You can access these slides on the course Github:
https://github.com/natrask/ENM1050

Welcome to

ENGR 1050
Intro to Scientific Computation

Prof. Nat Trask
Mechanical Engineering & Applied Mechanics
University of Pennsylvania

What is this class about?

Scientific Computation

Solving a technical problem using a computer

Our objectives:

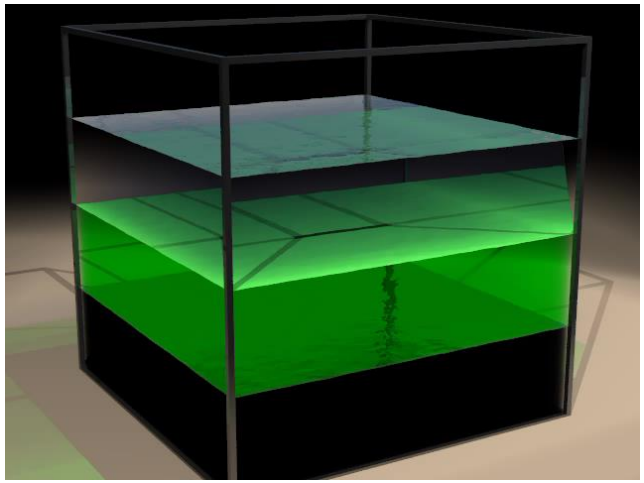
- Translate an English description of a technical problem into a computational model
- Choose a numerical method for solving that problem
- Programming basics to code and debug your approach
- Visualize data and interpret results

Why learn scientific computation?

Computers are used to **simulate** a wide range of systems, mechanical, chemical, economic to gather information that would difficult, dangerous, expensive or impossible to collect otherwise.

Can we predict how atomic weapons work without building them?

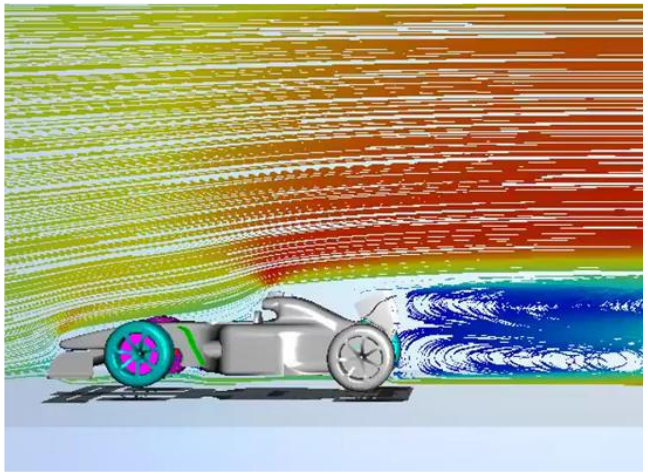
How will a surgical tool interact with the body?



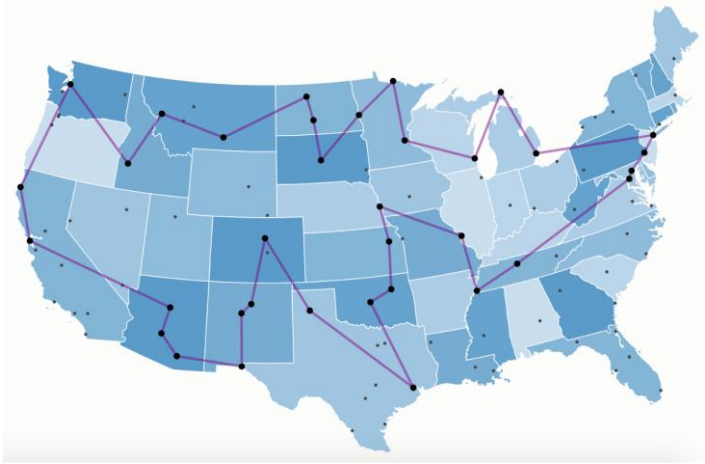
What happens when you put a lighter fluid under a heavier fluid?

Why learn scientific computation?

Computers are used to **optimize** designs or **evaluate** decisions when there are many possible options.



