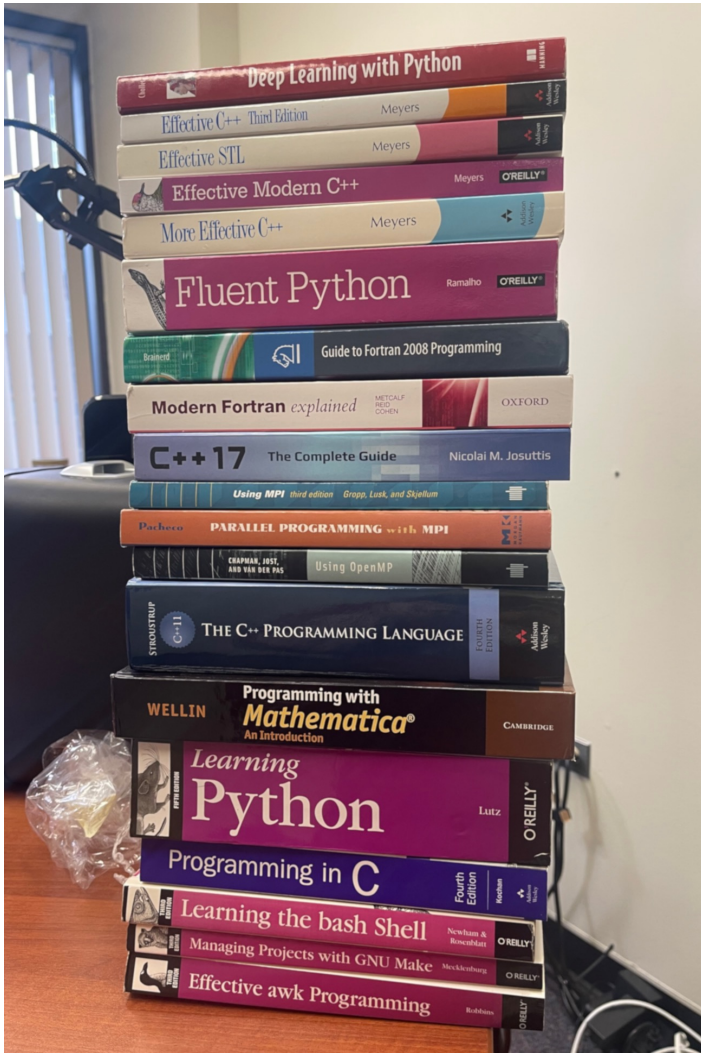
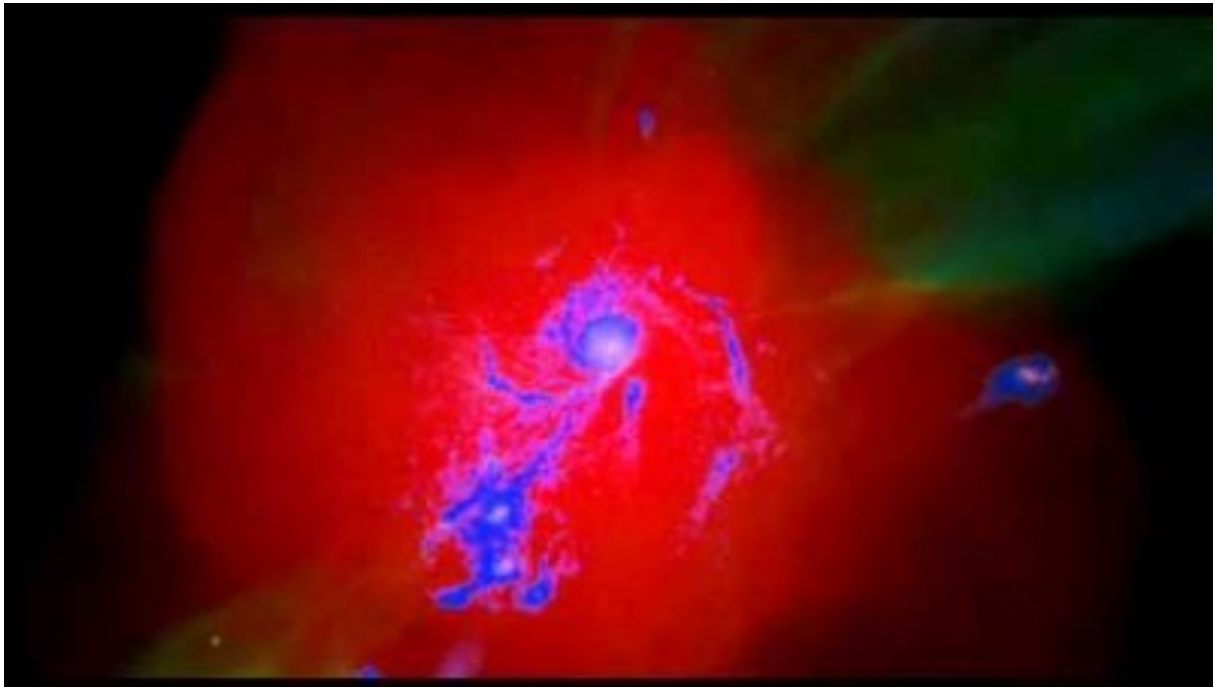
# PHY 504

## Computational Physics (~Int to Programming)

## Vivian Miranda – YITP (Assistant Professor)



Cosmological Simulation, Baryonic Evolution (UW Astro)

# PHY 504 Learning Goals

- Learn to use a bash language for Computational Science Tasks
  - File systems / bash / sed / / scp / ssh / sftp
- Learn to write programs in **C code** (2011 standard)
  - Get the job done. Pretty fast. **!!!! Pointers!!!!!**
- Learn to write programs in **basic modern** C++ 101 features
  - Widely adopted in experimental physics (ex: CERN ROOT)
  - Used: control experimental apparatus & data acquisition
  - Doesn't have nD array capabilities natively (need Armadillo lib)
- Learn how to write parallel OpenMP/MPI code

# PHY 504 What is the initial Plan?

- Lecture 1: Hello!
- Lecture 1-7: Bash/Shell
- Lecture 7-17: C with numerical examples
- Lecture 18: How to use the GSL Numerical Library
- Lecture 19-20: Basics of OpenMP – share memory parallelization
- Lecture 20-28: Basics of C++ including Object-Oriented Paradigm
- Lecture 29: How to create a Makefile.
- Lecture 30:  GIT version control.

