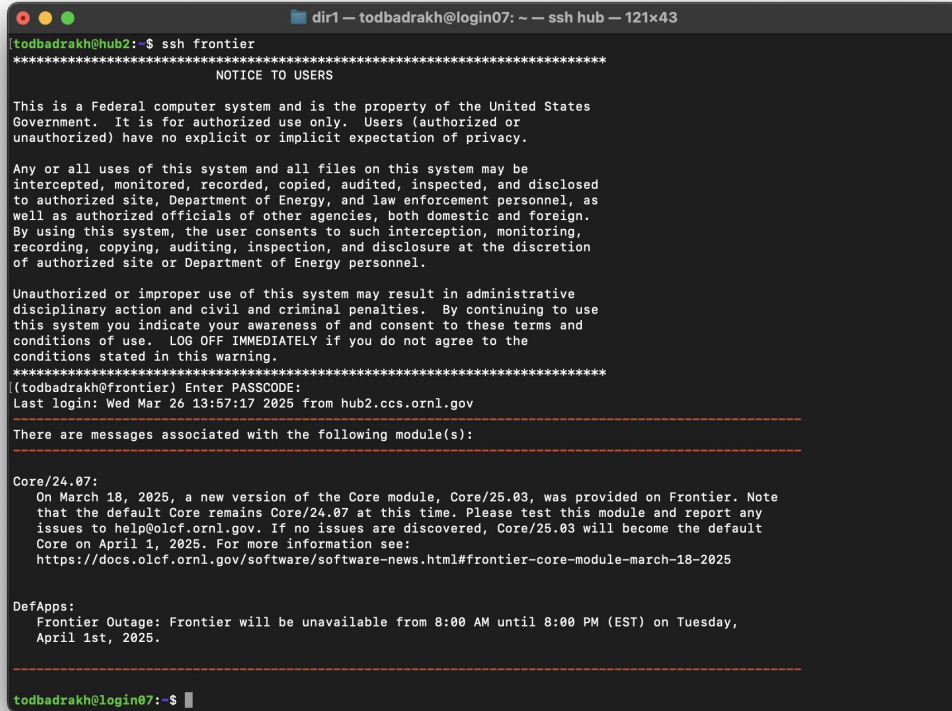
Command	Explanation
<code>cat file1.txt</code>	Display contents of file
<code>cat file1.txt file2.txt</code>	Concatenate two text files and display the result in the terminal
<code>cat file1.txt file2.txt > newcombinedfile.txt</code>	Concatenate two text files and write them to a new file
<code>cat >newfile.txt</code>	Create a file called newfile.txt. Type the desired input and press CTRL+D to finish. The text will be in file newfile.txt.
<code>cat -n file1.txt file2.txt > newnumberedfile.txt</code>	Some implementations of cat, with option -n, can also number lines
<code>cat file1.txt > file2.txt</code>	Copy the contents of file1.txt into file2.txt
<code>cat file1.txt >> file2.txt</code>	Append the contents of file1.txt to file2.txt
<code>cat file1.txt file2.txt less</code>	Run the program "less" with the concatenation of file1 and file2 as its input

Searching Within Files – grep 1

- Sometimes you want to search for patterns/strings in a file. As with other commands, `grep` takes many options. The `grep` command searches for “regular expressions” ...strings that contain characters with special meaning
- Simple case: find lines with the string ‘user’ in file1
`grep “user” file1`
- More complex: show lines ending with ‘user’ in file1
`grep “user$” file1`
- ...or perhaps lines beginning with ‘user’
`grep “^user” file1`
- Search all files in a dir1 for the string ‘user’
`grep -r “user” dir1/`

