Operation Catapult Python Programming Projects – Day 1

Welcome!

Left – One of last summer's teams 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 tablet PCs now

- They are not to leave the room
- Username:
- Password:
- Ignore the DropBox message for today.
- We may need to double up on tablets for today.
- Who has a laptop they'd be willing to use in the lab?
  - Ideally, about 10 of you can do so
  - Please bring it tomorrow
  - We'll help you install the software

Multitasking for a few minutes

- Answer a quick survey on Moodle
  - www.rose-hulman.edu/moodle
  - Catapult Brief Survey
- Whatever you want to do
- *Soon*, we'll do introductions:
  - Briefly tell us who you are, where you are from, and at least one interesting or unusual thing about yourself.
  - These are a few of my favorite things …
  - Briefly describe your programming experience (if any).
  - What attracted you to this project, and what you hope to do.

Operation Catapult
Python Programming Projects - Day 1

- Dr. Aaron Wilkin - Assistant Professor
  - Office: M-230
  - email: wilkin@rose-hulman.edu
- Coleman Gibson and Steve Trotta– Programming student assistants.
  - Sophomore CS majors
  - trottasm@rose-hulman.edu
  - gibsonjc@rose-hulman.edu
  - Here during most group project times

Format

- Today.
  - More talk and less hands-on computer activities.
  - But we will still have some hands-on computer activities.
- After today
  - More hands-on time and less talk.

Today – Session Details

Topics - 1

- Intro to the instructors and each other
- Project Mechanics and rules
- Tour of online materials:
  - Catapult web page, schedule, Safari books on-line, python.org, etc.
  - What we need to do before you can start your project.
  - Can defer specific project choice and teams until Later in the week.

Today – Session Details

Topics - 2

- Introduction to Python (overview mode)
  - Starting Python's IDLE integrated development environment.
  - Python help and tutorial
  - Environment
  - Demo circle and tic-tac-toe programs
  - Python as a calculator
  - numeric computation, importing math module
  - variables and functions
  - strings, lists, range, loops
  - scripts
  - variables

Introductions

- We'll all be spending about 60 hours in this room together. Let's get to know each other a little bit first.
- You:
  - Briefly tell us who you are, where you are from, and at least one interesting or unusual thing about yourself.
  - Briefly describe your programming experience (if any).
  - What attracted you to this project, and what you hope to do.

And who we are…

- Coleman
- Steve
- Aaron

Introductions

Some things about me, a few of which might even be relevant to what we do here.

"Why are we called professors?
Because we are supposed to profess!"
- Frank Young (CSSE dept. head 1987-2003)

And Can't Google Tell You Everything About That Now?

My immediate family …

Wife:

Dog:

I did, too:
Went to Arkansas State University

- Majored in Computer Science
  - Never before touched a computer!!!

Before CS, I was a musician…

On the same day…

- Before…

After…

After graduating

- Pay…
- Master’s
- Pay…
- PhD
- Pay…

And …

- NCR makes the ATM's that you use to get your parents' money!
- These machines are actually pretty intriguing to design. Say,

  How long does the machine need to hold out, against being ripped out of its location by a chain attached to a pickup truck?

Here at Rose, since 2014…

- I teach software engineering and computer science…
- SE is, roughly, "all the things you need to know to work in the software business, which you don't learn in computer science classes"!
- At Rose, you can major in either one, or both.
- Which is unusual – Rose has only one of 22 accredited software engineering programs in the country. (See http://main.abet.org/aps/Accreditedprogramsearch.aspx.)

Possible time out with the non-programming students

- Python is an easy to learn, powerful programming language.
- It's a great place to start learning what programming is.
- Teams of people with no experience regularly succeed in this class!
- You'll get a sample today, already…

Python is a useful computer language!

- Widely used in industry
- Often used, even by very experienced programmers, to run "scripts" to do stuff
- The graphics and game-playing libraries make it useful to "simulate" things
  - Could be real stuff, too, like showing "how something works"
  - The code is "portable" and easy to read
  - Often used to do a "rapid prototype"
  - Great for "parsing" strings of information
  - Large libraries available for special purposes, like NumPy and SciPy

Other preliminaries

- What to call Aaron
- What to call Coleman and Steve
- Feel free to interrupt and ask questions at any time.
- Who this is aimed at
  - But if you know more …
- Not intended to go over your head
  - Slow me down if you are lost
  - Ask questions if you don't "get it”

We may have exactly the right number of computers!

