Fundamentals of Python

Syllabus

Aniello De Santo

University of Utah
aniello.desanto@utah.edu

Aug 26, 2020

Organizational Information

Course	Fundamentals of Python
Course#	LING 5981/6080
Room	GC 2575
Time - Mondays	Asynchronous
Time - Wednesdays	W 3:00–4:20
Website	Canvas and Slack
Instructor	Aniello De Santo
Pronouns	he/his/him
Email	`aniello.desanto@utah.edu`
Office hours	M 1:30 – 3:00 ; W 2:00 – 3:50
Online only meeting	W 1:30 – 2:00
Office	Zoom

See the Canvas course page for more details and announcements.

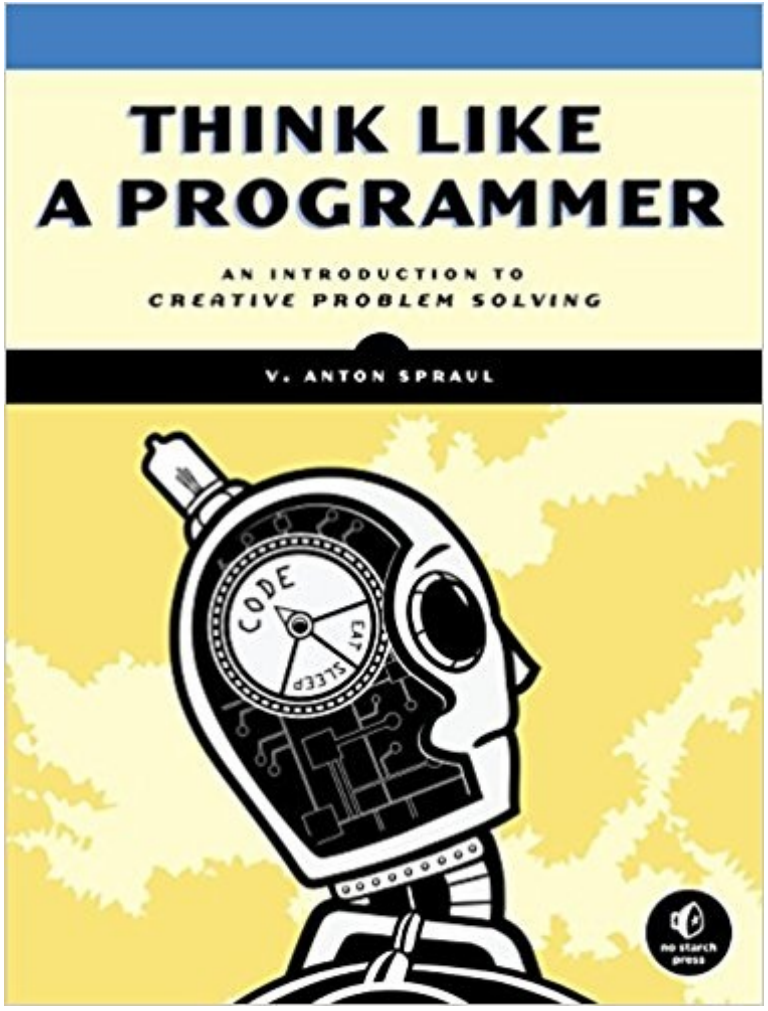
Catalog Description

This course is an introduction to programming principles using Python. Students will acquire basic programming skills, knowledge of fundamental coding concepts, and the ability to write scripts for simple language-oriented tasks. Linguistic examples will be used to motivate the introduction of new coding concepts. We will discuss the language technologies of our daily life - spam filtering, machine translation, and more - and how they work under the hood.
No previous training in mathematics, linguistics, or computer science required.

Learning How to Code?

By the end of the course, you will have basic Python literacy.

More importantly, you should have refined your problem solving skills.



An Experiment

1. Open some chat or messaging app on your phone.
2. Start typing a random letter.
3. Click the second word suggestion
   (the one in the middle).
4. Keep doing the previous step for a while (say, 10 to 15 words).
5. Did you get a reasonable sentence of English?