# PHY 504 Learning Goals

- Success: complete a few exercises before the next lecture
- Bring a laptop to see the slides better (the screen is too far)
- Learning a language will include a LOT of reading material
- There are a lot of supplementary books in this course
- Study the remaining slides we didn't cover in class is HW
- Questions: **Can our class go 15 min extra?**
- Common Courtesy is required.
    - Cellphones / conversation / earbuds – not allowed

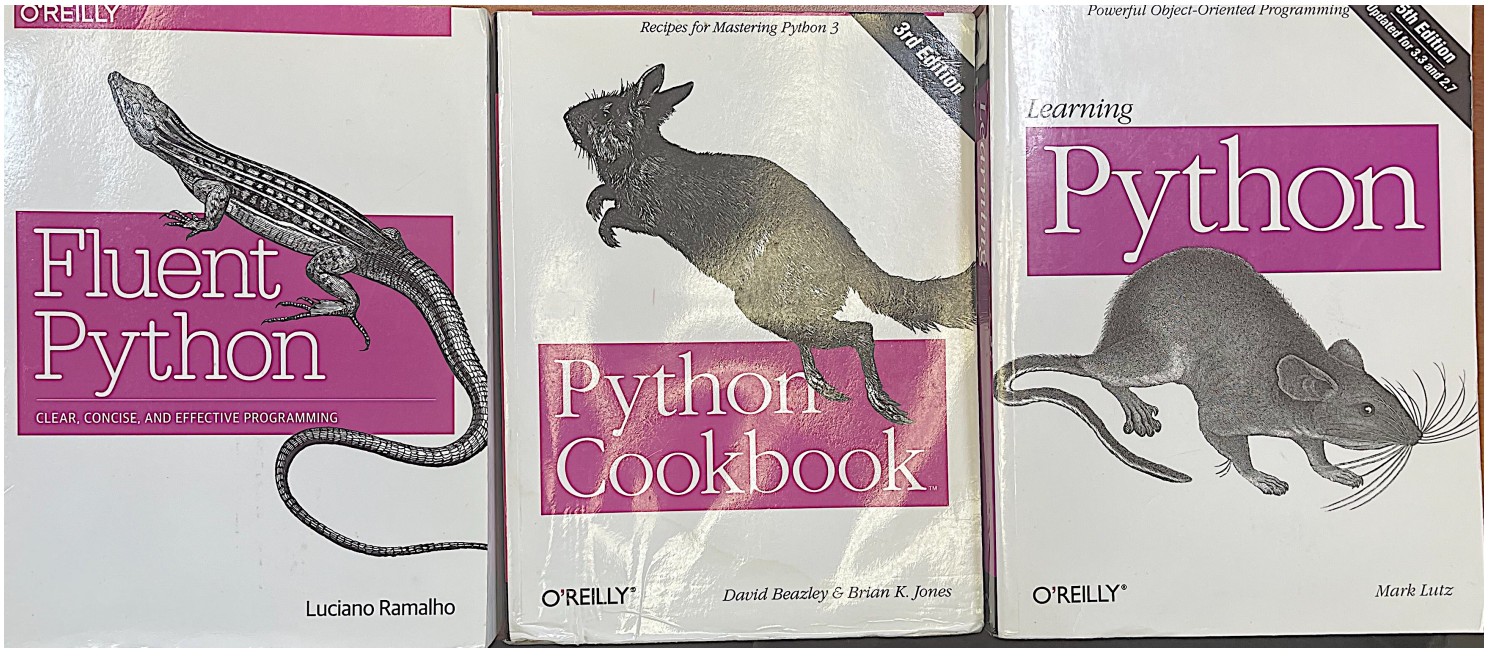
# PHY 504 Learning Goals

- Class time: M/W 11:00 am - 12:**30** pm (need the extra 15min)
- We don't have a TA. But I am happy to help you during office hours. Proposed Office Hours: M/W 4:30-6:00 pm.
- Send me an e-mail if you need additional help: vivian.miranda@stonybrook.edu
- You are a grad student (Master's and Ph.Ds). This course aims to help you succeed both in academia and in industry (if you decide to make this transition).

# PHY 504 Learning Goals

## We won't teach Python in this class.

- **Why? Are Profs. too old to learn a new language? No!**
  - Understanding low-level languages is vital in scientific computing
  - Many modules in Python are written in C++/C/Fortran
  - Simple to learn it "alone". Can provide general guidance if asked




Suggested Bibliography for self-studying Python

# Books: you will need them!

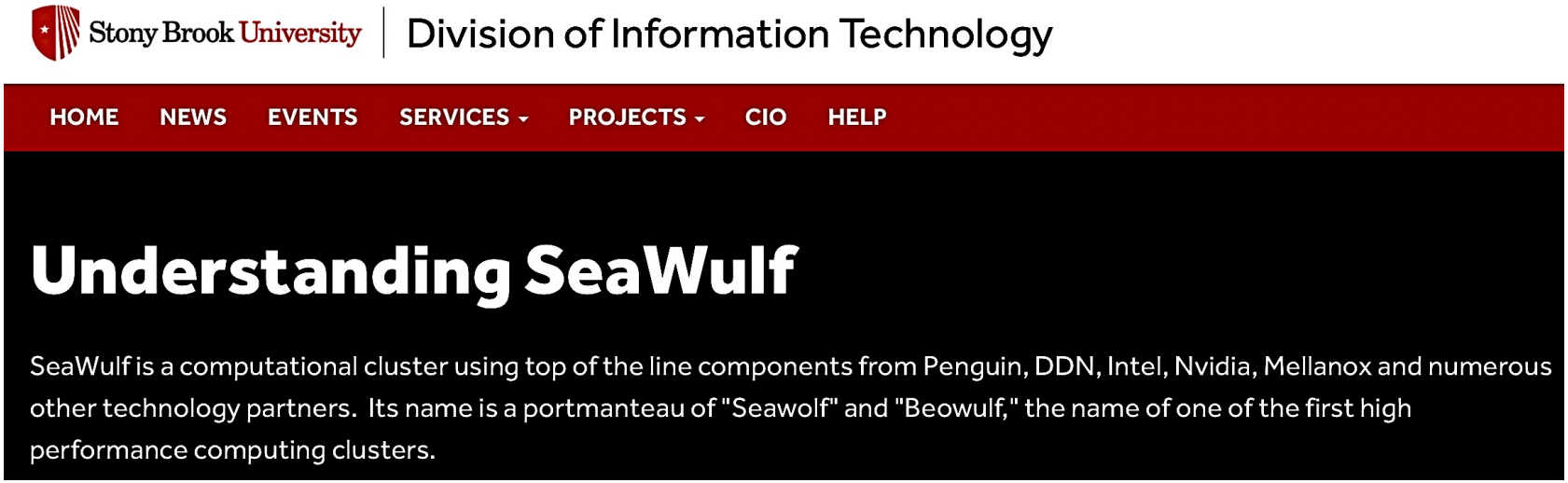
- I am going to use **MANY** books as material for classes.
- Why books? Sound knowledge of how the language works.
- Books can help you learn features better than AI/SO
- You don't need to buy/read them.
- However, I do suggest reading the introductory chapters.
- For 129 dollars per quarter – you can read O'Reilly books.

