

# **OIM3640 - Problem Solving and Software Design**



# Agenda

- Introducing yourself
- Introduction to the course
  - Syllabus
  - Term Project (*mentioned*)
  - Software
- Write your first Python program!

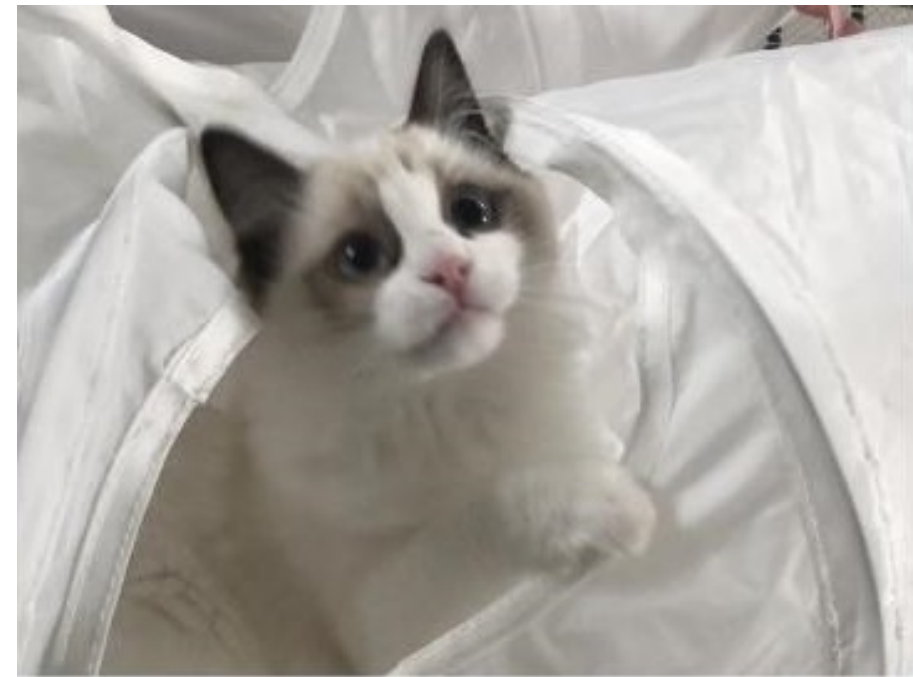
# Welcome! How are you doing?

Please introduce yourself, including:

- **Who** are you? **Where** did you come from?
- Are you a **sophomore, junior** or **senior**? What was your **best Babson moment**?
- What is your **concentration**? Are you going to rule the world with that?
- **Why** did you decide to take this class?
- Do you have any **programming experience**?
- How was your **2022** and your **winter break**? Did you do **anything exciting** or just binge-watch a lot of Netflix?
- How can we **remember** you?

# About Me

- Instructor: **Zhi Li** (李直)
- Email: [zli@babson.edu](mailto:zli@babson.edu)
- Office: Babson Hall 216D
- Office Hours:
  - In-person:
    - Tuesday: 11:30AM - 12:30PM
    - Thursday: 6:30PM - 7:30PM
  - Online via Webex: by appointment



# A Quick Survey

1. What **programming languages** have you heard of?
2. What **IDE** have you used?
3. Have you used **Git/GitHub** before?
4. Have you created **website(s)** before?

# What is this course about?

Well, let me first tell you that this course is **NOT** about...

- Computer science theories
- Solving problems without programming
- Data structure and algorithms
- Front-end technologies
- Game development and graphics programming
- Scientific computing, big data and data visualization
- Advanced libraries such as `pytorch` for deep learning
- Distributed systems, cloud computing and microservices

# Seriously, what is this course about?

- Variables, expressions and statements
- Basic data types (e.g. integers, strings, lists, dictionaries)
- Control structures (e.g. loops, if-else statements, for/while loops)
- Functions and modules
- File Input/Output
- Basic Object-Oriented Programming (OOP) concepts
- Exception handling
- Basic built-in and third-party libraries
- Database access and SQLite
- Text Analysis
- Web scraping
- Popular web frameworks (Flask)
- Basic Algorithm and problem solving strategies
- Debugging and testing
- Version control and collaboration
- Pair programming
- ...