An Experiment

1. Open some chat or messaging app on your phone.
2. Start typing a random letter.
3. Click the second word suggestion
   (the one in the middle).
4. Keep doing the previous step for a while (say, 10 to 15 words).
5. Did you get a reasonable sentence of English?

   I am a beautiful person who is the best of luck to you by the way to get the best of luck to you by the way to get the best of luck to you by the way to get the . . .

An Experiment (part 2)

Give me instructions
on how to open my water bottle

Goals and Objectives

- acquire core programming skills generalizable to any programming language
- master essential concepts and techniques in Python programming
- translate abstract computational models into fully functional source code
- develop learning autonomy and the ability to deepen your programming knowledge through self-study

Prerequisites

- **What You Need**
  - ability to operate a computer
    (use a web browser, install software, edit text files)
  - willingness to play around with open-ended problems
- **What You WON'T Need**
  - programming experience
  - math (except for addition, multiplication and fractions)
  - linguistics

Course Set Up

- **Lecture notes and HW** written in Jupyter Notebook
- I suggest using Google CoLab for execution
- **Participation and peer discussion** Course Slack channel
  (get in touch if your invite has expired)

Two Types of Instruction

Monday	Asynchronous guided self study Notebooks with study materials uploaded on Canvas
Wednesday	in-class programming sessions in Python **come with questions!**

Online only students...

... will meet with me once per week for regular check ins.

What happens if we all move online?

In case of a sudden move to full online instruction, Wednesday lectures will become discussion sessions via Zoom. Each student will be required to submit a question beforehand.

Two Types of Instruction

Monday	Asynchronous guided self study Notebooks with study materials uploaded on Canvas
Wednesday	in-class programming sessions in Python **come with questions!**

Online only students...

... will meet with me once per week for regular check ins.

What happens if we all move online?

In case of a sudden move to full online instruction, Wednesday lectures will become discussion sessions via Zoom. Each student will be required to submit a question beforehand.

Grading Components

Class Participation (25%)

- Both in class and **online**!
- **Examples**:
  - ask questions (in person, via email, via Slack)
  - help fellow students (in person, via Slack)
  - link to relevant online materials (e.g. in Slack)
  - ⋮
- **Why?**
  - Encourages you to ask questions.
  - Helping others is a great way of learning.
  - We want to have some fun, too.

Grading Components [cont.]

Python Exercises (70%)

- once per week
- programming in **Python**
- assigned on Mondays
- due the following Sunday at 11:59pm
- assigned via Canvas, collected via email
- late hand-ins?
- **Why?**
  - Learning programming is like learning a new language
    $\Rightarrow$ needs constant practice
  - Even a little bit of programming experience is incredibly useful.

Grading Components [cont.]