- But if you brought your own laptop and want to use it, we'll help you get it set up.
  - If you haven't registered your "MAC" address with IAIT yet, that's the first step

Class Discussion-Time Etiquette

- Let's all work together toward getting you up to speed on Python quickly.
- Feel free to ask me questions (including "are you **sure** that what you said is right?" questions) at any time.
- No question is too dumb (except the one you don't ask).
- **Don't** do things that will distract others from the learning process. (such as …)
  - texting, etc.
  - using the speakers on your computer or other audio device.
- **Do** interact with your neighbors about the things we are discussing, exercises you are doing, etc.

Why do people think this isn't distracting?

→

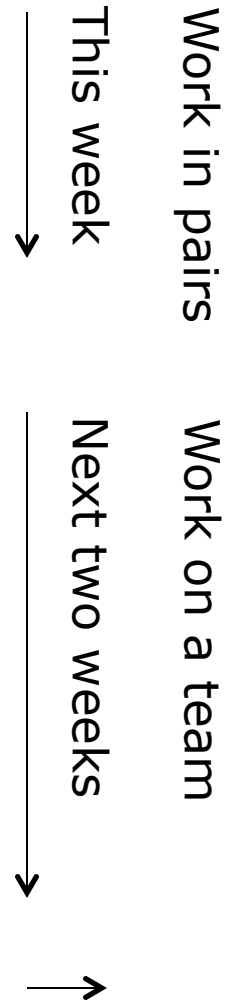
Lab Etiquette (continued)

- The lab during project time is **not** the place for you to text, to update Facebook, to browse the internet for things unrelated to Python programming, to send unrelated email or instant messages, listen to your music when working with others, etc.
  - You are welcome to do these things at other times, in our lab, in the Public computer lab (in Logan Library – I'll verify this) or the Dynamics lab (O-203 – should often be open).
  - If one of your partners begins to get "off track", a reminder from you may help.
- Let's speak to each other kindly, and only with words that your mother (and my mother) would not be embarrassed to hear us say.
- If you need to use the bathroom, get a drink of water, etc., it is not necessary to ask first.
- If your program uses sound, please use headphones, not speakers, to test it.

→

Schedule

- For most of this week
  - Instruction in the classroom.
    - Aimed at those with no programming background.
    - If you have experience and want to go faster, feel free to use the Python tutorial or the Zelle textbook.
  - Work in the lab on programs that I assign.
- Thursday (or so)
  - Finalize groups and projects.
- After that (probably beginning next Monday)
  - Work on your group's project.
  - Get help from Coleman, Steve and me as needed.
  - There will still be occasional "all together" classroom time when there is something that I think most groups need to know about.



Field Trip – to Beckman Coulter

- In Indianapolis
  - See link to company info on our home page
- TBA
- Field trip etiquette - Need to dress for a visit to a company with labs:
  - Long pants
  - Shirts with sleeves
  - Regular, closed-toe shoes (vs sandals)

Working with a partner on this week's assignments

- Remember that the main goal is for both of you to learn, not just to complete the assignment.
- Discuss what you are doing, to make sure that both of you understand it.
- Let the person with the least computer background do the "driving" most of the time
  - This makes it easier for that person to "put on the brakes" when he/she does not understand something.

→

Computer programming
Python Programming

- **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.
  - **I expect to learn a lot also.**
- **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!
- IMHO, in terms of learning and sense of accomplishment, most other Catapult projects are **wimpy** when compared to the programming projects.

Sometimes you may feel like you are in the dark for a while

- **But when the light comes on, it will be bright and beautiful!**
- **Sometimes you will need help in finding a path through the dark.**
  - **Don't be afraid to ask …**
    - **Me**
    - **Coleman**
    - **Steve**
    - **other students**

Learning and Doing

- The first week we'll focus on learning about Python and its various libraries.
- Repeat
  - There will be a little bit of discussion time, followed by time for you to learn by doing.
  - 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.

Learning By Doing…

- It’s the Rose-Hulman way!
- Try things yourself
- Experiment
- It’s ok to fail
- Engineers learn by breaking things
- It’s summer – be a little daring

Think John Cage on the piano!

Experimenting with the person next to you – an 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!
- We'll hear the results afterwards

How to avoid learning Python and OOP!