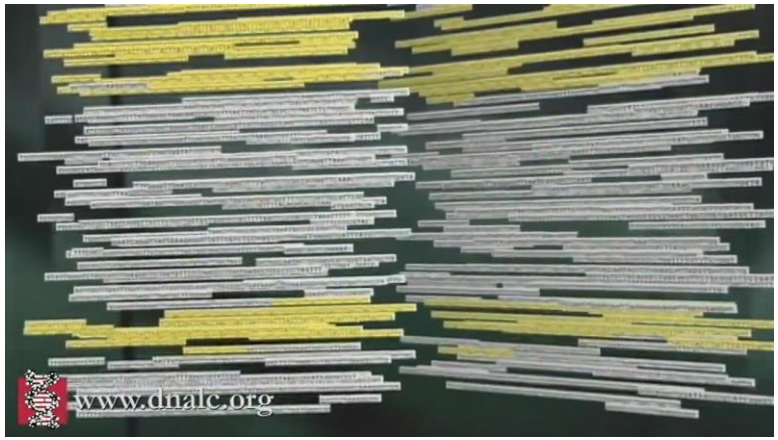
What is the car body that minimizes drag?



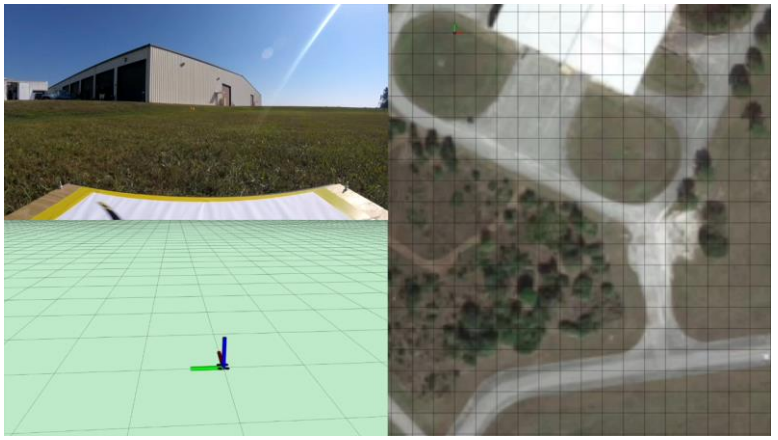
What is the minimum length route through cities?

Why learn scientific computation?

Computers are used to sift through and **analyze large data** sets to find trends and verify hypotheses.



Can we sequence the human genome?



Can we figure out where a robot is from a camera feed?

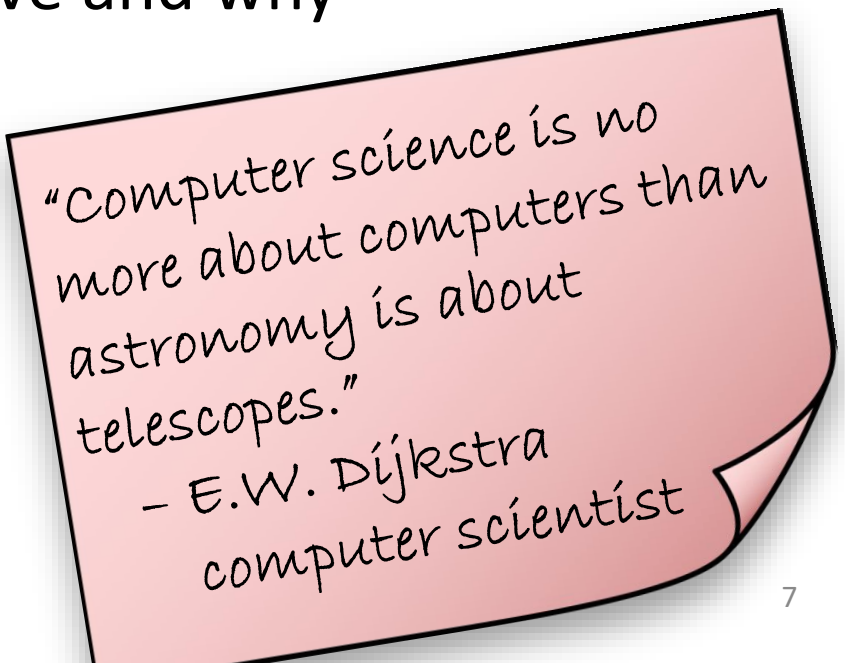
Let's zoom in on the "computation" part

Name some things that …

- Use computers in your day-to-day activities
- Would be really hard to do without a computer
- Are very easy for you to do but very hard for a computer to do

Q: What can be computed?

