

# Operation Catapult

## Python Programming Projects – Day 1



# Welcome!

*Left* – A past Catapult team in action. They built a system to help solve high-school geometry problems. On the way, the team invented a new design tool, seen on the board here, which combines story boards and user stories (the Post-its on the pictures).

# Turn on machines now

---

Raspberry Pis are not to leave the room

Use only for projects

Who has a laptop they want to use?

- Please bring them tomorrow
- We'll help you install the software

# Your teaching team

---



Dr. Steve Chenoweth  
Associate Professor CSSE  
[chenowet@rose-hulman.edu](mailto:chenowet@rose-hulman.edu)  
Office @ Monech Hall, D215B

---

Derek Grayless  
Programming Student Assistant  
CSSE  
[grayledw@rose-hulman.edu](mailto:grayledw@rose-hulman.edu)

Anthony Galea  
Programming Student Assistant  
CSSE  
[galeaac@rose-hulman.edu](mailto:galeaac@rose-hulman.edu)

# Get to know each other

---

We will be in this room for 70+ hours over 3 weeks.

Name, hometown, and something interesting or unusual about yourself.

Why did you pick this project? What you hope to learn/do?

If you have programming experience, tell us about it.



# Website

---

Go to: <https://www.rose-hulman.edu/class/csse/catapult/>

Take 5 mins to explore what's there

If you have a laptop, the assistant will help you start to install the software.

# Overall Plan

1/2

---

For the first 5 days

- Instruction in the classroom aimed at those with no programming background
- Work in the lab on programs that we assign
- Start thinking about projects

# Overall Plan

2/2

Monday (or so)

- Finalize groups and projects

After that

- Work on your group's project
- Get help from your instructors and assistants
- There will be occasional “all together” classroom time



# Computer Programming with Python

---

**A lot of fun.** You should have a great sense of accomplishment by the end.

**A lot of learning.** About programming, about Python, about using pre-defined packages to handle many of the details.

**A lot of work.** Some of the things you will accomplish will not come easily. But you'll feel great when you have done them!



# Learning and Doing

---

Learning **Python**:

Repeat...

- Discussion time, followed by time for you to learn by writing small programs

...until you have mastered the basic concepts

Everyone will be doing the same small programs. Students with more programming experience may want to add extra features or they may want to work ahead

After that, groups will choose and work on their specific projects

# Maximize your learning

---

## DO

Your share of the driving

Ask questions

Ask for help when you are stuck

Help others around you or try new things if you are experienced

## DON'T

Let your partner take over

Think you are the only one who is confused

Try to figure it all out alone

Sit idle while waiting for others

# Class Discussion-Time Etiquette

---

**Goal:** learn Python quickly, so can get to projects

**Do** ask me questions at any time

**Do** interact with your neighbors about the things we are discussing, exercises you are doing. Always think, whether typing or observing!

**Don't** do things that will distract us from the learning process such as ...

- IM, email, ...
- using the speakers on your computer or other audio device.

# Lab Etiquette

---

Project time is time to focus on your project

- **not** the time to surf the web or send email or texts
- If a partner begins to get “off track”, a reminder from you may help
- We will take breaks

Let's speak to each other kindly, and only with words that our mothers would not be embarrassed to hear us say

Be very careful with drink in the labs

If you need to use the bathroom, get a drink of water, etc., it is not necessary to ask first

# What if you already know something (or lots!) about programming?

---

You'll still probably learn some new and important things in the next couple of days

If you find a concept or assignment to be easy, please put that to good use by patiently helping others around you to understand it

And think of “extra” features to add to the assigned programs

# Algorithms, Languages, Translators

---

What is an algorithm?

How do we communicate an algorithm to another person?

- Depends on the person. Has to be in a language they can understand.
- Baking a cake. Telling my mother vs. telling my 5-year old son.

# Language Barriers



What if we want to have Pierre perform an algorithm for us, but we do not know his language?