  **legally** at **https://learning.oreilly.com/home/**

  - Example: C++ from Bjarne Stroustrup by Addison-Wesley Professional

# Why learning C/C++?

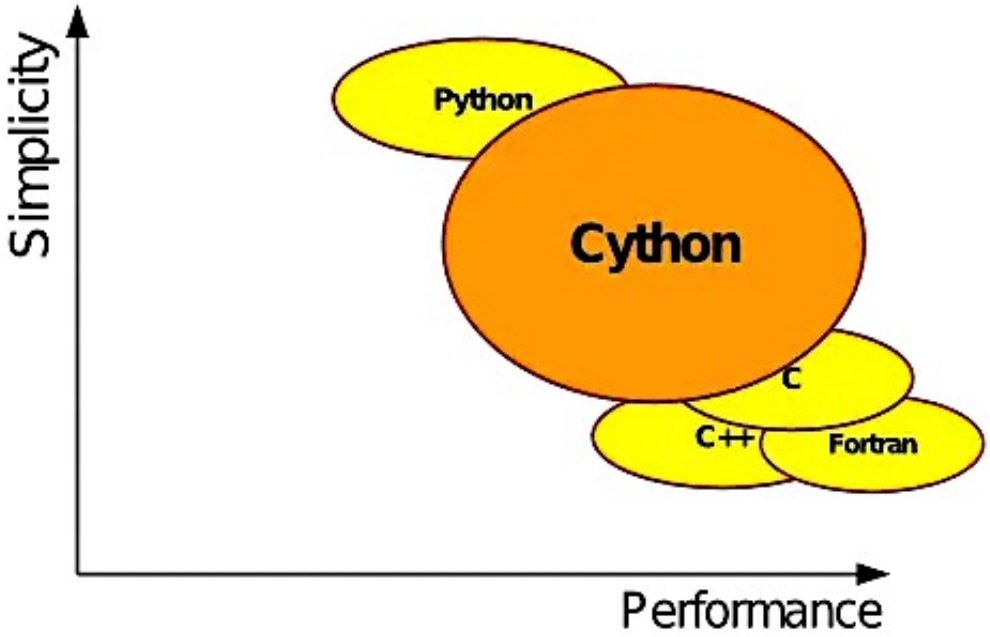
- Even if your research pipelines are Python-based, there are a lot of possible integrations with compiler C/C++.
- There is a lot of research code in compiler languages.
- Understand C/C++ = "free your mind" to think about physics.
- C/C++ forces you to think about RAM. RAM is EXPENSIVE!




New nodes have 96 cores but only 256GB of RAM

$70,000 (I asked them!) to upgrade them to 512GB of RAM!

# Modules in Python written in C




Credit: https://medium.com/prismapp/cython-c-extensions-for-python-33acbd53c5e8

**Python Code**

```
import example
result = example.do_something()
```

Magic of C pointers

**C Code**

```
static PyObject * example(PyObject *self)
{
    // do something
    return Py_BuildValue("i", result);
}
```

Credit: https://stackoverflow.com/a/18764177/2472169



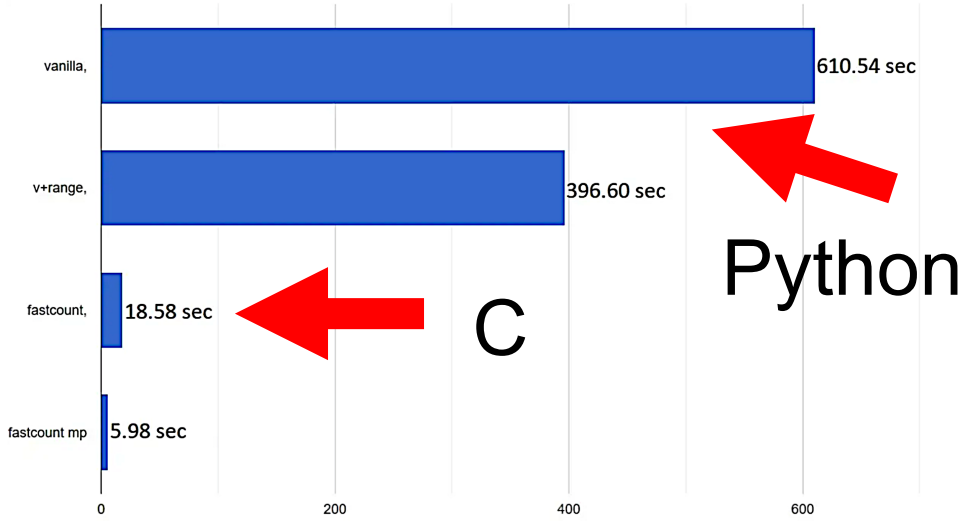
Mike Huls

Dec 8, 2021 · 9 min read · Member-only · Listen

## Write Your Own C-extension to Speed Up Python by 100x

How to write, compile, package and import your own, superfast C-module into Python

Credit: https://towardsdatascience.com/write-your-own-c-extension-to-speed-up-python-x100-626bb9d166e7



# Modules in Python written in C++




**pybind11 — Seamless operability between C++11 and Python**






Guilherme Fickel

May 13, 2022 · 5 min read · Listen

**100x speedup on Python with just a touch of C++**

Credit: https://medium.com/@guilhermefickel/100x-speedup-on-python-with-just-a-touch-of-c-f83136545023

```cpp
#include <pybind11/pybind11.h>

int add(int i, int j) {
    return i + j;
}

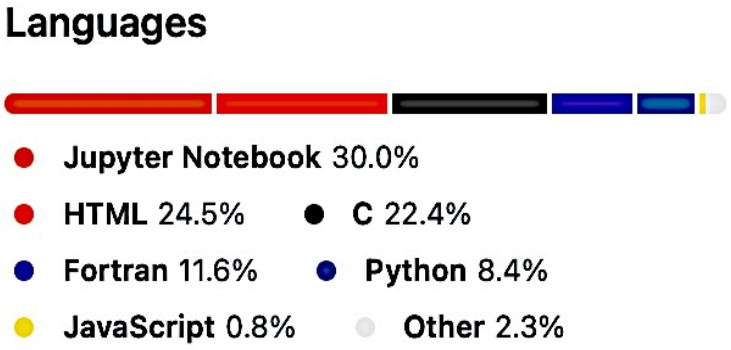
PYBIND11_MODULE(example, m) {
    m.doc() = "pybind11 example plugin"; // optional module docstring

    m.def("add", &add, "A function that adds two numbers");
}
```

# Python vs C & C++

- **Python is slow, so does it make a bad language?**
  - Absolutely NOT!
  - Within reason, **Human time** is more critical than **Computer time**
  - Computers are expensive. RAM is expensive!
    - SBU supercomputer has limited RAM per core
  - Research pushes computers to their limits.

The distribution of languages in the code I use in my research




Credit: https://github.com/CosmoLike/cocoa

Legacy code is written in many languages.

Don't reinvent the wheel.

Rewriting code destroyed companies.

# Don’t reinvent the wheel

- **Rewriting code has destroyed entire companies.**

  Be open to learning different languages! Learn them well!

**Stay away from useless internet wars on computer languages.**

When you join a research group/company – there are legacy codes that limit your choices.

Credit: https://huseyinpolatyuruk.com/lessons-learned-from-rewriting-code-in-my-10-years-as-a-developer/

## Lessons learned from rewriting code in my 10+ years as a developer

04 FEBRUARY 2019

**12 months.**

That’s the time we wasted for rewriting our software from scratch.

12 long months in the software market.

Without innovation...

Without moving forward...

Really, I cannot stop asking this question to myself.

*What could we achieve in this fast-moving world in 12 months?*

*“Tuesday, January 20, 2015, 5:10 PM, AntiMalware product is finally going its first public beta”*

# LTA / First Challenge: StackOverflow

Challenge (show me at the beginning / end of the semester): StackOverflow



Edit profile | Profiles

## Vivian Miranda

Member for 12 years, 7 months | Last seen this week | Visited 1801 days, 1 consecutive

stonybrook.edu/commcms/... | New York, NY

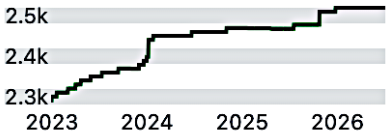
Profile | Activity | Saves | Settings

- Summary
- Answers
- Questions
- Tags
- Articles
- Badges
- Following
- Bounties

### Summary

**REPUTATION**

2,527




Top tag: c++

Next privilege: Cast close & reopen votes — 3,000 Rep.