- Problem solving, using computer as a tool
- Finding general solutions to problems
- Finding efficient solutions to problems
- Thinking about what kinds of problems are hard or slow for computers to solve and why



Syllabus and Schedule posted on Canvas

ENGR 1050: Introduction to Scientific Computation
Fall 2023

(last updated: August 30, 2023)

Description

ENGR 1050 provides an introduction to computation and data analysis using Python – an industry standard programming environment. The course covers the fundamental of computing including variables, control structures, and functions. These concepts are illustrated through examples and assignments that show how computing is applied to various scientific and engineering problems. Examples are drawn from the simulation of physical and chemical systems, the analysis of experimental data, the simulation of dynamic systems, and control of sensors and actuators.

Course Objectives

By the end of this course, you should be able to:

- Translate an English simulation or design problem into a computational model
- Choose a numerical method for analyzing or simulating that engineering system
- Code and debug your chosen computational approach
- Produce a visualization for interpreting the results

Prerequisites

The course does not assume any prior programming experience but will make use of basic concepts from calculus and engineering. Relevant mathematical and engineering principles will be communicated in detail as needed. If you have doubts about your preparedness for the course, please visit the professor's OH.

Teaching Staff

Instructor:	Cynthia Sung Assistant Professor MEAM	crsung@seas.upenn.edu Office hours: M 3:15pm-4:15pm, R 12pm-1pm
Teaching Assistants:	Bibit Bianchini Jeremy Wang Niall Hosein Michelle Lin Tobia Ruth	Alice Li Josh Leshinskie Alexander Qi Alfredo Vazquez

Office Hours

The professor and TAs will hold office hours every week. We will announce these hours during the first week of class and also post them on our Google calendar and Ed Discussion. Please check the calendar for any updates throughout the semester.

ENGR 1050: Introduction to Scientific Computation
Course Schedule (Fall 2023)

(last updated: August 30, 2023)

Week	Dates	Topics	Assignments
1	8/28-9/3	Course introduction and mechanics, python intro, variables and variable types, basic math	
2	9/4-9/10	Plotting, lists/arrays	HW 1
3	9/11-9/17	Decision trees, conditionals, Boolean expressions, pseudocode	HW 2
4	9/18-9/24	Filtering, for loops, range, list comprehension	HW 3
5	9/25-10/1	Search, while loops, break, continue	HW 4
6	10/2-10/8	Modularizing code with functions, variable scope, interactive plots	HW 5
7	10/9-10/15	Numpy arrays and math	HW 6
8	10/16-10/22	OpenCV, image processing, masks, edge detection	HW 7
9	10/23-10/29	Video processing and tracking	Quiz 10/25
10	10/30-11/5	Scipy, Numerical integration, solve_ivp	HW 8
11	11/6-11/12	Polynomial fits	HW 9
12	11/13-11/19	Animations	HW 10 Final project proposal due
13	11/20-11/26	Putting it together	HW 11
14	11/27-12/3	Python scripts	HW 12
15	12/4-12/10	Comparison with MATLAB, MATLAB basics	
Finals	12/11-12/21	Final projects	Final projects

Course Mechanics

Weekly assignments on computing

1 in-class quiz during the semester (10/25)

1 open-ended final project

We'll discuss ~mid-semester

Who am I?

Dr. Nat Trask

Associate Professor

Mechanical Engineering & Applied Mechanics

Secondaries in AMCS, Materials

You can call me: Nat, Prof Trask, Dr Trask

Who am I?

I joined Penn faculty in August 2023

I was senior staff at Sandia National Laboratories 2016-2023, and hold a joint faculty appointment still

I earned my Ph.D. in Applied Mathematics at Brown University in 2015, and hold dual a dual Bachelor degree in Mechanical Engineering and Applied Mathematics from University of Massachusetts.