- Don’t stay on-task. Browse the internet or do instant messaging instead.
- Don't bother learning the terminology.
- Be satisfied with (only) getting the assignments working, even if you don’t understand them.
- Let someone else take over your keyboard and move too fast for you. She knows more than you do about programming, and you wouldn’t want to slow her down!
- Even if you are totally confused, don't ask for help. That might show your classmates or professor what you don't know!

How to avoid this project (continued)

- When you don't understand something, believe that you're the only one who is so stupid, so don't ask any questions!
- When you don't understand something, assume that it will come to you eventually; move on to the next thing (and wait for your lack of understanding to trip you up later).
- Don't bother Steve, Trevor, Austin, or fellow-Catapulters with your problems and questions; you can probably figure things out yourself
  - (in about 3 months!)

So … Do ask questions!

- Participate!
- Get involved!
- Don’t sit passively and let things roll by you.
- They may roll **over** you**!**

What if you already know something 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.
- Feel free to think of "extra" features to add to the assigned programs.

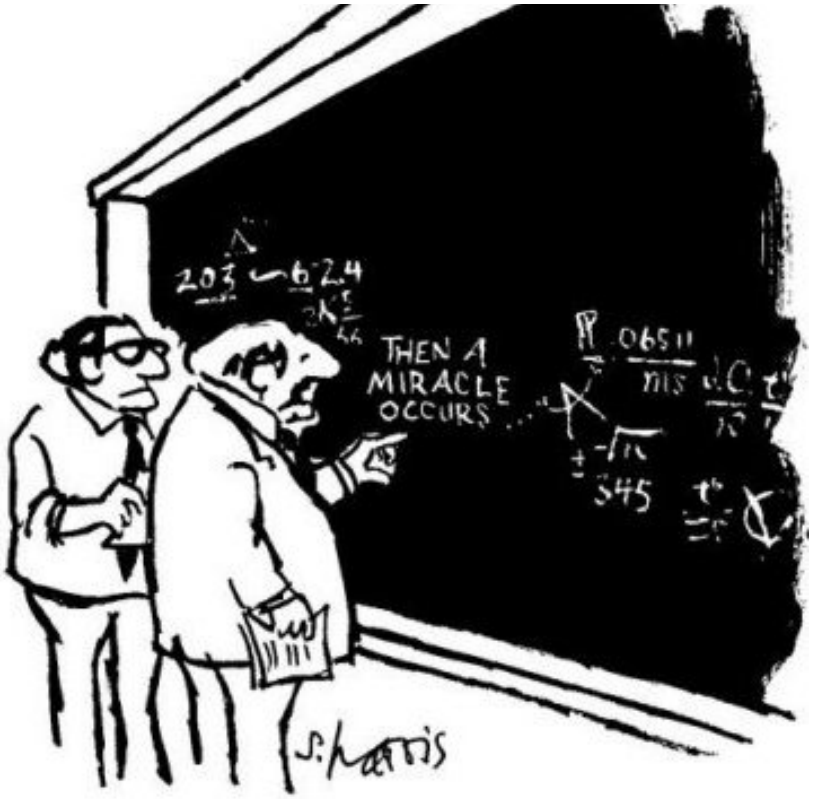
Working with a partner

- For the learning assignments this week, you will work with another person (preferably someone whose programming experience is similar to yours).
- Take turns "driving" the computer. If one of you spends more time in control, it should be the person who feels least comfortable.
- Each of you should be responsible for making sure that both you and your partner understand what's going on.
  - Ask each other "understanding" questions.
  - If neither of you is sure of the answer, ask someone else.

Some software places have special furniture for pair programming!

Understanding terminology and using it properly can enhance communication

- Some important terms you will learn:
  - function
  - argument
  - module
  - string
  - list
  - class
  - method



"I THINK YOU SHOULD BE MORE EXPLICIT HERE IN STEP TWO."

The Big Secret!

- I don’t know everything about the Python or its class libraries.
- Are you horrified?
- There are dozens of packages, containing hundreds of classes with thousands of methods.
- I am enjoying learning all about Python, and Coleman, Steve and I will enjoy working with you as well.
- If you want to do something that we don't know how to do, we will try hard to point you to something that will help you do it, and to help you figure it out if you need us.