**BADGES**

1 | 20 | 29

Newest: Yearling

Next badge: Excavator — 0/1

**IMPACT**

~268k people reached

0 posts edited

4 helpful flags

317 votes cast

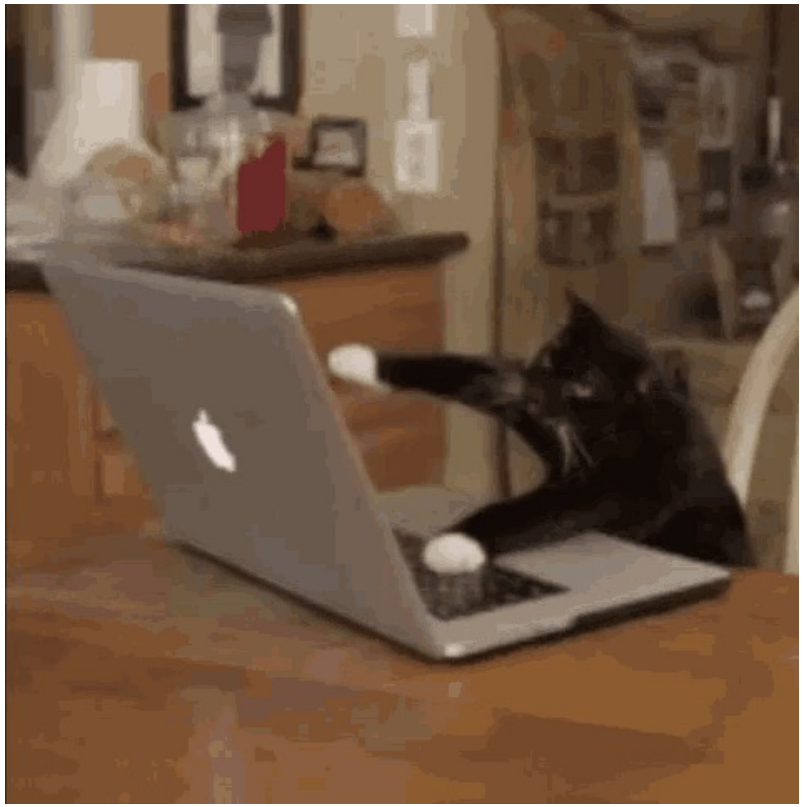
When learning C/C++ and GSL Library, I created a StackOverflow account and tried to answer some questions.

Do the same (harder w/ AI)! There are a lot of unanswered questions

# Lab Machines vs Laptop/workstation

We have machines in this library that help to avoid the nightmare of library installations

But, to translate what you will learn here for research, you need to install compile/libs on your machines.



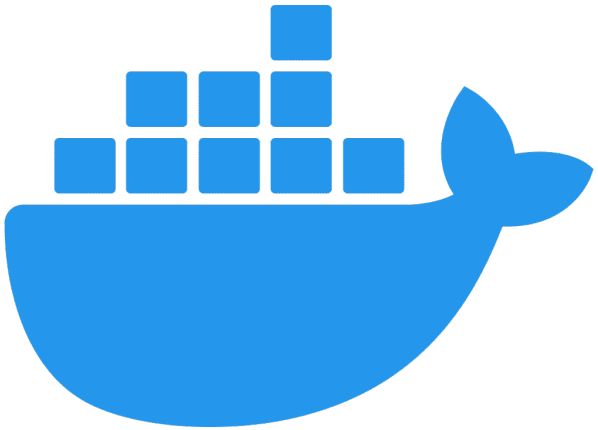
**<u>This can be a nightmare.</u>**

There is a way I can help you: **DOCKER**

# Libraries / Docker

- Setting up libraries/compiler versions can be a nightmare
  - Libraries are a big issue in all languages – including Python.
  - Official policy: use computers in the lab.
  - However, I understand if you want to use your laptop.
    - Relevant if you want to apply what you learn here in research immediately

# Unofficial workaround: Docker



docker®

- Works on Mac (Intel/M1) / Win / Linux
- 'Smart' Virtual Machine: containers
- Created a ~5GB container for this class
- Container code can teach you about Linux
- **You must first install the docker-engine**

# Libraries / Docker

## Created a few GBs Docker container for this class

**https://hub.docker.com/repository/docker/vivianmiranda/whovian-phy504lite**

vivianmiranda / Repositories / whovian-phy504lite / General

General Tags Builds Collaborators Webhooks Settings

vivianmiranda/whovian-phy504lite

Last pushed 20 days ago

Image with compiler & libraries needed to run code and HW on my PHY504 classes.

*Add a category* INCOMPLETE

**Docker commands**

Public view

To push a new tag to this repository:

```
docker push vivianmiranda/whovian-phy504lite:tagname
```

**Tags**

DOCKER SCOUT INACTIVE
Activate

This repository contains 1 tag(s).

| Tag | OS | Vulnerabilities | Health | Pulled | Pushe |
|---|---|---|---|---|---|
| latest | | Security unknown | E | an hour ago | 20 days ag |

**Scout health score** BETA

Last pushed tag: latest

**Checked for:**

High-profile vulnerabilities found




Source code: THIS IS QUITE ADVANCED (here is just for record)!!!!

**https://github.com/vivianmiranda/docker/blob/master/phy504lite/Dockerfile**

# Libraries / Docker

## How do you Docker if you adopt this workaround?

- 1st time: Open the terminal & go to the folder that the VM will have access to (**restricting this folder is crucial**!)

```
docker run --platform linux/amd64 --hostname phy504 --name phy504_2026 -it -p 8888:8888 -v "$(pwd)":/home/whovian/host/ -v ~/.ssh:/home/whovian/.ssh:ro vivianmiranda/whovian-phy504lite
```

```
RiverSong:2026_PHY504_Spring vivianmiranda$ docker run --platform linux/amd64 --hostname w
hovianphy504 --name phy504_2026 -it -p 8888:8888 -v "$(pwd)":/home/whovian/host/ -v ~/.ssh
:/home/whovian/.ssh:ro vivianmiranda/whovian-phy504lite
Unable to find image 'vivianmiranda/whovian-phy504lite:latest' locally
latest: Pulling from vivianmiranda/whovian-phy504lite
f784408d7713: Already exists
fa890f5db61d: Pulling fs layer
2dbb3cd36122: Pulling fs layer
4af53af69398: Pulling fs layer
66230e7eec23: Pulling fs layer
```

# Libraries / Docker

## How do you Docker if you adopt this workaround?

```
********************************************************************************
*                                                                              *
* Welcome to Docker Whovian-PHY504 docker image. I hope this VM                *
* will help you learn shell, C and C++. Have fun!                              *
*                                                                              *
********************************************************************************


You are running this container as user with ID 1005 and group 1005,
which should map to the ID and group for your user on the Docker host. Gre
at!

whovian@whovianphy504:~$
whovian@whovianphy504:~$ gcc --version
gcc (Ubuntu 15.2.0-11ubuntu1) 15.2.0
Copyright (C) 2025 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
.

whovian@whovianphy504:~$ g++ --version
g++ (Ubuntu 15.2.0-11ubuntu1) 15.2.0
Copyright (C) 2025 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
```

Host files will be seen inside the VM at **/home/whovian/host/**

**DON'T GIVE VM ACCESS TO ALL YOUR FOLDERS**

Why? We are going to learn powerful and dangerous bash commands.