The Unix Operating System

- Frontier command line environment



```
dir1 — todbadrakh@login07: ~ — ssh hub — 121x43
[todbadrakh@hub2:~]$ ssh frontier
*****
NOTICE TO USERS

This is a Federal computer system and is the property of the United States
Government. It is for authorized use only. Users (authorized or
unauthorized) have no explicit or implicit expectation of privacy.

Any or all uses of this system and all files on this system may be
intercepted, monitored, recorded, copied, audited, inspected, and disclosed
to authorized site, Department of Energy, and law enforcement personnel, as
well as authorized officials of other agencies, both domestic and foreign.
By using this system, the user consents to such interception, monitoring,
recording, copying, auditing, inspection, and disclosure at the discretion
of authorized site or Department of Energy personnel.

Unauthorized or improper use of this system may result in administrative
disciplinary action and civil and criminal penalties. By continuing to use
this system you indicate your awareness of and consent to these terms and
conditions of use. LOG OFF IMMEDIATELY if you do not agree to the
conditions stated in this warning.
*****
([todbadrakh@frontier] Enter PASSCODE:
Last login: Wed Mar 26 13:57:17 2025 from hub2.ccs.ornl.gov

-----
There are messages associated with the following module(s):
-----

Core/24.07:
On March 18, 2025, a new version of the Core module, Core/25.03, was provided on Frontier. Note
that the default Core remains Core/24.07 at this time. Please test this module and report any
issues to help@olcf.ornl.gov. If no issues are discovered, Core/25.03 will become the default
Core on April 1, 2025. For more information see:
https://docs.olcf.ornl.gov/software/software-news.html#frontier-core-module-march-18-2025

DefApps:
Frontier Outage: Frontier will be unavailable from 8:00 AM until 8:00 PM (EST) on Tuesday,
April 1st, 2025.

-----
todbadrakh@login07:~$
```

Login

SSH client

- Open Terminal or Powershell

```
ssh <username>@frontier.olcf.ornl.gov
```

- Enter your password when prompted.

Browser method

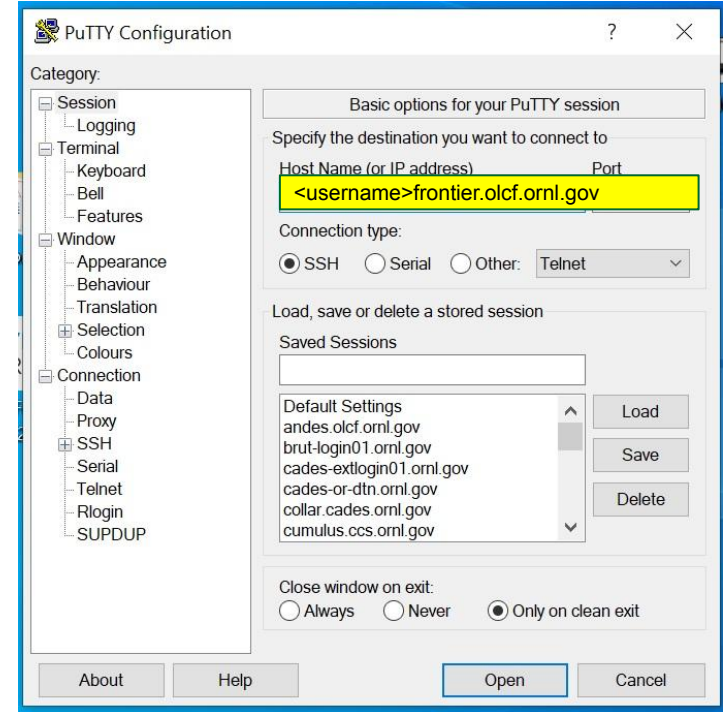
- Open browser
- Go to <https://jupyter.olcf.ornl.gov/>
- Login with your csep#### Username and PIN followed by your Token Passcode
- Choose “HPC and AI Crash Course lab”
- Navigate to the terminal icon and click it.
- Then type

```
ssh <username>@frontier.olcf.ornl.gov
```

- Enter your PIN followed by the token code

Windows Putty

- Click Open Putty
- Enter<username>@frontier.olcf.ornl.gov
- Click Open
- Enter your password when prompted.



The Vim Text Editor

Questions to Answer

- What is Vim?
- How do I use Vim?
- Where can I learn more about Vim?

What is Vim?


- “Vim is a highly configurable text editor built to enable efficient text editing. It is an improved version of the vi editor distributed with most UNIX systems”
- Modal command-line text editor
- Common in HPC environments
 - Edit source code, job submission scripts, config files, & others!
- Vi and Vim are often used synonymously
 - Vim is a superset of Vi

Installing Vim

- Available by default on all OLCF systems
- Easily installable to your own laptop

SPONSOR
Vim development

VOTE
for features

 the editor

BUY
the Vim book

HELP
Ugandan

not logged in ([login](#))

ENHANCED BY [Google](#)

[Home](#)
[Advanced search](#)

[About Vim](#)
[Community](#)
[News](#)
[Sponsoring](#)
[Trivia](#)
[Documentation](#)

Download

[Vim from GitHub](#)
[Vim from Mercurial](#)
[List of Mirrors](#)
[Sources](#)
[Patches](#)
[Development](#)
[Runtime files](#)
[Script links](#)
[Translations](#)
[Old stuff](#)

[Scripts](#)
[Tips](#)

Downloading Vim

Vim is available for many different systems and there are several versions. This page will help you decide what to download.