**You:** Can you take me to the Eiffel tower?

**Pierre:** Eh?

- We could hire a translator (a.k.a. compiler) to translate all of the instructions into French, and give them to Pierre.
- We could instead hire an interpreter. We read the next instruction to be done, the interpreter translates it into French, and Pierre does it.
- In the first case, it takes longer before Pierre can get started; in the second case, execution of each instruction is a little bit slower.
- Python is more like that interpreter.

# Understanding terminology and using it properly can enhance communication



Some important terms you will learn:

- function
- argument
- parameter
- string
- list
- class
- method



# Experimenting with the person next to you – a fun exercise!

---

Take a card

Find something to write with

Number off – 1, 2, 1, 2, 1, 2, 1, 2, ...

Without looking at what anyone else is doing...

- If you are a “1”, write down any problem!
- If you are a “2”, write down any solution!

Then...

# The experiment, continued...

---

Pair up with someone next to you who is the other number

Find some harebrained way to make the solution solve the problem!

Let's hear about some of the interesting problems and solutions!!!



# Python 1/2

---

Invented in 1990 by Guido van Rossum.

Named for “Monty Python’s Flying Circus”

Now has a huge worldwide following.



# Python 2/2

---

Makes it simple to do simple things.

Has the power to do complex things.

We think you will like it!

“Official” web site is [python.org](https://python.org).

- Also links on our website



# Other Resources

---

See resources linked to from our website

## Python Tutorials

- <https://docs.python.org/3/tutorial/>

## Python Programming E-Books

- Go to <https://www.rose-hulman.edu/academics/learning-and-research-facilities/logan-library/index.html>
- Click the + sign. Then, under Catalog
- Use “Python Programming” as the search term
- Select a book that you like, log in, and download the pdf

# I have no idea what you are talking about!

---

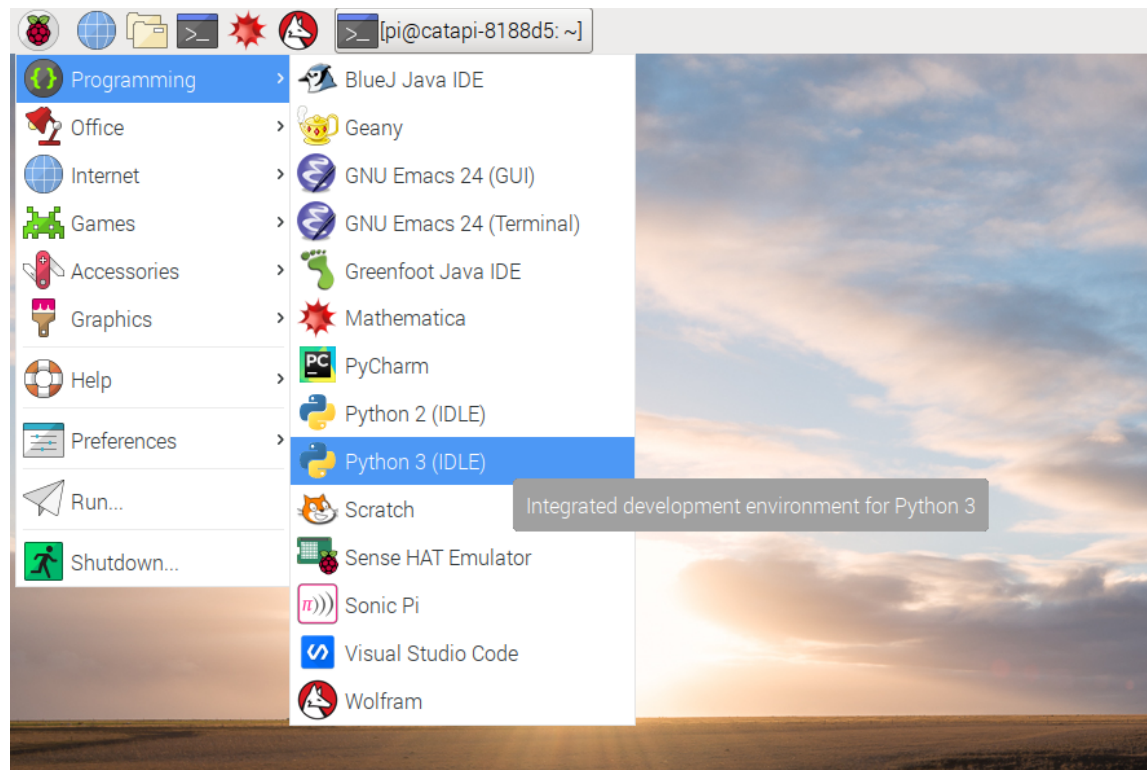
If I go too fast, or for any other reason you are not getting it, don't let me go on.