# What really matters are ...

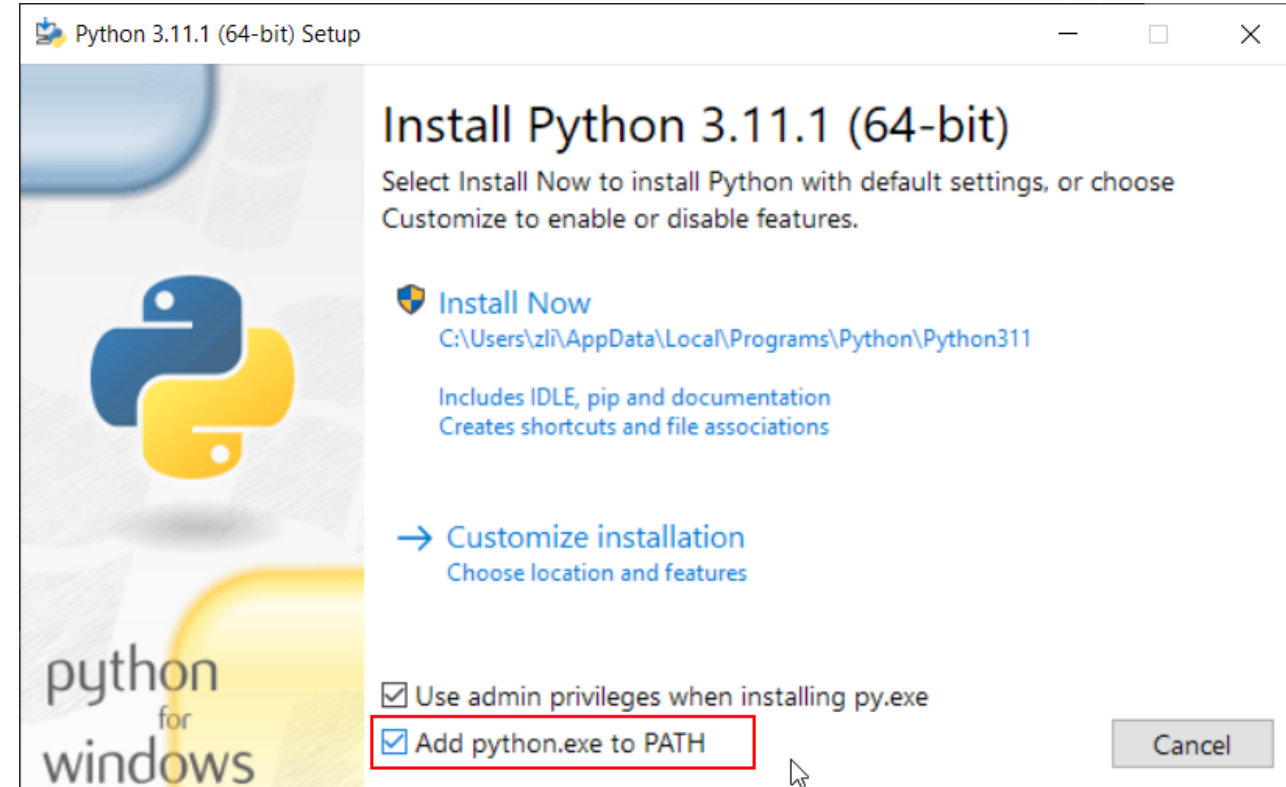
- Familiarizing yourself with basic programming concepts
- Thinking like a software engineer and a computer scientist
- Learning how to learn programming and how to get "unstuck"
- Collaborating effectively with engineers and other team members through the use of tools and clear communication
- Equipping you with the tools and mindset to succeed after completing this course

# Syllabus

- Course Description and Learning Objectives
- Prerequisites and Textbook
- Software (next slide)
- Exercises / Assignments / ~~Exam~~
- Term Project
- Grading
- Course Policies