# Libraries / Docker

## How do you Docker if you adopt this workaround?

```
**************************************************************************
*                                                                        *
* Welcome to Docker Whovian-PHY504 docker image. I hope this VM          *
* will help you learn shell, C and C++. Have fun!                        *
*                                                                        *
**************************************************************************


You are running this container as user with ID 1005 and group 1005,
which should map to the ID and group for your user on the Docker host. Gr
eat!

whovian@whovianphy504:~$ exit
exit
RiverSong:2026_PHY504_Spring vivianmiranda$ docker start -ai phy504_2026


**************************************************************************
*                                                                        *
* Welcome to Docker Whovian-PHY504 docker image. I hope this VM          *
* will help you learn shell, C and C++. Have fun!                        *
*                                                                        *
**************************************************************************


You are running this container as user with ID 1005 and group 1005,
which should map to the ID and group for your user on the Docker host. Gr
eat!
```

- After downloading the **image** *whovian-phy504lite* and creating the **container** named *phy504_2026*, how do you stop/start containers?

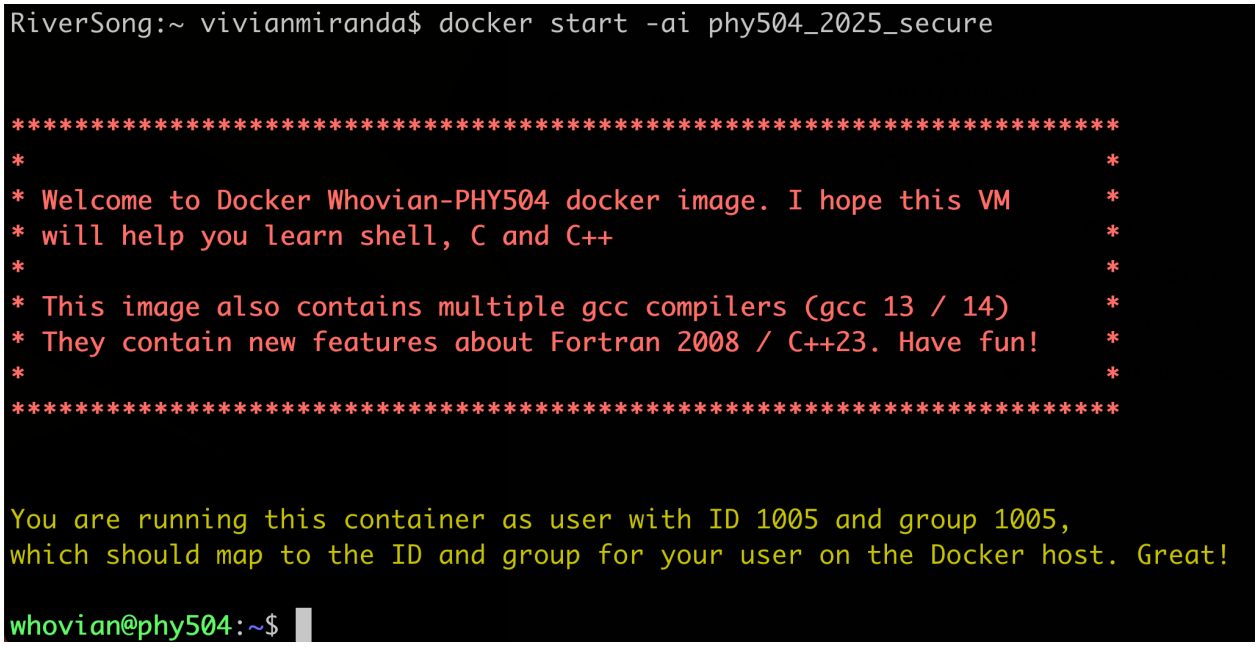
docker start -ai phy504_2026

# Libraries / Docker

If you want absolute security that your mistakes won't affect your host files, use the following command instead. Then the VM has not access to the host.

```
docker run --platform linux/amd64 --hostname phy504 --name phy504_2026_secure -it vivianmiranda/whovian-phy504lite
```

To start this Docker, run `docker start -ai phy504_2026_secure`



# Libraries / Docker

```
FROM ubuntu:24.10

# Fix DL4006
SHELL ["/bin/bash", "-o", "pipefail", "-c"]

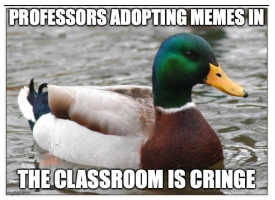
ENV DEBIAN_FRONTEND noninteractive
RUN apt-get update \
  && apt-get install -yq --fix-missing software-properties-common apt-utils build-essential \
  && apt-get update \
  && apt-get update \
  && apt-get install -yq --fix-missing --no-install-recommends \
    git autoconf dpkg git-lfs xz-utils pkg-config libgflags-dev \
    libhdf5-dev libarpack2-dev nano make wget curl libleveldb-dev \
    sqlite3 libv4l-dev fonts-liberation tzdata unzip grep libsnappy-dev \
    sed libsqlite3-dev sudo locales fonts-dejavu libgoogle-glog-dev  \
    libx264-dev libtool automake zlib1g-dev libcurl4-gnutls-dev g++ gfortran \
    emacs sed vim man-db manpages-posix cmake cm-super ghostscript dvipng \
# ------------------------------------------------------------------------
  && ln -s /bin/tar /bin/gtar \
  && echo "en_US.UTF-8 UTF-8" > /etc/locale.gen \
  && locale-gen \
# ------------------------------------------------------------------------
  && curl -L "https://github.com/krallin/tini/releases/download/v0.19.0/tini_0.19.0.deb" > tini.deb \
  && dpkg -i tini.deb \
  && rm tini.deb \
# ------------------------------------------------------------------------
# Install OpenSSH for MPI to communicate between containers
  && apt-get update \
  && apt-get install -y --no-install-recommends openssh-client openssh-server \
  && mkdir -p /var/run/sshd \
# ------------------------------------------------------------------------
# Allow OpenSSH to talk to containers without asking for confirmation
  && cat /etc/ssh/ssh_config | grep -v StrictHostKeyChecking > /etc/ssh/ssh_config.new \
  && echo "    StrictHostKeyChecking no" >> /etc/ssh/ssh_config.new \
  && mv /etc/ssh/ssh_config.new /etc/ssh/ssh_config \
# ------------------------------------------------------------------------
  && apt-get clean \
  && apt-get autoremove --yes \
  && rm -rf /var/lib/apt/lists/* \
  && rm -rf /usr/share/doc/ \
  && rm -rf /usr/share/man/

ARG NB_USER="whovian"
ARG NB_UID=1005
```