**Stop me, ask a question!**

During demo times, if you are stuck, raise your hand and we will come to help you.

# Python Demo

Open up the IDLE program from the RaspberryPi and follow along



# Python Demo

---

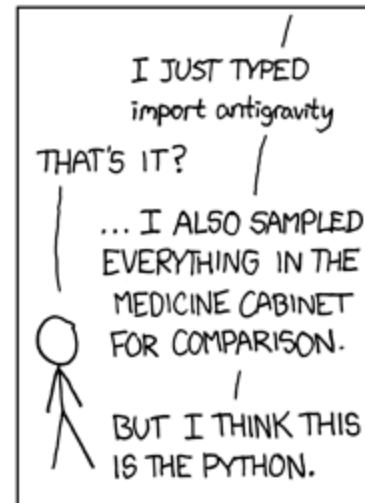
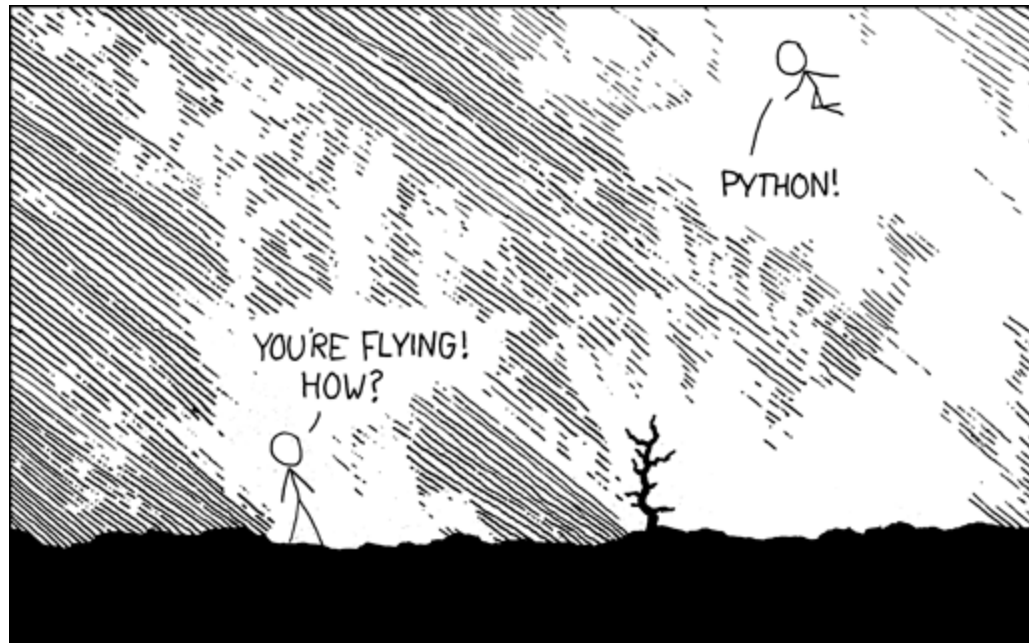
Ask questions as they come up

- If your questions are “advanced programming” questions, I may defer the answer so as not to confuse the beginning programmers

Demo of some Python features

- Enjoy the ride
- I will post most everything I show you to the website
- You'll use many of these features tomorrow





# Do the Catapult Surveys before you go

---

On Moodle