Take a break for 5 minutes

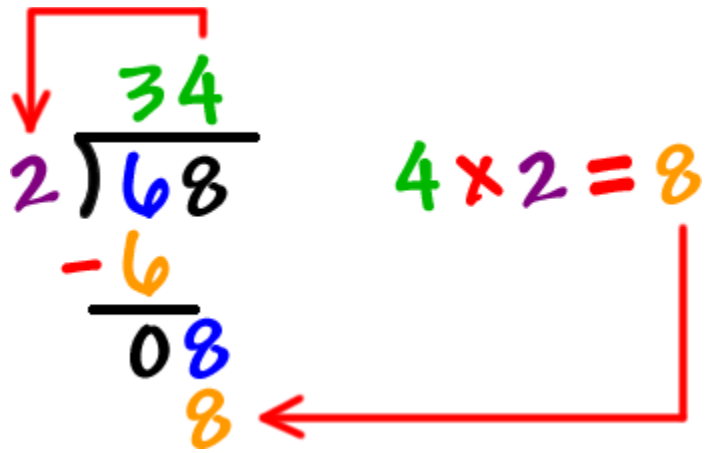
The Python language was named after Monty Python

- So we'll watch one of Monty Python's most famous sketches.

- http://www.youtube.com/results?search_query=monty+python+dead+parrot&aq=2&oq=monty+py

Algorithms, Languages, Translators

- What is an algorithm?
  - Like long division.
- How do we communicate an algorithm to another person?
  - Depends on the person. Has to be in a language they can understand.
  - E.g., explaining *this* long division to a child who already knows long multiplication, versus one who doesn't



→

Language Barriers

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

You: Can you take me to the Eiffel tower?
Pierre: Eh?

→

Python

- Invented in 1990 by Guido van Rossum.
- Named for "Monty Python's Flying Circus"
- Now has a huge worldwide following.
- "Official" web site is python.org .
- Makes it simple to do simple things.
- Has the power to do complex things.
- I think you will like it!

Here's Guido, showing what he is partial to…

As far as we know, this Rossum did not invent any Universal Robots!

I have no idea what you are talking about, Aaron!

- 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 "all together" demo times, if you are stuck, raise your hand and look at Coleman or Steve; they will come to help you.
- Or perhaps someone near you can help.

Login and Website

Most of the things I'm using are at:

http://www.rose-hulman.edu/class/csse/catapult/2015-S2/index.html

If your Firefox browser isn't set for this as a home page, then after the first time typing this in, you can put it on your desktop!

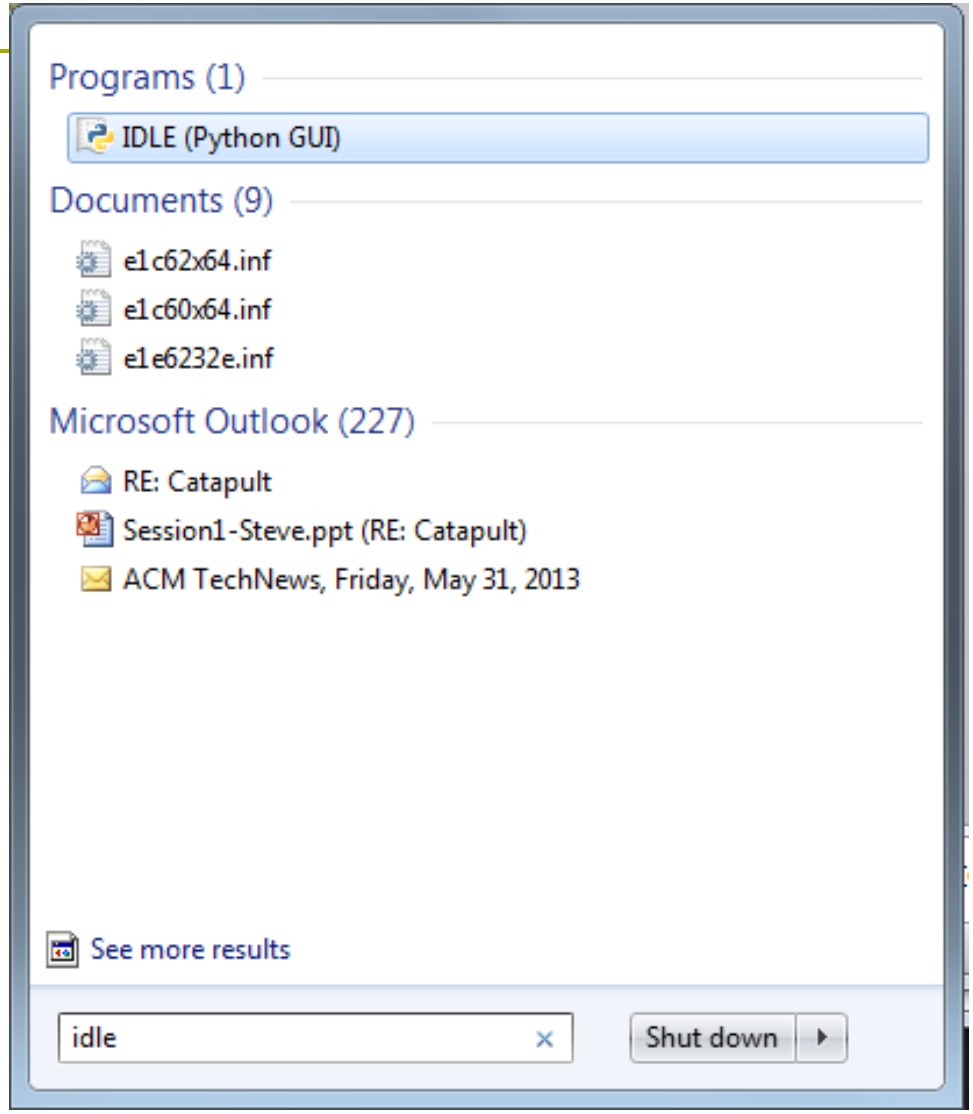
Most of the material that is there now is from last session; it will be updated as we go along, but I wanted to have something for advanced students to look at.