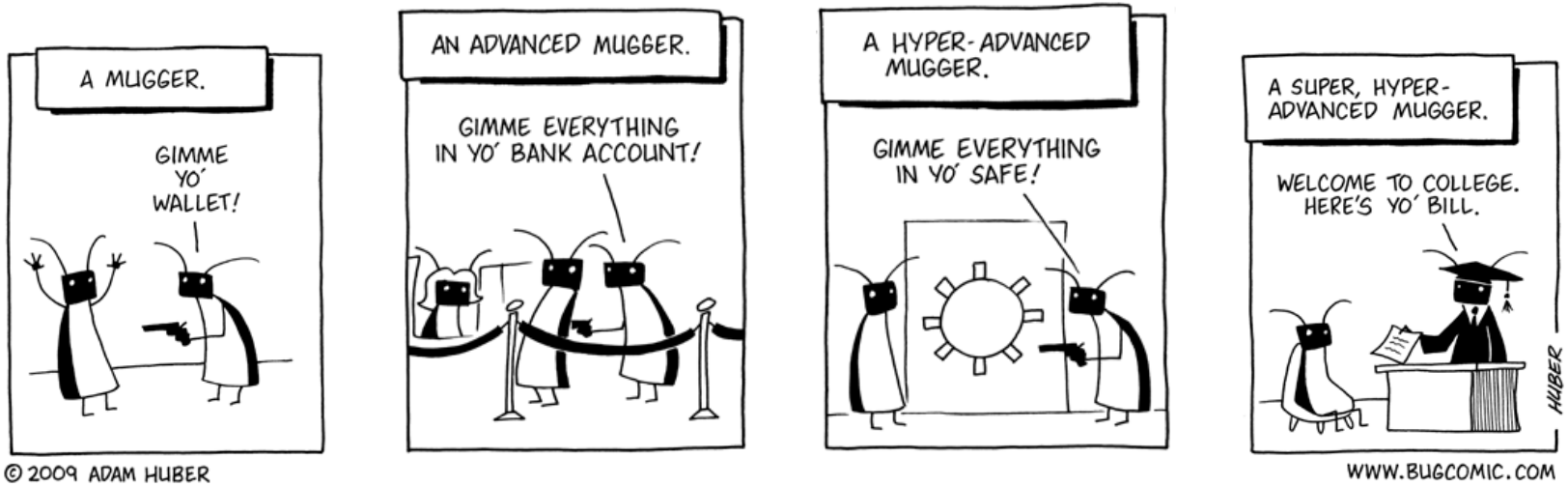
Midterm (5%)

- Tentatively in **Week 9**
- Pen-and-paper coding assignment
- **Why?**
  - Force you to check how your study method is working, and eventually correct course
  - Pen-and-Paper coding helps focus on solving the problem and not on the minor details of the coding language (i.e. Python).

Dealing with Fails

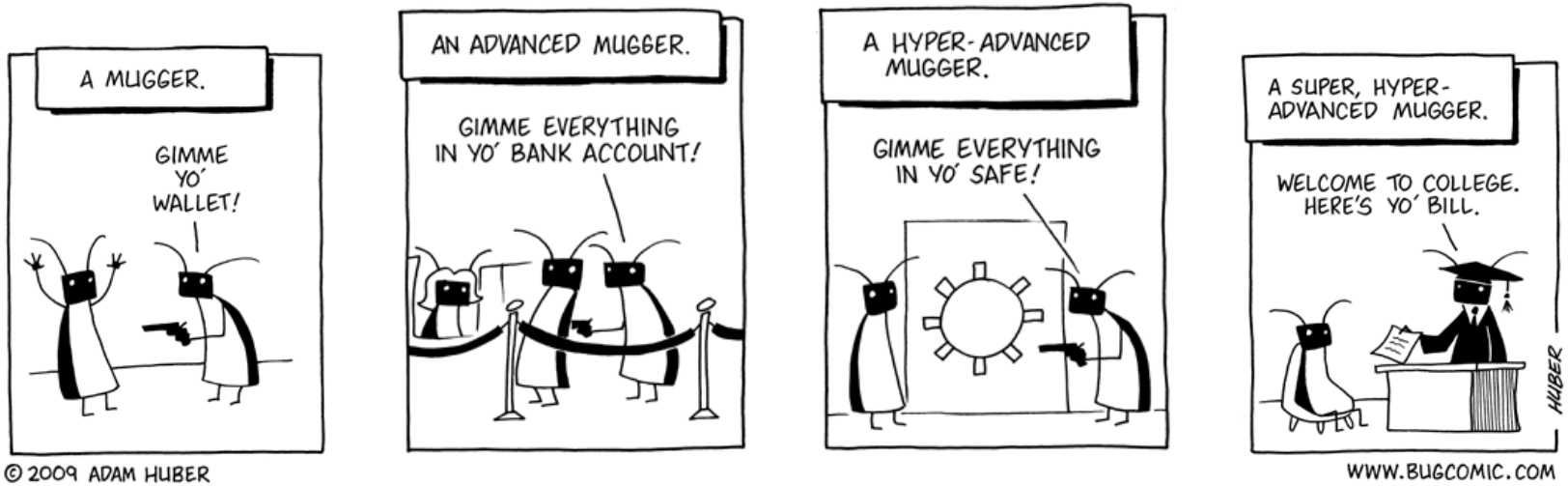
- Extra-credit opportunities (5% each)
  - Participate to department events
  - Participate to online experiments (info on Canvas)
- Optional **Final project** for Python.
  - Extra-credit, worth up to 10% of the total grade
  - Due during finals week together with a class survey.