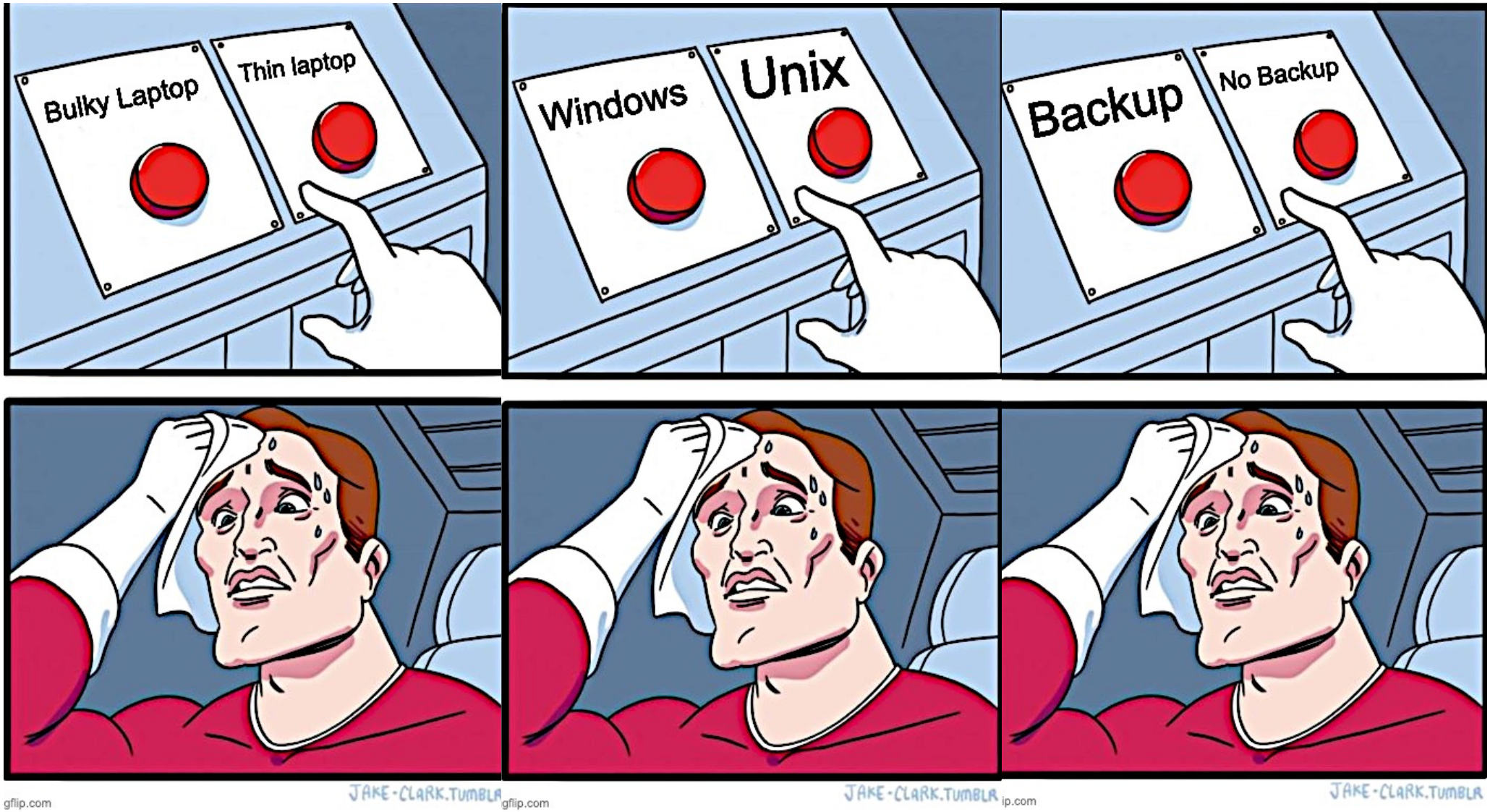
## How to create a Docker Image?

- Docker can be programmed in **Bash**.
- Docker introduces new commands: **RUN**, **ARG**, **SHELL**, **FROM**
- It exemplifies the usefulness of being proficient in a shell terminal

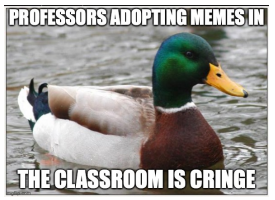
# Long-Term Advice




Few things I tell my students when they start working with computational physics (LTA for life, not for this course necessarily)



# LTA: laptop choices

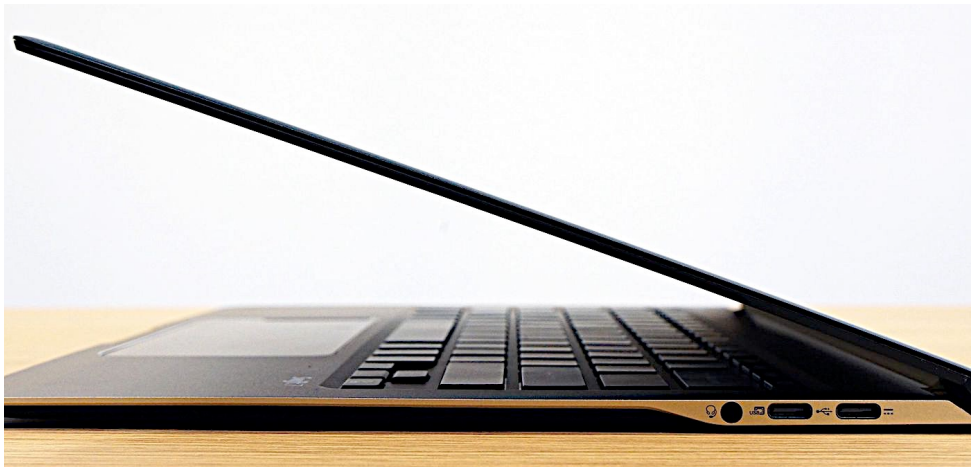



- Laptops are meant for presentations/writing.
  - **H**igh-**P**erformance **L**aptops are $$$. Bad cooling! TERRIBLE IDEA
  - HPLs are heavy – terrible battery life. Slim is better (**NO** Chromebook)
  - A good battery is a life-saver when you are traveling (e.g., conferences)











# LTA: invest in a workstation/server

- Laptops are meant for presentations/writing.
- Build a desktop to perform calculations that are not so large for HPC.
- Buy used parts. Create a small intranet to connect laptop/desktop.