Most popular:

MS-Windows: Recent and signed MS-Windows files are available on the [vim-win32-installer site](#). The current stable version is [gvim_8.2.2825.exe](#). An alternative is the [standard self-installing executable](#), currently version 8.2.2824.

Unix: See the [GitHub](#) page, or [Mercurial](#), if you prefer that. There is also an [Appimage](#) which is build daily and runs on many Linux systems.

Mac: See the [MacVim](#) project for a GUI version and [Homebrew](#) for a terminal version

Details and options for:

- [Unix](#)
- [PC: MS-DOS and MS-Windows](#)
- [Amiga](#)
- [OS/2](#)
- [Macintosh](#)
- [Others](#)

Where can I learn more about Vim?

- In command line: `$ vimtutor`
- vim-adventures.com
- www.vim.org/docs
- Help reference in Vim: `:`:help``

How do I use Vim?

Let's go to the command line

```
[maccarthy@login05.frontier ~]$ vimtutor
```

```
$git clone https://github.com/olcf/foundational\_hpc\_skills.git
```

```
$cd foundational_hpc_skills/intro_to_vim
```

```
$ vim intro_vim_with_exercises.txt
```

```
=====
Welcome to the VIM Tutor - Version 1.7
=====

Vim is a very powerful editor that has many commands, too many to
explain in a tutor such as this. This tutor is designed to describe
enough of the commands that you will be able to easily use Vim as
an all-purpose editor.

The approximate time required to complete the tutor is 30 minutes,
depending upon how much time is spent with experimentation.

ATTENTION:
The commands in the lessons will modify the text. Make a copy of this
file to practice on (if you started "vimtutor" this is already a copy).

It is important to remember that this tutor is set up to teach by
use. That means that you need to execute the commands to learn them
properly. If you only read the text, you will forget the commands!

Now, make sure that your Caps-Lock key is NOT depressed and press
the j key enough times to move the cursor so that lesson 1.1
completely fills the screen.
=====
Lesson 1.1: MOVING THE CURSOR

** To move the cursor, press the h,j,k,l keys as indicated. **

      ^
    < h   k   l >      Hint: The h key is at the left and moves left.
          |           The l key is at the right and moves right.
          j           The j key looks like a down arrow.
          v

1. Move the cursor around the screen until you are comfortable.

2. Hold down the down key (j) until it repeats.
   Now you know how to move to the next lesson.

3. Using the down key, move to lesson 1.2.

NOTE: If you are ever unsure about something you typed, press <ESC> to place
you in Normal mode. Then retype the command you wanted.

NOTE: The cursor keys should also work. But using hjkl you will be able to
move around much faster, once you get used to it. Really!
=====
Lesson 1.2: EXITING VIM

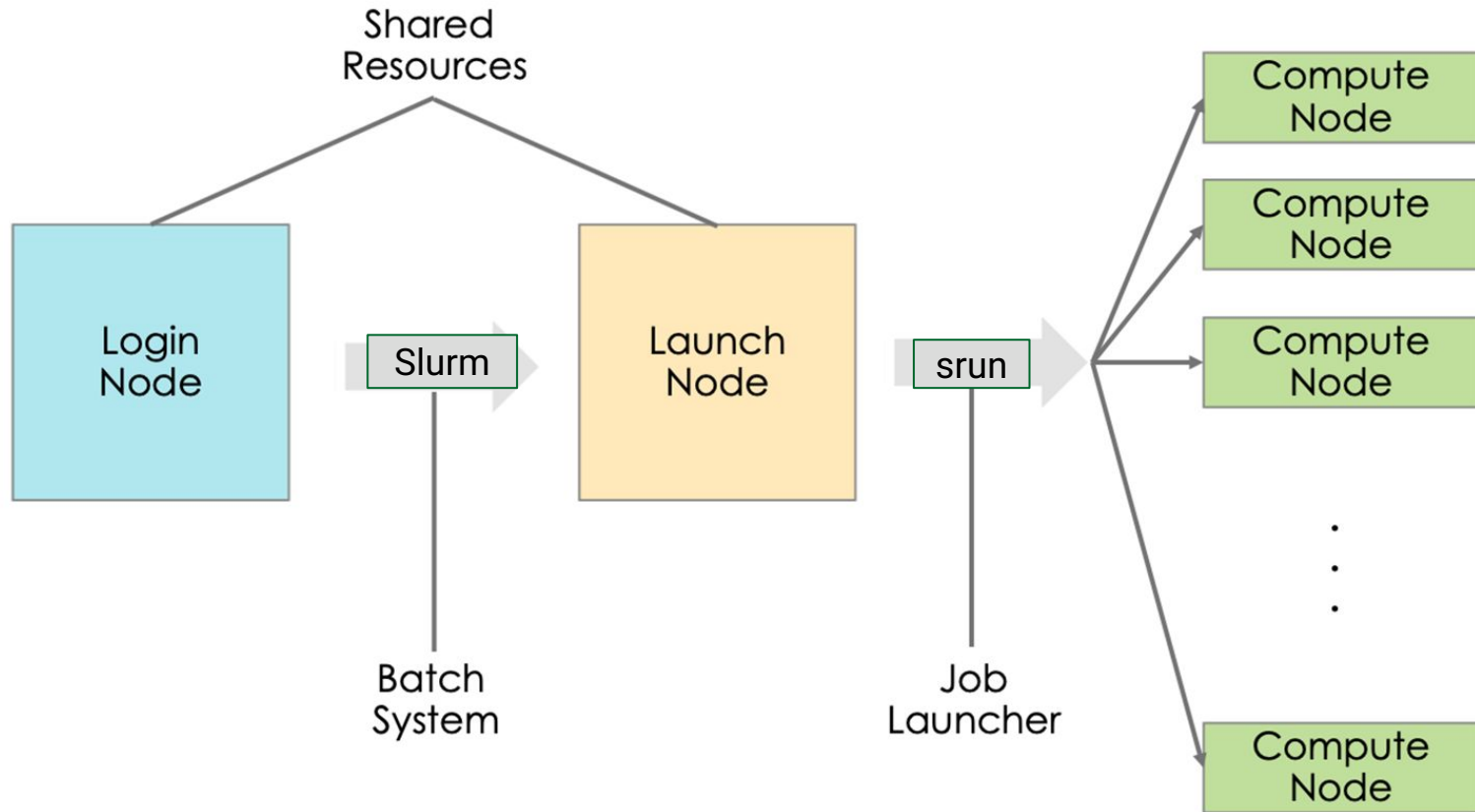
"/tmp/tutorZosa8E" 972 lines, 33583 bytes
```