Soapbox: Thoughts on Grades



- Students are caught up in the **grade bubble**:
  - If I get good grades I will get a job.
  - If I get bad grades I will fail in life.
- In the real world, nobody cares about your GPA.
- Don't focus on grades!
- Focus on mastering the skills you need to get the job you want.

Soapbox: Thoughts on Grades

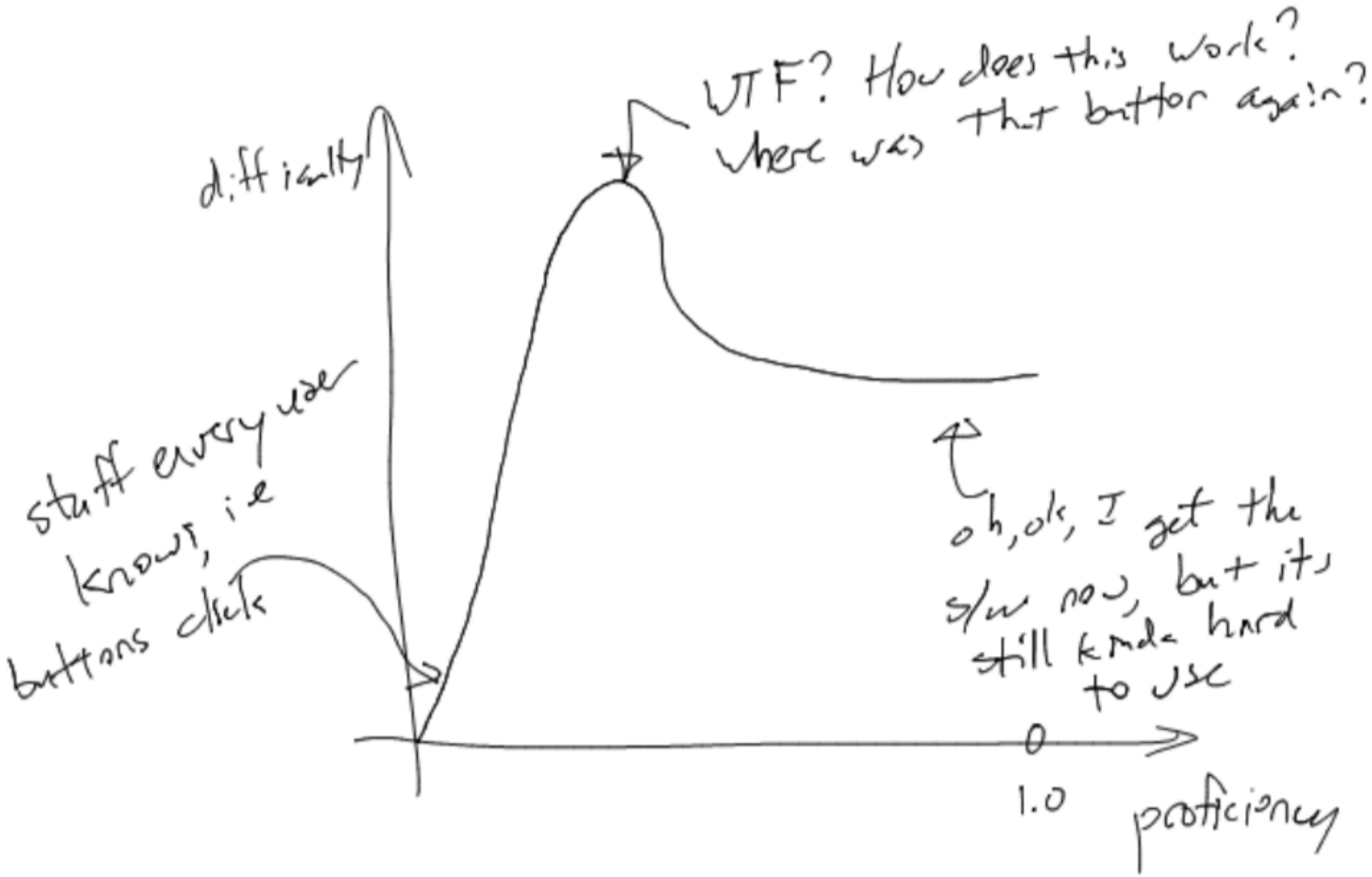


- Students are caught up in the **grade bubble**:
  - If I get good grades I will get a job.
  - If I get bad grades I will fail in life.
- In the real world, nobody cares about your GPA.
- Don't focus on grades!
- Focus on mastering the skills you need to get the job you want.