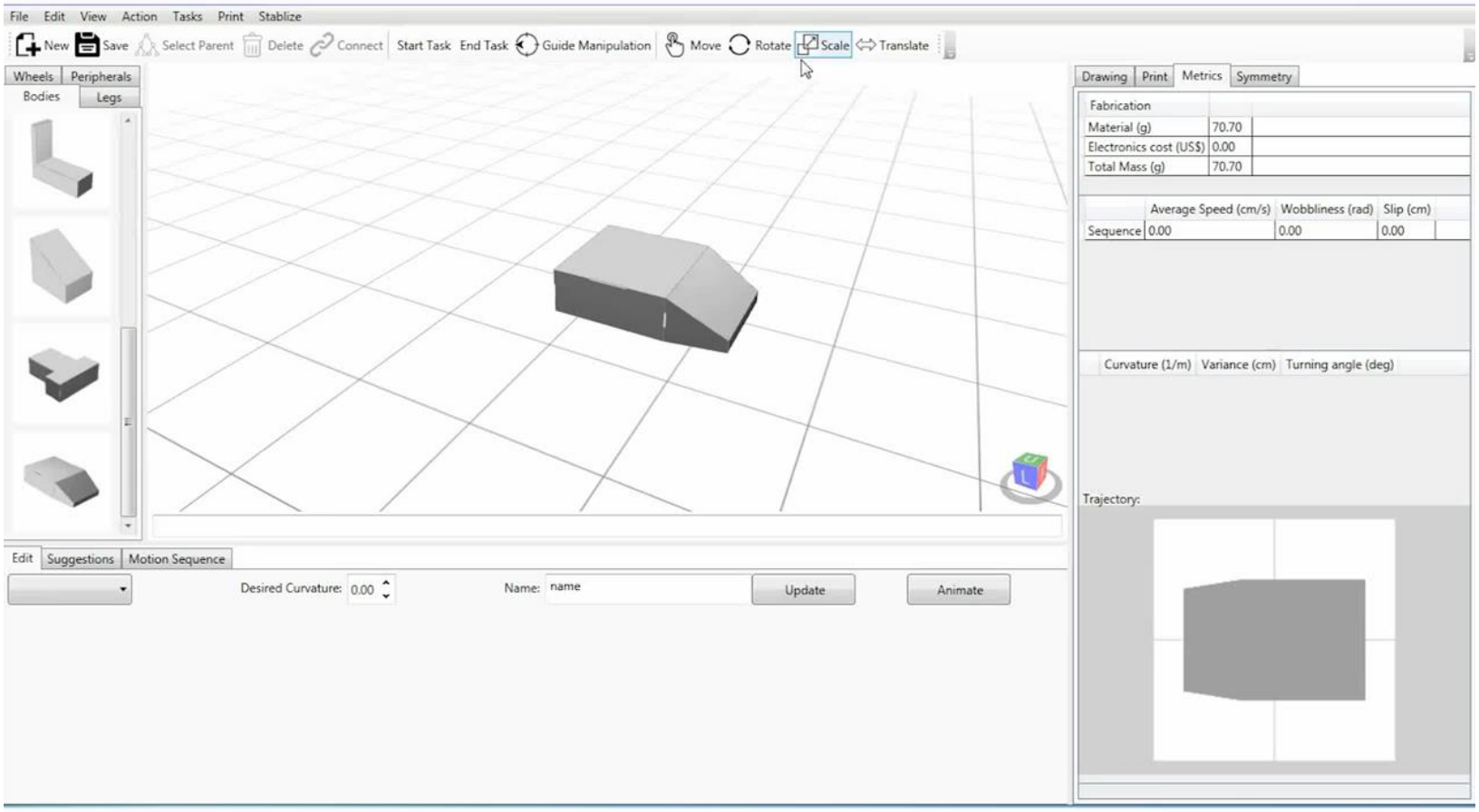
I've run a consulting simulation business since 2010.

My primary research interests are:

- Simulating problems that are too complicated to derive models for with pencil and paper
  - Developing scientific machine learning theory and tools for high-consequence engineering applications
  - Developing advanced simulation tools for multiphysics
- Developing AI for autonomous scientific discovery

Who am I?

Computation is a core components of my research.



Interactive Robogami, 2017

How do you contact me?