The Unix Operating System

Resource Schedulers

- There are two node categories on most HPC systems
 - Login node
 - Compute Node
- On most systems, you land directly on a login node once you sign-on.
- Login nodes are shared and not for intensive compute or IO operations.
- Thus, jobs are supposed to be run on compute nodes.
- HPC Resource schedulers are needed to be able to run jobs on compute nodes.
- Schedulers manage and distribute computational resources across a cluster, ensuring fair and efficient access for multiple users.
- There are several types of schedulers
 - Slurm Scheduler
 - PBS/Torque scheduler
- Frontier has a slurm scheduler

Login and Compute Nodes



Frontier Slurm Scheduler

Slurm provides 3 ways of submitting and launching jobs on Frontier compute nodes: **batch scripts, interactive, and single-command.**

NOTE: Jobs launch on the allocated compute nodes, with the first compute node in the allocation serving as head-node.

Slurm commands associated with the 3 methods:

<code>sbatch</code>	Used to submit a batch script to allocate a Slurm job allocation. The script contains options preceded with <code>#SBATCH</code> .
<code>salloc</code>	Used to allocate an interactive Slurm job allocation, where one or more job steps (i.e., <code>srun</code> commands) can then be launched on the allocated resources (i.e., nodes).
<code>srun</code>	Used to run a parallel job (job step) on the resources allocated with <code>sbatch</code> or <code>salloc</code> , or used to create a resource allocation and run a job step in a single command.