Soapbox: Our Role in This

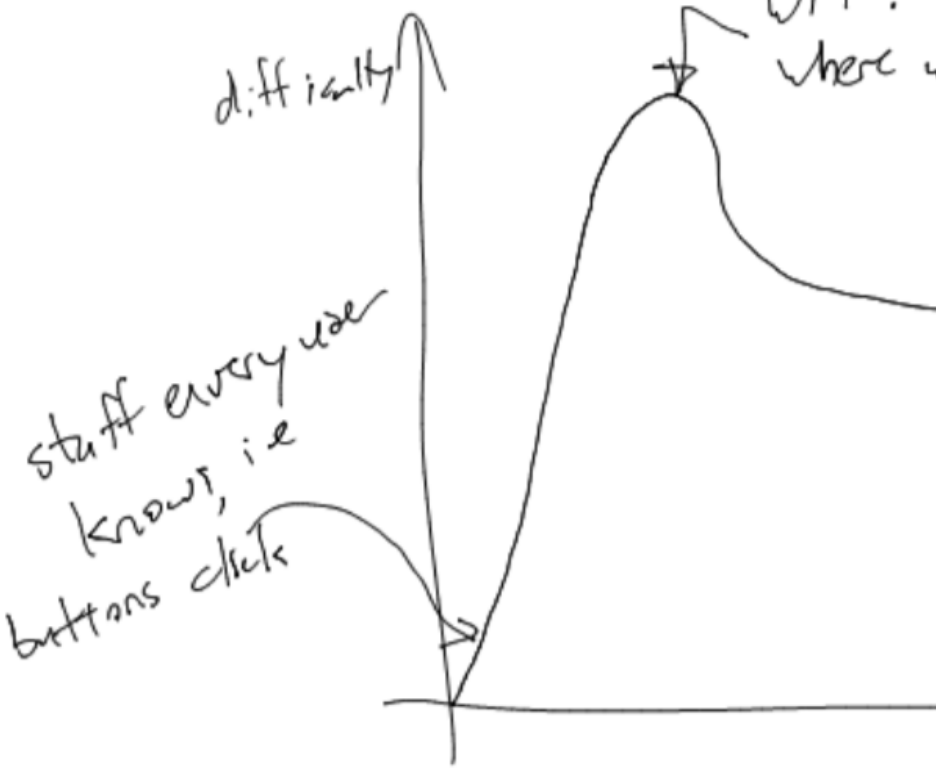
- Instructors are the academic equivalent of a **fitness trainer**.
- You're paying thousands of dollars for us to get you into shape, and we've developed a program for you that will do that.
- But you are the one who has to move their body.
- Bad techniques like cram learning may get you a good grade, but you're cheating yourself out of true progress.
- If you aren't working towards long-term intellectual growth, you're flushing tons of money down the toilet.

The steep learning curve!



The steep learning curve!

- Don’t despair! It takes time!



