Before We Start

Talk to your neighbors:
Coffee or tea? Or something else?

Music: 123 24su Lecture Tunes

Instructor: Joe Spaniac
TAs: Andras Daniel, Eric Nicole, Sahej Trien, Zach
Lecture Outline

• Announcements

• Finishing up Lines & Graphs

• Abstract Classes
  - Revisiting Graphs
  - Ciphers

• Revisiting Reflections
  - Creative Project 1
Announcements

• Programming Assignment 1 due Wednesday, July 3rd at 11:59 PM
  - Recommend getting started early (trickier assignment)
• C1 grades and feedback will also be released Wednesday
  - General grading turnaround is ~1 week
• Resubmission Cycle 1 will be released later today
  - Due next Friday, July 5th at 11:59pm
  - Eligible assignment(s): C1
• Quiz 1 is Tuesday, July 2nd!
  - Topics: Object-Oriented Programming, Testing, Inheritance, Polymorphism
  - Check Ed later today for a post containing logistics and a practice quiz
  - Also see Check-in 1 (resources tab) for an example quiz problem
Lecture Outline

• Announcements

• **Finishing up Lines & Graphs**

• Abstract Classes
  - Revisiting Graphs
  - Ciphers

• Revisiting Reflections
  - Creative Project 1
Lecture Outline

• Announcements

• Finishing up Lines & Graphs

• **Abstract Classes**
  - Revisiting Graphs
  - Ciphers

• Revisiting Reflections
  - Creative Project 1
Revisiting Graphs
Revisiting Graphs
**Abstract Classes**

- **Mixture of Interfaces and Classes**
  - **Interface similarities:**
    - Can contain (abstract) method declarations
    - Can’t be instantiated
  - **Class similarities:**
    - Can contain method implementations
    - Can have fields

- **Is there identical / nearly similar behavior between classes that shouldn’t inherit from one another?**
Revisiting Graphs

Graphable

Point

Line

Graph

MonoGraph

MultiGraph

ColorGraph

--- Implements

--- Extends

- Utilizes

Abstract Classes

Classes

Interfaces
Ciphers

DeclaredType x = new ActualType();

Cipher c = new SwapCipher('A', 'B');
c.encryptFile("test.txt");
Advanced OOP Summary

• Allow us to define differing levels of abstraction
  - Interfaces = high-level specification
    - What behavior should this type of class have
  - Abstract classes = shared behavior + high-level specification
  - Classes = individual behavior implementation

• Inheritance allows us to share code via “is-a” relationships
  - Reduce redundancy / repeated code & enable polymorphism
    - Still might not be the “best” decision!
  - Interfaces extend other interfaces
  - (abstract) classes extend other (abstract) classes

• You’re now capable of designing some pretty complex systems!
Design in the “real world”

• In this course, we’ll always give you expected behavior of the classes you write
  - Often not the case when programming for real
  - Clients don’t really know what they want (but programmers don’t either)

• My advice:
  - Clarify assumptions before making them (do I really want this functionality?)
  - Don’t let preemptive optimization make you freeze!
  - **There’s no one right answer**
    - Weigh the options, make a decision, and provide explanation
    - Iterative development: make mistakes and learn from them
    - Be receptive to feedback and be willing to change your mind
Lecture Outline

• Announcements

• Finishing up Lines & Graphs

• Abstract Classes
  - Revisiting Graphs
  - Ciphers

• Revisiting Reflections
  - Creative Project 1
Revisiting Reflections

• Throughout this course, we’ll ask you to form opinions on topics
  - Provide exposure to issues so you can decide for yourself

• Opinions aren’t formed in a vacuum
  - Exposure to various viewpoints reinforces/challenges perspectives
  - Shouldn’t be making arbitrary decisions
    - Rationalization is often important! (Not always necessary, but helps in communication)

• Integrating reflections to in-class components
  - Discuss opinions, challenge assumptions, potentially change minds
  - Please be respectful of other people’s opinions
    - There are no “right” or “wrong” answers to these questions
    - Everyone has different experiences with the world that informs their decisions
C1 Reflection

• Video: *The Moral Bias Behind your Search Results*

• Q3: Do you think whoever comes up with moral rules and judgements surrounding search engine ranking results at Google should have that power / responsibility?

• Q5: Do you think that software reflects the biases of the programmer? Have you ever encountered bias programs / applications?