Slurm Scheduler

```
1  #!/bin/bash
2  #SBATCH -A ABC123
3  #SBATCH -J RunSim123
4  #SBATCH -o %x-%j.out
5  #SBATCH -t 1:00:00
6  #SBATCH -p batch
7  #SBATCH -N 1024
8
9  cd $MEMBERWORK/abc123/Run.456
10 cp $PROJWORK/abc123/RunData/Input.456 ./Input.456
11 srun ...
12 cp my_output_file $PROJWORK/abc123/RunData/Output.456
```

You submit with **sbatch <script-name>**

Option	Example Usage	Description
-A	#SBATCH -A ABC123	Specifies the project to which the job should be charged
-N	#SBATCH -N 1024	Request 1024 nodes for the job
-t	#SBATCH -t 4:00:00	Request a walltime of 4 hours. Walltime requests can be specified as minutes, hours:minutes, hours:minutes:seconds days-hours, days-hours:minutes, or days-hours:minutes:seconds
--threads-per-core	#SBATCH --threads-per-core=2	Number of active hardware threads per core. Can be 1 or 2 (1 is default) Must be used if using <code>--threads-per-core=2</code> in your <code>srun</code> command.
-d	#SBATCH -d afterok:12345	Specify job dependency (in this example, this job cannot start until job 12345 exits with an exit code of 0. See the Job Dependency section for more information)
-C	#SBATCH -C nvme	Request the burst buffer/NVMe on each node be made available for your job. See the Burst Buffers section for more information on using them.
-J	#SBATCH -J MyJob123	Specify the job name (this will show up in queue listings)
-o	#SBATCH -o jobout.%j	File where job STDOUT will be directed (%j will be replaced with the job ID). If no -e option is specified, job STDERR will be placed in this file, too.
-e	#SBATCH -e joberr.%j	File where job STDERR will be directed (%j will be replaced with the job ID). If no -o option is specified, job STDOUT will be placed in this file, too.
--mail-type	#SBATCH --mail-type=END	Send email for certain job actions. Can be a comma-separated list. Actions include BEGIN, END, FAIL, QUEUE, INVALID_DEPEND, STAGE_OUT, ALL, and more.
--mail-user	#SBATCH --mail-user=user@somewhere.com	Email address to be used for notifications.
--reservation	#SBATCH --reservation=MyReservation.1	Instructs Slurm to run a job on nodes that are part of the specified reservation.

DEMO

1. Srun Job Launcher
2. Basic Unix Vim
3. Basic Workflow
4. Password in a Haystack

What is the Programming Environment?

At the highest level, the PE is your shell's build- and run-time environment.

- Compilers, compiler wrappers, tools, scientific libraries, runtimes, etc.
- See output of running `env`

OLCF offers many software packages for users

Managed by users through session environment variables.

- Search paths
 - `PATH`, `LD_LIBRARY_PATH`, `LIBRARY_PATH`, `PKG_CONFIG_PATH`, ...

Much of the available software cannot coexist simultaneously in your environment.

LMOD Environment Modules

Frontier runs LMOD to manage environment complexity

Build- and runtime-environment software managed with LMOD

- <https://lmod.readthedocs.io>

Usage:

```
$ module -t list           # list loaded modules
$ module avail            # show modules that can be loaded given current env
$ module help <package>  # help info for package (if provided)
$ module load <package> <package>... # add package(s) to environment
$ module unload <package> <package>... # remove package(s) from environment
$ module reset           # restore system defaults
$ module restore <collection> # load a saved collection
$ module spider <package> # deep search for modules
$ module purge          # clear all modules from env
```

LMOD Environment Modules – `module avail`

- The **module avail** command shows *only what can be loaded given currently loaded packages*.
- Full or partial package names limit output to matches

```
$ module avail
...
----- [ Cray Programming Environment ] -----
PrgEnv-amd/8.3.3          cray-mpixlate/1.0.5
PrgEnv-amd/8.4.0          cray-mpixlate/1.0.6
PrgEnv-amd/8.5.0          cray-mrnet/5.0.4
PrgEnv-amd/8.6.0          (D)   cray-mrnet/5.1.0
PrgEnv-cray-amd/8.3.3     cray-mrnet/5.1.1
PrgEnv-cray-amd/8.6.0     (D)   cray-mrnet/5.1.4          (D)
PrgEnv-cray/8.3.3        cray-openshmemx/11.5.7
PrgEnv-cray/8.6.0        (L,D)  cray-openshmemx/11.6.1
...
```

Where:

L: Module is loaded

D: Default Module

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".