TL/DR: Ask for Help

- **Take advantage of me**
  I put a lot of effort into helping you achieve your goals:
  - self-study materials
  - office hours
  - availability via email and zoom

  Give me (constructive) feedback on problems with the materials!
- **Take advantage of each other**
  Your peers are a valuable resource, too. Discuss homework, exchange ideas, share notes. Collaborate, help each other.
- **Don't wait too long**
  The Matthew effect also applies to education: the rich get richer, the poor get poorer. If you sense yourself falling behind, ask for help right away. The longer you wait, the worse it gets.

Getting Help

- By default: use Slack

Optimizing Response Time

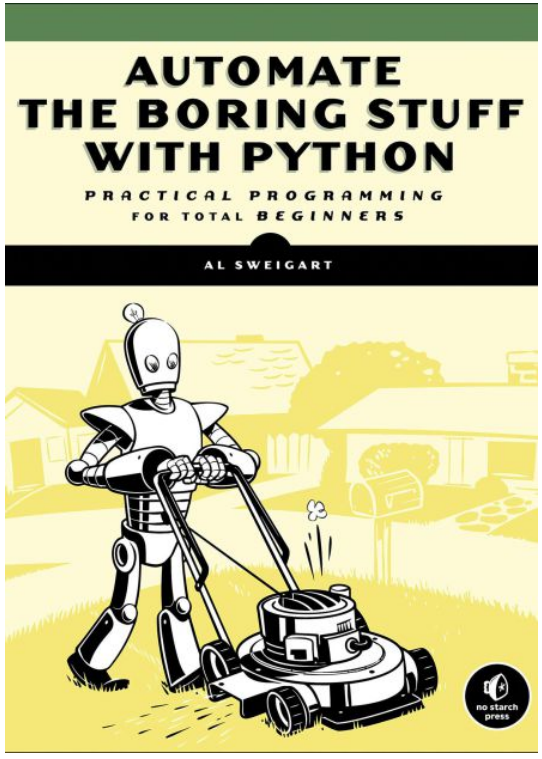
- Minor technical issue?
  → Slack > Me
- Homework/General Python Question?
  → Slack > Me
- Grading question? → Me
- Personal Issues? → Me

Getting Help [cont]

- Contacting me:
  - *{aniello.desanto}@utah.edu*
  - Put [LING 5981/6080] at the beginning of the email *Subject*
  - Reply time usually $<$ 24h (no guarantee during weekends!)
  - If you plan to come to our office hours but anticipate a long meeting, drop me a line the day before.
  - If there's a scheduling conflict, I'll let you know. Radio silence means everything is fine.

Supplementary Textbook (Optional!)

- Al Sweigart (2015):
  Automate the Boring Stuff with Python
- online version **free**
- digital versions and hardcopy around $25
- supplementary videos on Youtube
- **It is not required**
  but it's a good source to consult if something is unclear.



Tentative Schedule

Week	Monday	Wednesday	HW # Due
1	Syllabus	Intro to Jupyter and Colab	No HW Due
2	Notebook 1: Variables and Data Types		No HW Due
3	Labor day	Notebook 2: Flow control	HW 1 Due
4	Notebook 3: Lists and for-loops		HW 2 Due
5	Notebook 4: String Methods		HW 3 Due
6	Notebook 5: Dictionaries (fully online)		HW 4 Due
7	Notebook 6: File IO (fully online)		No HW Due
8	Notebook 7: While loops		HW 5 Due
9	Midterm Practice and Q&A	Midterm	HW 6 Due
10	Notebook 8: Function definition		No HW Due
11	Notebook 9: REG Expressions		HW 7 Due
12	Notebook 10: Classes		HW 8 Due
13	Notebook 10: Advanced N-gram		HW 9 Due
14	Buffer (maybe data manipulation with pandoc)		HW 10 Due
15	Buffer & General discussion (fully online)		No HW Due
16	Finals week	Class Survey Due	No HW Due

A Recap: Format from now on

1. Notebook X will be uploaded Sunday night
2. Work on it through the week
   - Play with the example code
   - Do the practice exercises
   - start looking at the HW questions
3. Come on Wednesday with questions
   - We will look together at the main concerns
   - We will look at the HW exercises
   - We will do extra practice
4. Submit the HW via email by Sunday night.