

Other Resources Available

Linux.org's Free Online Courses: <http://www.linux.org>

SUSE Linux Enterprise Server 10 (Fundamentals):

<http://www.novell.com/training/freelearning/course/view.php?id=122>

- Introduction
- Understanding the Linux Story
- Use the Linux Desktop
- Use the KDE Desktop Environment
- Administer Linux with YaST
- Manage Directories and Files
- Locate and Use Help Resources
- Work with the Linux Shell and Edit Text Files
- Manage Users, Groups and Permissions

Norman Matloff's Unix and Linux Tutorial Center (Professor at University of California at Davis):

<http://heather.cs.ucdavis.edu/~matloff/unix.html>

- Introductory to Advanced Topics

Linux Desktop 101: <http://linux.about.com/c/ec/1.htm>

- This course was developed based on the material in the “User Guide to Using the Linux Desktop”, originally published by United Nations Development Programmes, Asia-Pacific Development Information Programme. Includes 14 weekly lessons for users of a modern PC running the Linux operating system. Screen shots are included in each lesson.

Introduction to C Programming Courses

Cornell Virtual Workshop:

- [An Introduction to C Programming:](#) This module is for the beginning programmer who wants to learn the effective use of the C language. If you have never programmed before, you can also use this document to learn the basic concepts of programming; however, you may want to have other references to guide you.

Additional Resources in C Programming

MIT Open Courseware (Daniel Weller and Sharat Chikkeru):

<http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-087-practical-programming-in-c-january-iap-2010/>

- **Practical Programming in C:** A course previously taught at MIT, now freely available. Lecture notes, assignments, and solutions – downloadable as well as available online. This course provides a thorough introduction to the C programming language, the workhorse of the UNIX operating system and the dialect of embedded processors and

micro-controllers. The first two weeks will cover basic syntax and grammar, and expose students to practical programming techniques. The remaining lectures will focus on more advanced concepts, such as dynamic memory allocation, concurrency and synchronization, UNIX signals and process control, library development and usage. Daily programming assignments and weekly laboratory exercises are required.

Computer Science for Everyone (formerly called "Higher Computing for Everyone") (Carl Herold):

<http://www.computerscienceforeveryone.com>

This website is designed for the beginner to provide easy to understand programming lessons and tutorials. This website starts with the "C" programming language, and expands from there.

- Introduction to C, programming concepts and principles:
http://www.computerscienceforeveryone.com/Course_1/
- Writing example programs based on previous course material:
http://www.computerscienceforeveryone.com/Course_2/

C Programming Methodology and Data Structures (Video Lectures):

<http://freevideolectures.com/Course/2519/C-Programming-and-Data-Structures>

- An in-depth course

Programming in C (Suresh Jagannathan, Purdue University):

<http://www.cs.purdue.edu/homes/cs240/>

- Downloadable lectures and videos

Wikiversity-C:

<http://en.wikiversity.org/wiki/C>

Learn C The Hard Way: <http://c.learncodethehardway.org/>

- *Learn C the Hard Way* will fill in the "beginning programmer" gap in the literature on the C Programming language by teaching good modern C programming practices and avoiding habits that lead to buffer overflows, security defects, and other problems that even experienced programmers cause.

Introduction to Fortran Programming Courses

Cornell Virtual Workshop:

- [An Introduction to Fortran Programming](#)
This module is for the beginning programmer who is interested in learning the effective use of the Fortran language. If you have never programmed before you can also use this document to learn the basic concepts of programming; however, you may want to have other references to guide you.

Examples of Additional Resources

Fortran Programming for Engineering and Science Students (Dr. Ugur Guven):
<https://www.udemy.com/introduction-to-fortran-programming-for-engineering-students/>

- Fundamentals of Fortran, Subroutines, Loops
- Sign-up required, but free

About Fortran Tutorial:

<http://www.fortrantutorial.com/index.php>

- This Fortran study guide is a “hands-on” introduction to **programming** using Fortran. The emphasis in this course is to learn how to program, rather than to learn Fortran.