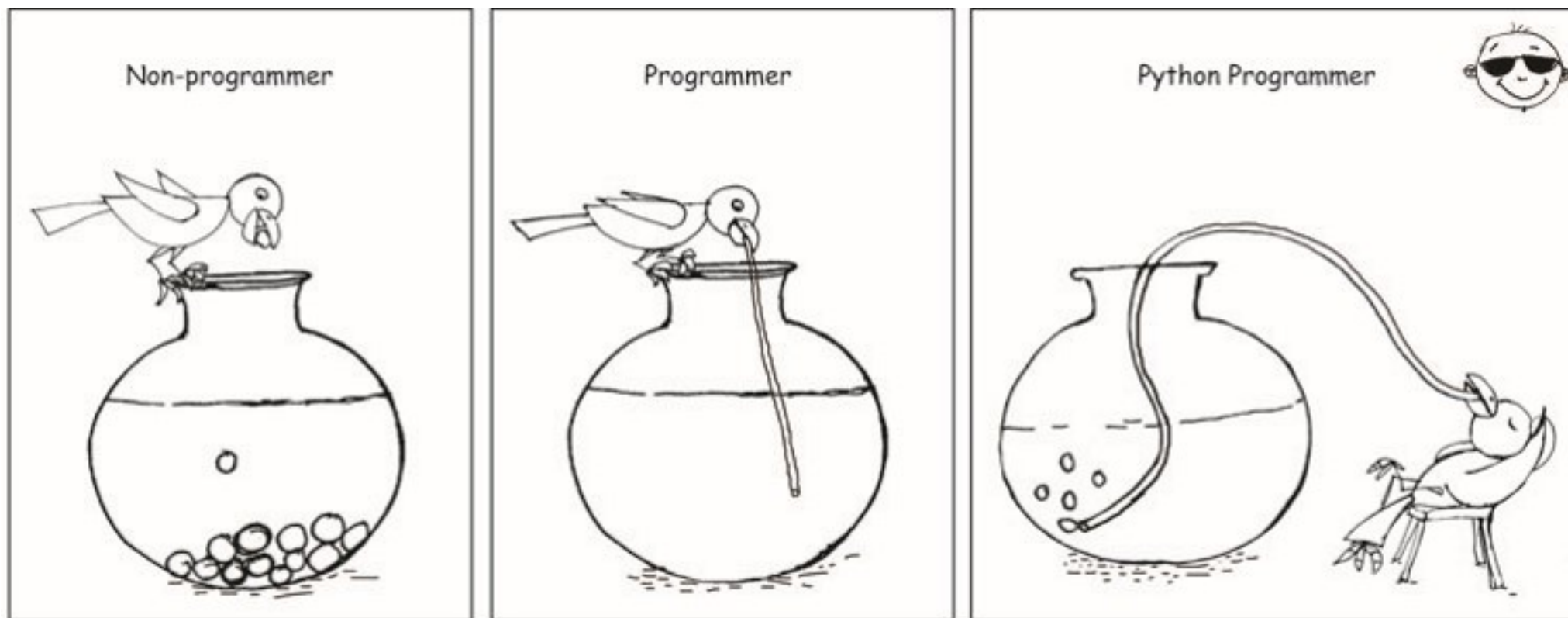
# Software

- **Python**
  - **"Add python.exe to PATH"**
- Visual Studio Code (**VSCode**), and extensions
  - Python
  - vscode-icons
- **GitHub** Desktop
  - Sign up for [GitHub](#) (using Babson email)