Mainly: In class, in office hours, via Ed

Email: crsung@seas.upenn.edu

Please email me only for very private class matters or for non-class-related topics

Office Hours: R 12pm-1pm

This week. Will post full OH schedule with TAs later today

Our Teaching Assistants

Bibit Bianchini
MEAM

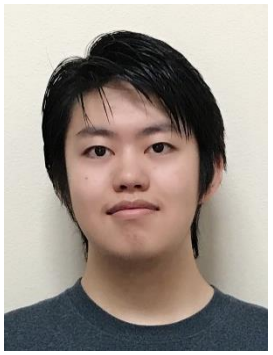
Alice Li
ESE

Jeremy Wang
GRASP

Niall Hosein
Biophysics

Josh Leshinskie
MEAM

Michelle Lin
MEAM



Alex Qi
Neuroscience

Tobia Ruth
VIPER

Alfredo Vasquez
BIOE

Class Format

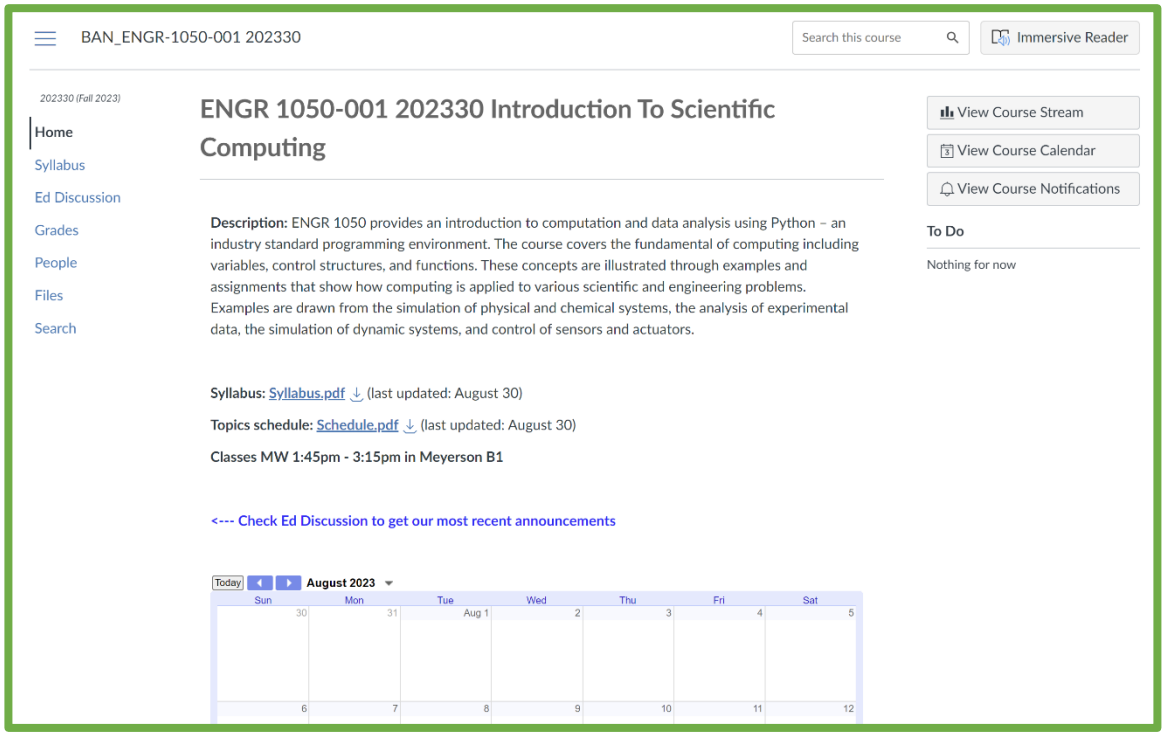
- This course covers scientific computation
- But it is also the first time many of you have programmed before

The most important thing is **PRACTICE**

- Lectures will start with a short presentation of the day's material, but the focus will be practice exercises
- Weekly homework assignments for you to practice coding longer programs on your own

How will we communicate?