LMOD Environment Modules – module spider

- Use **module spider** (instead of **module avail**) to search for modules
 - Finds packages that cannot be loaded given current environment
 - Shows requirements needed to make package available

```
$ module spider hdf5
-----
hdf5:
-----
Versions:
  hdf5/1.12.1-mpi
  hdf5/1.14.3-mpi
  hdf5/1.14.5
Other possible modules matches:
  cray-hdf5  hdf5-vol-cache  hdf5-vol-log  hdf5-vol-async  cray-hdf5-parallel  cray-netcdf-hdf5parallel
-----
For detailed information about a specific module use the module's full name.
For example:

  $ module spider Foo/1.2.3-----
For detailed information about a specific "hdf5" package (including how to load the modules) use the module's full name.
Note that names that have a trailing (E) are extensions provided by other modules.
For example:

  $ module spider hdf5/1.10.4
```

Compiler Environments

CCE
cce/18.0.1 (D)
cce/18.0.0

(OpenMP Offload)
(OpenACC)*
(HIP)*

AMD
amd/6.2.4 (D)
amd/6.3.1

(OpenMP Offload)
(OpenACC)*
(HIP)

GCC
gcc/12.2.0 (D)
gcc/11.2.0

(OpenMP Offload)*
(OpenACC)*

Programming Environments (PrgEnv)

PrgEnv-cray
PrgEnv-cray/8.6.0 (D)
PrgEnv-cray/8.5.0

PrgEnv-amd
PrgEnv-amd/8.6.0 (D)
PrgEnv-amd/8.5.0

PrgEnv-gnu
PrgEnv-gnu/8.6.0 (D)
PrgEnv-gnu/8.5.0

How to get the repo

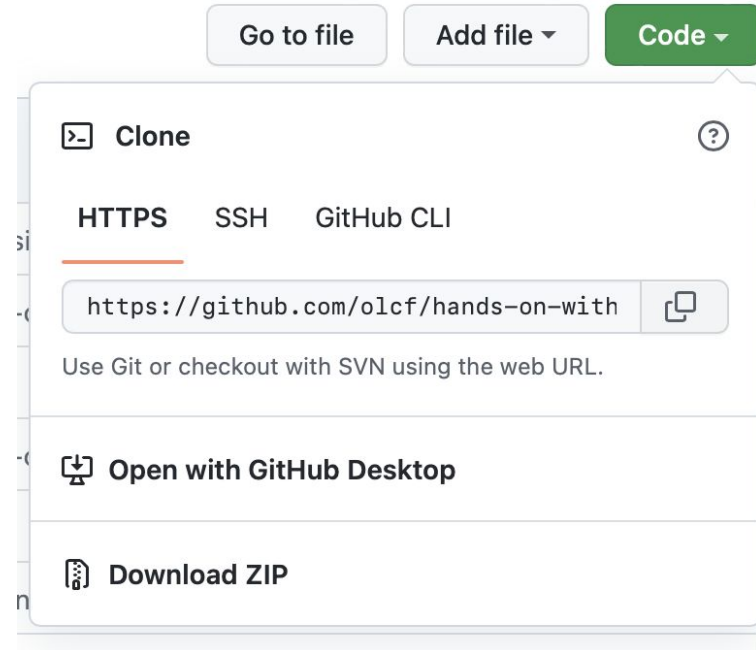
Direct your browser to

<https://github.com/olcf/hands-on-with-frontier>

Click on the  to copy the address.

Go to your command line and type
“git clone”
then paste in the address.

```
$ git clone https://github.com/olcf/hands-on-with-frontier
Cloning into 'hands-on-with-frontier'...
remote: Enumerating objects: 3242, done.
remote: Counting objects: 100% (695/695), done.
remote: Compressing objects: 100% (272/272), done.
```



Join the Course SLACK Workspace

<https://bit.ly/4hyqLHh>



**Want to Work at Oak Ridge
National Laboratory?**

**Fill out our Talent pool form
---->**

Jobs: <https://jobs.ornl.gov>

**Internships:
<https://education.ornl.gov>**

Oak Ridge National Laboratory