Today, I have an expensive server, but I started cheap as a Ph.D. student



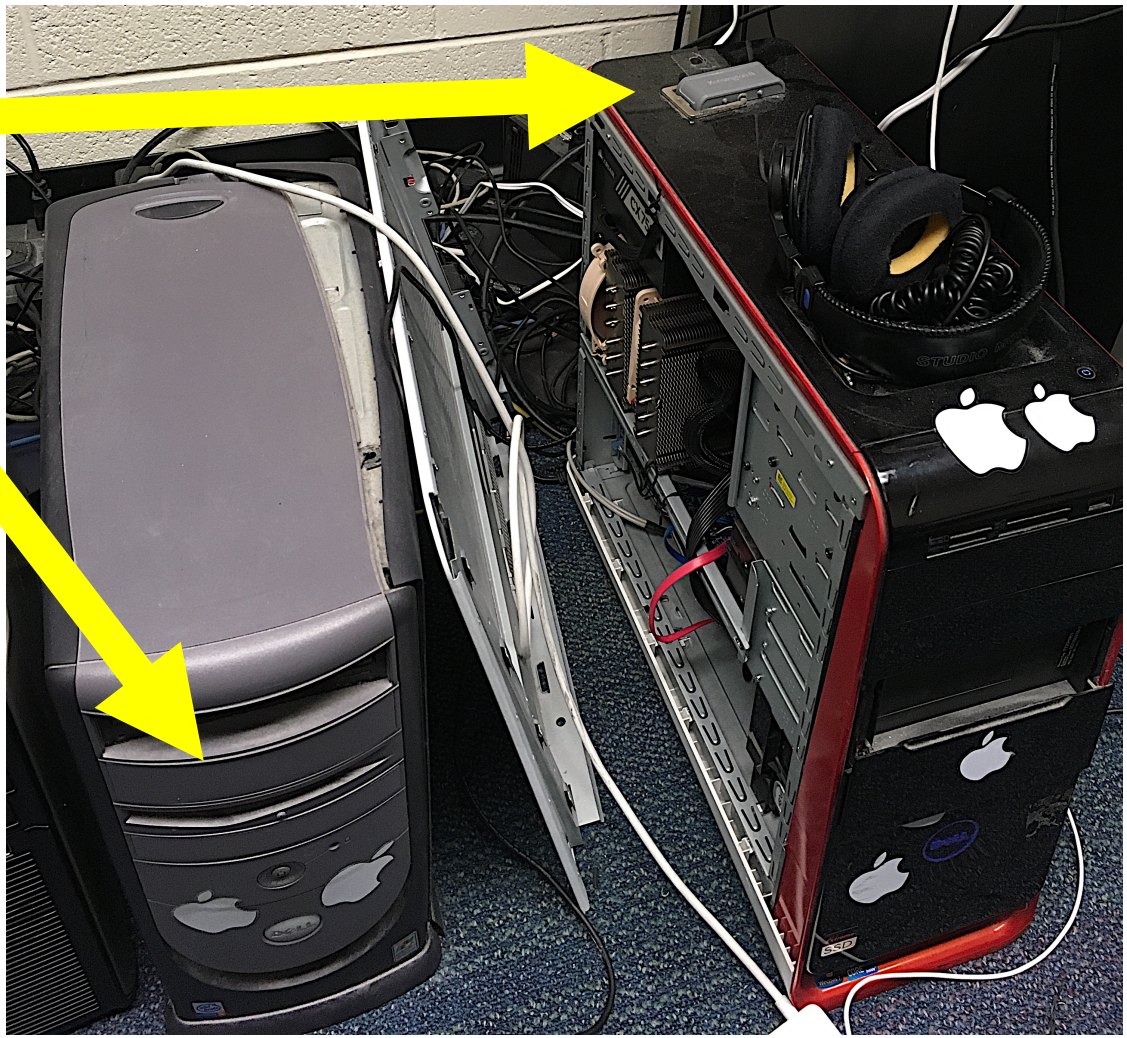
# LTA: invest in a workstation/server

- Laptops are meant for presentations/writing.
- Build a desktop to perform calculations that are not so large for HPC.
- Buy used parts. Create a small intranet to connect laptop/desktop.

When I was a PhD student.

I found it in the trash and created an email server (worked for 3 years).

We will learn to use **ssh** / **sftp** tools to integrate servers and laptops!



# LTA: Operational System

- The best OS is the one you are most productive on. That said
  - Unix systems have native support for Bash
  - Laptops meant for presentations/writing. Linux has issues (drivers)
  - Windows/OSX: expensive tools that may be free on Linux
  - Windows: use WSL (Windows Subsystem for Linux) or docker.
  - Linux is amazing for **servers. Long-term: laptop/server duality**



# LTA: Operational System

- Which flavor of Linux? Stay away from useless internet wars!
  - My two cents: Ubuntu 20.04 LTS is easier / simpler for beginners (my server OS)
  - If you choose Ubuntu – only install LTS (Long-Term Service) versions
  - You still have to install & configure libraries. Docker is awesome for that!



# LTA: Backup

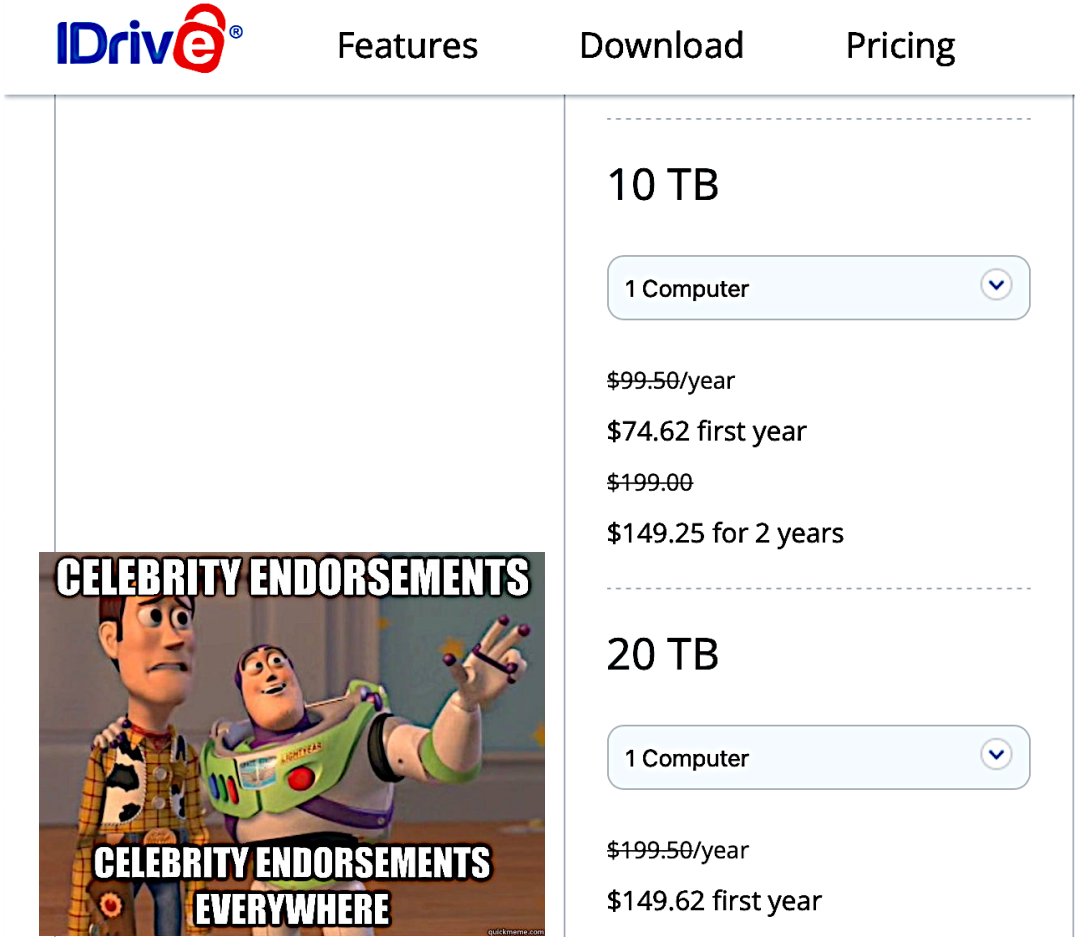
- You need both offline and online backups of your work
  - Dropbox: NOT backup. **$$$ part of Dropbox is synchronization**
  - Online companies offer backup of a limited number of machines
  - Be careful if you work with sensitive data (e.g., health science)

Example: I use IDrive.

U$74.62/year for 10TB

(only one computer)

**Synchronization is not needed in online backups**



# LTA: Read medium.com

- Awesome articles about many languages (and machine learning)
  - Subscription is USS 5/month (like Wikipedia – not written by reporters)



Ibtissam Makdoun

Jun 3, 2022 · 4 min read · Member-only · Listen

## Prepare Data for Machine Learning

Split Data to Train/ Validation/ Test

After cleaning data to prepare it for Machine Learning. In this article we will split the data into three samples. Training, Validation and Test.






Dino Cajic

Feb 24, 2020 · 21 min read · Member-only · Listen

## A High-Level Introduction to the C Programming Language

# We need to address AI!

## AI can be an extraordinary tool to help experienced developers

VM How do I print a format integer with six digits in C

To print an integer with six digits (padded with leading zeros) in C, you can use the printf function with the format specifier %06d. Here's a simple example:

c Cop

```c
#include <stdio.h>

int main() {
    int number = 123;
    printf("%06d\n", number);  // Will print: 000123

    number = 7;
    printf("%06d\n", number);  // Will print: 000007

    number = 123456;
    printf("%06d\n", number);  // Will print: 123456

    return 0;
}
```

The format specifier breaks down as:

- % starts the format specifier
- 0 tells printf to pad with zeros instead of spaces
- 6 specifies the minimum field width
- d indicates it's a decimal integer

You can also use other numbers instead of 6 to get different widths of padding.