Other Resources

- Optional textbook (the one we use for our RHIT first programming class):
  - Zelle: *Python Programming*
  - We'll try to get a couple of copies for the lab.
- http://docs.python.org/tut/tut.html
- Safari Tech Books online
  - Go to http://www.rose-hulman.edu/Library/, under Databases choose Safari, then click Go. Then click Programming, and Python
  - Useful books:
    - Python for the Absolute Beginner
    - Learning Python
    - Programming Python
    - Python in a Nutshell
  - Our license only allows a few simultaneous users, so please logout when you are done.

Python Demo

- You should open up the IDLE program from your desktop and follow along.
  - Take turns, if sharing a machine
- If it's not on your desktop, you can go to the Start menu → and type *idle* (it's the Python GUI, like it says) :



Python Demo, cntd

- Once again, ask questions at any time.
- This is a "first glimpse"
  - Not "from now on, you gotta know all this!"
- If your questions are "advanced programming" questions, I may defer the answer so as not to confuse the beginning programmers.
- First look at Help, then demonstrate some Python features.
- We will do more detail on many of these Python features tomorrow. For today, just enjoy the ride.
- At the end today, we'll note how all this ties into your team projects!

Python Programming Project List - 1

- **Strategy Game** — Build a computer program that allows users to play your favorite board game, such as Connect 4, Boggle, or Scrabble. Then add a computer player who is good enough to defeat most human players. Perhaps we can have multiple computer players battling each other.
- **Action Game** — This is the most common Catapult computer programming project. Build a computer program that implements a classic game like Asteroids, PacMan, Frogger, or Space Invaders. Or make up your own game! Students with prior programming experience may want to incorporate networking so that multiple players can participate remotely.

Python Programming Project List - 2

- **Auction Site** — Write a client-server program that allows people to advertise items for sale, to bid on items, to provide feedback on good and bad transactions with other users, etc.
- **Database Application** — Learn to write a Python program to create and maintain a database. Examples might include cataloguing your CD collection, organizing major league baseball statistics, managing your "little black book" of romantic interests or your school's grade information for current students, or perhaps simulating an "on-line store" or an on-line catalog of information about your favorite subject.

Python Programming Project List - 3

- **Spelling Checker / Suggester** — When your document contains a misspelled word, many spell checkers provide suggestions of alternate words. Do you ever get frustrated when your misspelling is very simple (you left out a letter, added an extra letter, or transposed two adjacent letters) but the computer's list of suggestions does not include the word that you meant to use? Perhaps your checker/ suggester can do better!
- **Make up your own program** — Design and write an imaginative, exciting computer application that may dazzle your friends back home. It can fit one of the above categories, or be something entirely different that you'd like to explore.