An Interactive Fortran 90 Programming Course (University of Liverpool):

<http://www.liv.ac.uk/HPC/HTMLFrontPageF90.html>

- This HTML/Java course describes the Fortran 90 Programming language and assumes a basic knowledge of programming.

Fortran 90 Tutorial (Dr. C.-K. Shene, Department of Computer Science, Michigan Technological University):

<http://www.cs.mtu.edu/~shene/COURSES/cs201/NOTES/fortran.html>

A Mathworks Link Exchange for the Technical Computing Community:

<http://www.mathtools.net/Fortran/index.html>

Fortran-Related Links Site:

<http://www.stat.wisc.edu/~loh/fortran.html>

Compile and Execute Fortran-95 Online:

http://www.compileonline.com/compile_fortran_online.php

Python for HPC

Cornell Virtual Workshop:

- [An Introduction to Python](#)
Python is a programming language designed with ease of programming and readable code as its foremost goals. Python has risen to prominence in scientific computing as the ideal tool for doing data conversions, scripting parameter studies, and in facilitating the scientific workflow. In this online course, a quick overview of the language is presented, along with a few tricks to maximize the utility of Python for engineering and scientific modeling.
- [Python for High Performance](#)
While Python is a scripting language; it has plenty of facilities for high performance computing. This article covers some of its features and libraries that are particularly helpful when moving scientific code to a large cluster resource. It also includes specific recipes for compilation and execution on the TACC cluster.

Additional Resources for Python

Scientific Python Workshop/PDF Slides (*Pat Miller, Center for Applied Scientific Computing*):

https://computing.llnl.gov/tutorials/python/sci_python.pdf

- Using Scientific Python
- Scientific Python II
- Using Python Numeric
- MPI Parallel Programming In Python

Python Site:

<http://docs.python.org/2/library/debug.html>

- These libraries help the user with Python development: the debugger enables you to step through code, analyze stack frames and set breakpoints, etc., and the profilers run code and give you a detailed breakdown of execution times, allowing you to identify bottlenecks in your programs.

Google: Introductory Python:

<https://developers.google.com/edu/python>

- This free class is for people with a little bit of programming experience who want to learn Python; it includes written materials, lecture videos, and lots of code exercises to practice Python coding. These materials are used within Google to introduce Python to people who have limited programming experience (enough to know what a “variable” or “if statement” is). The first exercises work on basic Python concepts like strings and lists, building up to the later exercises which are complete programs dealing with text files, processes, and http connections.

Parallel Programming Concepts Courses

Cornell Virtual Workshop:

- [Parallel Programming Concepts and High-Performance Computing](#)
This course entails concepts concerning parallel processing and its efficient realization within different hardware and software environments.

Additional Resources for Parallel Programming

An Introduction to Parallel Programming (*Ruud van der Pas, Senior Staff Engineer, Sun Microsystems, Inc.*):

Introduction to Parallel Computing (Blaise Barney, Lawrence Livermore National Library):

https://computing.llnl.gov/tutorials/parallel_comp/

- This tutorial, the first of eight tutorials in the 4+ day “Using LLNL’s Supercomputers” workshop, is intended to provide only a brief overview of the extensive and broad topic of parallel computing as a lead-in for the tutorials that follow it. As such, it covers only the basics of parallel computing, and is intended for someone who is just becoming acquainted

with the subject and is planning to attend one or more of the other tutorials in this workshop.

Udacity: Introduction to Parallel Programming with CUDA (John Owens, University of California, Davis; David Luebke, NVIDIA):

<https://www.udacity.com/course/cs344>

- Video course. Free, but Udacity sign-up required. Learn the fundamentals of parallel computing with the GPU and the CUDA programming environment. In this class, participants learn about parallel programming by coding a series of image processing algorithms. Participants will be able to program and run assignments on high-end GPUs, even if they do not own one.

Coursera (<http://www.coursera.com>)

A resource to use for online search of courses and education on a wide range of topics and levels in many fields.