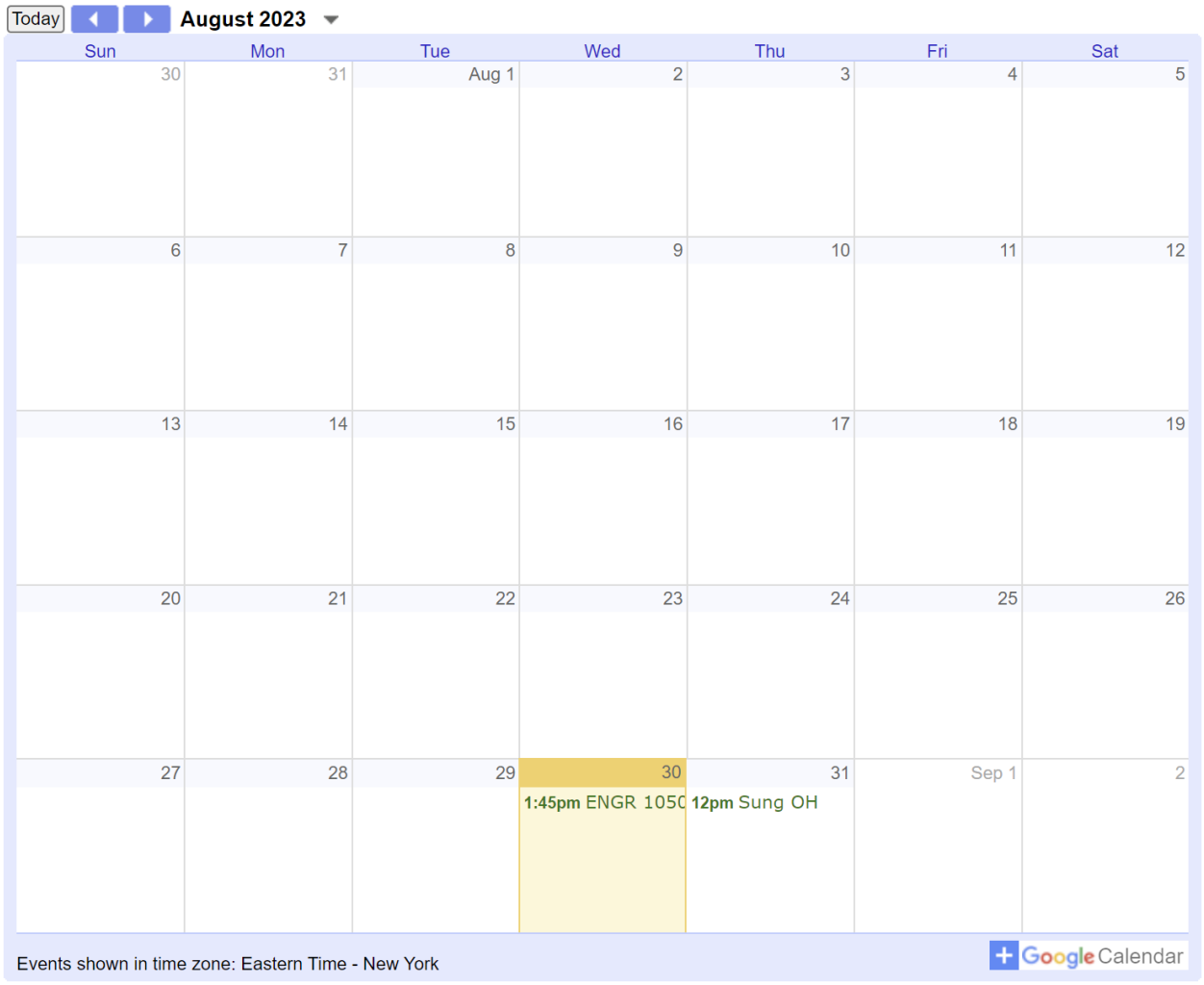
Canvas: https://canvas.upenn.edu



- Assignments
- Grades
- Lecture notes
- Course schedule

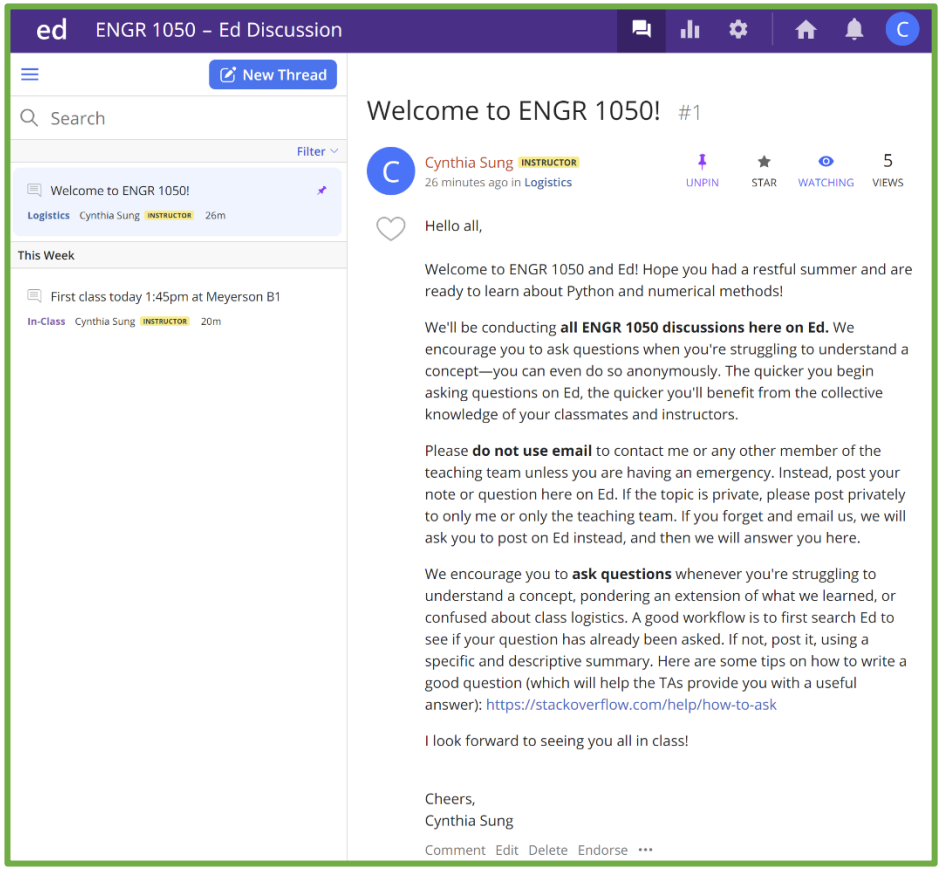
How will we communicate?

Google Calendar – office hours



How will we communicate?

Ed Discussion



- Questions and comments
- Post anonymously (hidden name) if you don’t want other students to identify you
- Post privately (only to instructors) if you are asking about something personal

Collaboration Policy

I encourage you to work together.

Any work you submit should be your own work.

Acknowledge your collaborators.

Ask yourself how the collaboration enhanced your learning. If you have trouble answering this question, you are probably engaging in the wrong type of collaboration.

AI Collaboration Policy

I encourage you to use the tools you have available to you to learn the material.

You should understand any work that you submit.

Acknowledge your collaborators.

Realize that computers have their own limitations and use these tools thoughtfully.

What is a computer program?

A **computer program** is a **sequence of instructions** telling the computer how to perform a computation

A program:

1. Takes input from a user or file
2. Processes it
3. Produces output

Most importantly, **programming** is about **rules**

Computers do **exactly** what we tell them to do

What is a programming language?

A **programming language** is a **formal language** to express instructions.

Natural languages

- Syntax less crucial
- Ambiguous
- Redundant
- Idioms and metaphors

Programming languages

- Strict syntax
- No ambiguity
- Concise
- Literal

Writing a program

1. Identify problem
2. Make solution strategy
3. Break up solution into subtasks
4. Write down sequence of operations using syntax of the programming language
5. Test

Our tool of choice: Python

Python is a **high-level, general-purpose language** with a focus on **code readability**.

"Python is powerful... and fast; plays well with others; runs everywhere; is friendly & easy to learn; is Open." – Python website

We will be using Google's Colaboratory environment.

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

– Colab website

What questions do you have?

Let’s get set up

In-class exercises

In-class exercises will normally be done in pairs.

For today only, I would like you to do them individually to make sure everyone learns how to interact with Colab.

You can work with someone if you were not able to bring your own computer.

Whatever you don’t finish in class, you should finish at home and turn in the following Tuesday.

These are graded on a combination of completion and correctness.