# How to Learn (Python) Programming?

# Python is ...Simple and Elegant!



---

# How to draw an Owl.

---

*"A fun and creative guide for beginners"*

---

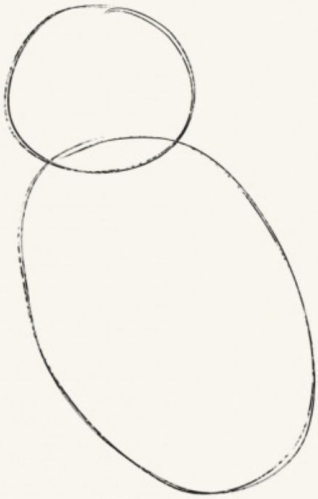


Fig 1. Draw two circles

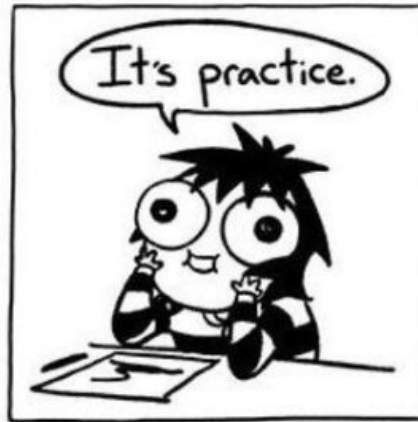


Fig 2. Draw the rest of the damn Owl

**... but programming is hard, like, really hard!**

**DO NOT** take the “couch potato” approach





©Sarah Andersen

**Practice!**

**Practice!**

**Practice!**



**DO NOT** copy and paste!





# Ask Questions

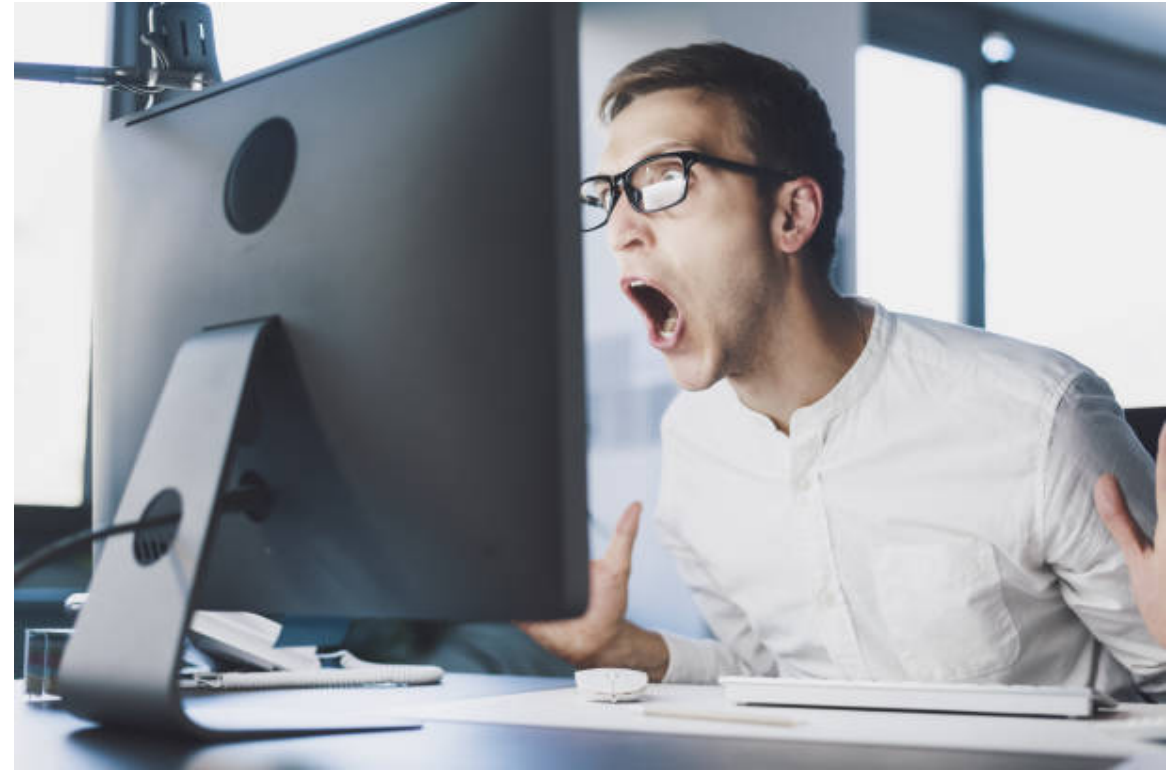


# Ask Questions the Smart Way

- The XY Problem
- How do I ask a good question
- How To Ask Questions The Smart Way
- Getting Answers

# DO NOT panic!

- Almost everyone hits a rough patch in the course at some point.
- Don't let it discourage you.
- It's normal!





## What if you got "stuck"

- Take a break
- Break the problem down
- Keep trying
- Debug
- Ask for help



# More on Learning to Code

- People **new** to programming don't know what it is like to be a programmer, or to take a programming course.
- Having unrealistic expectations, such as expecting things to work perfectly **the first time**, can lead to disappointment.
- It's important not to give up at the first sign of an error.
- **Expect:**
  - to spend hours, entire nights, even multiple days getting things to work
  - to learn a lot by yourself
- The **real learning** happens when you encounter obstacles and overcome them.
- Avoid relying on **spoon-fed** answers, as it can lead to weaker problem-solving skills in the long run.

# How to Cheat without Being Caught

- If you're going to cheat, here are some tips:
  - Do not submit code that has a matching md5sum as your friend's code.
  - Don't share your code with others if it is supposed to be individual work.
  - Avoid simply changing comments and spacing, as the code can be tokenized to eliminate these differences.
  - Changing variable names, moving definitions, or copying only part of other's code will be detected as well.
  - You may not use code found on the internet or written by AI.
- If you still decide to cheat, the **only way to cheat safely** is to rewrite the assignment from scratch.



# Collaboration Policy

- For final projects and other specified group projects, you may work in a team and submit a single, joint assignment.
  - It's important to review and follow the guidelines as it may vary per project.
  - You also have the option to work independently.
- Forming study groups is encouraged.
  - However, for group work, you must not share code with anyone other than your assigned partner. For individual work, sharing code with any other person is prohibited.
- To truly learn and understand, you must independently write your own code.
- Any violation of this collaboration policy will be handled under the school's academic misconduct policy and may result in a failure of the course.

# Questions?

**NOW, PREPARE  
TO GET YOUR  
HANDS DIRTY!**