Claude (https://claude.ai)

In C, to print an integer with six digits (zero-padded if necessary), you can use the `%06d` format specifier in the `printf` function. Here's an example:

c Copy Edit

```c
#include <stdio.h>

int main() {
    int number = 42;

    // Print the number with six digits, zero-padded
    printf("%06d\n", number);

    return 0;
}
```

**Explanation:**

- `%d` : Specifies that you're printing an integer.
- `06` : Indicates that the width of the field should be 6, and if the number has fewer digits, it will be padded with leading zeros.

**Output:**

Copy Edit

```
000042
```

If the integer has more than six digits, it will be printed as-is without truncation. For example, if `number = 1234567` , the output will simply be `1234567` .

ChatGPT (https://chatgpt.com)

# We need to address AI!

## AI can be an extraordinary tool to help experienced developers

VM How do I print a format integer with six digits in C

To print an integer with six digits (padded with leading zeros) in C, you can use the printf function with the format specifier %06d. Here's a simple example:

c Cop

```c
#include <stdio.h>

int main() {
    int number = 123;
    printf("%06d\n", number);  // Will print: 000123

    number = 7;
    printf("%06d\n", number);  // Will print: 000007

    number = 123456;
    printf("%06d\n", number);  // Will print: 123456

    return 0;
}
```

The format specifier breaks down as:

- % starts the format specifier
- 0 tells printf to pad with zeros instead of spaces
- 6 specifies the minimum field width
- d indicates it's a decimal integer

You can also use other numbers instead of 6 to get different widths of padding.

In C, to print an integer with six digits (zero-padded if necessary), you can use the `%06d` format specifier in the `printf` function. Here's an example:

c Copy Edit

```c
#include <stdio.h>

int main() {
    int number = 42;

    // Print the number with six digits, zero-padded
    printf("%06d\n", number);

    return 0;
}
```

**Explanation:**

- `%d` : Specifies that you're printing an integer.
- `06` : Indicates that the width of the field should be 6, and if the number has fewer digits, it will be padded with leading zeros.

**Output:**

Copy Edit

```
000042
```

If the integer has more than six digits, it will be printed as-is without truncation. For example, if `number = 1234567`, the output will simply be `1234567`.

## However, AI does not imply we don’t need to learn languages anymore. How will we proceed?

# AI compromise. “Penn & Paper”

In the Era of Claude and ChatGPT, using HW as part of the final grade makes no sense. This is my compromise.

1. We will provide plenty of HW (lots of questions). You are allowed to use AI to help you understand HW.
2. HW won’t count toward the final grade.
3. There will be three “pen-and-paper” exams. You will be required to code **without using a compiler**. (You can **replace** paper w/ Emacs/Vim/Nano)
4. Questions will be based on HW.
5. Allowed: handwrite a generic 1/2-page note per language
6. Generic = no HW question. Just generic language features.

# AI compromise. “Penn & Paper”




“Pen-and-paper”?! OMG! Vivian, you are mean! Hear me out: I learned at the UChicago from a great professor that we need to be our own compilers.

At UC, all C++ exams were pen-and-paper, and there was no AI in 2012!

Tools help, but we can’t rely on them to get our work done.

If you don’t want to write on paper, use Emacs/Vim/Nano. **But the file extensions must be “.txt” (no colors)**

# Citations

I had a lot of advice from former instructors of this course (including access to their notes). I hope their notes will help this course to be more useful to you all



Prof. Michael Zingale



Prof. Alan Calder




Prof. Douglas Swesty

All praises to them, all blame on me

# Why Fortran? Connection w/ Python

## Many Python modules are written in Fortran! Again, Speed!

## The F2PY Project

Credit: https://www.f2py.com

Credit: https://www.numfys.net/howto/F2PY/

The F2PY project is created to unify the efforts of supporting easy connection between Fortran and Python languages. The project will provide software, documentation, and support to solve a difficult task of automatically generating Python wrappers to Fortran libraries. It will also coordinate the further development of the F2PY tool and related software. Our ultimate aim is to have a tool that can automatically generate Python wrappers to any Fortran program, be it written in Fortran 66/77 or in Fortran 90/95/2003.

The development of the F2PY Project is carried out under the Google Code F2PY project.

```
f2py -c filename.f95 -m mercury
```

The extension module is then called `mercury` and can be imported as follows.

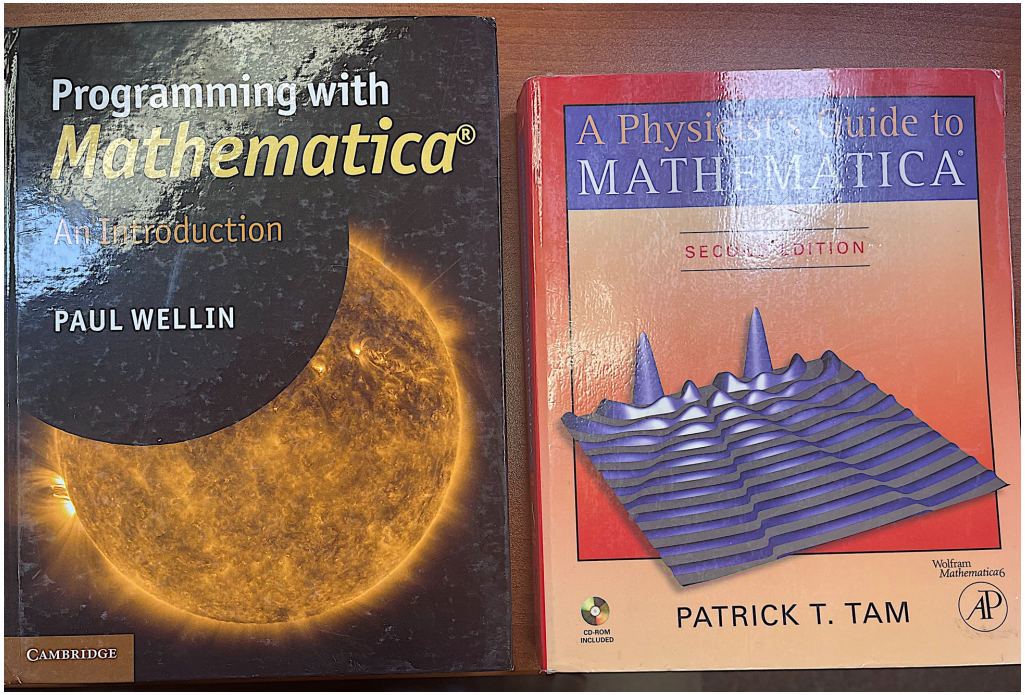
```
import mercury
print(mercury.__doc__)
```

```
This module 'mercury' is auto-generated with f2py (version:2).
Functions:
  x,y,steps = getorbit(z0,h,a,b)
```

Credit:https://nbviewer.org/urls/www.numfys.net/media/notebooks/mercury.ipynb

# PS: Theoretical Physicists

- **Wolfram Mathematica**: language in theoretical physics
  - High-level, no need for a deep understanding of resources
  - Incredible symbolic capabilities
  - Pretty good in numerical calculations
  - Amazing Documentation




Suggested Bibliography for studying Wolfram Language