Provides an introduction to efficient serial and parallel computing using Fortran 90, OpenMP, MPI, and Python; and software development tools such as version control, Makefiles, and debugging:

- High Performance Scientific Computing (University of Washington through Coursera): <https://www.coursera.org/course/scicomp>
A ten-week course for students without a multiprocessor computer. Amazon Web services are available for class projects at a very low cost.
- **Basic Concepts: Programming Languages** (University of Washington through Coursera):
<https://www.coursera.org/course/proglang>
This course teaches basic programming skills in ML, Racket, and Ruby – not languages used in the HPC – so this is a programming theory course, at an intermediate-level.

Message-Passing Interface (MPI) Basics

Cornell Virtual Workshop:

- [Message-Passing Interface \(MPI\)](#)
MPI is a de facto standard specifying the interface and functionality of a message-passing library, a collection of routines for facilitating communication (exchange of data and synchronization) among the tasks in a distributed memory parallel program. MPI is the first standard and portable message-passing library that offers good performance. This module will introduce you to MPI, provide an overview of the functionality and key concepts, and give you a first look at how to use it.

Additional Resources for Message-Passing Interface

Tutorial on MPI: The Message-Passing Interface (William Gropp, Mathematics and Computer Science Division, Argonne National Laboratory):

<http://www.mcs.anl.gov/research/projects/mpi/tutorial/gropp/talk.html>

MPI Point-to-Point Communications

Cornell Virtual Workshop:

- [MPI Point-to-Point Communication](#)

This module details and differentiates the various types of point-to-point communication available in MPI. Point-to-point communication involves transmission of a message between a pair of processes, as opposed to collective communication, which involves a group of processes.

MPI Collective Communications

Cornell Virtual Workshop:

- [MPI Collective Communications](#)

The purpose of collective communication is to manipulate a shared piece or set of information. In this module, you will be introduced to these routines in three categories: synchronization, data movement, and global computation.

Example of Additional Resources

MPITutorial.com: <http://www.mpitutorial.com>

This site is dedicated to providing all the information needed to learn MPI and progress to topics such as parallel I/O and hybrid parallel programming.

- [MPI Broadcast and Collective Communication](#)
<http://www.mpitutorial.com/mpi-broadcast-and-collective-communication/>

MPI Advanced Topics

Cornell Virtual Workshop:

- [MPI Advanced Topics](#)

This module will introduce you to some of the advanced capabilities of MPI beyond ordinary message-passing, including how to customize your environment in the following areas: derived datatypes; groups of processes and their associated communicators; virtual topologies among processes; and parallel I/O using MPI-IO. Application to specific architectures such as Ranger, Lonestar, etc. are discussed.

OpenMP

Cornell Virtual Workshop:

- [OpenMP](#)

In the shared-memory environment that Ranger has on each node, it is much easier to

introduce parallelism into your code with OpenMP than to do pthread programming from scratch or to use MPI. This module introduces OpenMP and describes how to use it.

Additional Resources for OpenMP

OpenMP (Blaise Barney, Lawrence Livermore National Library):

<https://computing.llnl.gov/tutorials/openMP/>

- This tutorial, one of eight tutorials in the 4+ day “Using LLNL's Supercomputers” workshop, is geared to those who are new to parallel programming with OpenMP. Basic understanding of parallel programming in C or Fortran is required. For those who are unfamiliar with Parallel Programming in general, the material covered in [EC3500: Introduction to Parallel Computing](#) would be helpful.

SC08 OpenMP Tutorial (Tim Mattson and Larry Meadows, both of Intel):

<http://openmp.org/wp/2008/11/sc08-openmp-hands-on-tutorial-available/>

- Day-long tutorial with downloadable slides and code exercises.

MATLAB Programming

Cornell Virtual Workshop:

- [MATLAB Programming](#)
MATLAB provides matrix manipulation, plotting, and general-purpose scientific programming capability, as well as functionality through specialized “toolboxes” such as the Optimization toolbox, the Statistics toolbox, the Signal Processing toolbox, the Image Processing Toolbox, etc.

Profiling and Debugging

Cornell Virtual Workshop:

- [Profiling and Debugging](#)
This module describes how to obtain detailed performance data for jobs on Ranger. It also discusses tools and techniques for online parallel application debugging.

Other Useful URLs

Compile Online:

<http://www.compileonline.com>

- Compile and execute many web technologies and programming languages online for simple codes. (*No need to install compiler to learn*)