Recurring Section 3

Aniruth – 5 PM

Announcements

- The topics for the content review depend upon responses to the final question in the recurring section feedback form!
  - I'll make a guess as to what to review, but if there's anything specific you'd like me to cover, please add it there.
  - It's editable and will send to your email so you can access it later.
  - Feedback Form: https://forms.gle/anpxMKNFThCtYLz56
- From the welcome form, there's a range of experience; I'm interpreting this to mean I should cover conceptual things that might cause confusion.
  - Even if you already know it, it won't hurt to review!

Content Review

Interfaces

- Interfaces define functionality (a set of methods) that are then implemented by classes.
- These methods are not implemented unless the default keyword is used.
- For example, if I have a List, I don't really care about whether I'm using an IntList, a SLList, a DLList, etc. as long as I can add more items, remove items, get items, get the size, etc.
  - These specific things that I should be able to do for any list are the method signatures included in the interface, implemented by classes that can actually do it.
- Classes can implement multiple interfaces.

Examples of Interfaces

- Say I want to get from LA to SF. What do I need to get there?
  - A time to leave
  - A time to arrive
  - Ability to carry items
  - Some degree of comfort
  - Knowledge of the cost
- But, for this example, it doesn't matter what specific implementation I use to get there – plane, car, bus, train, ship, walking, bike, etc. if I know these things.

Class Inheritance (Extends)

- When two classes have similar implementation, then use extends so the subclass can inherit implementations, nested classes, and variables from the superclass.
- For example, say I know I'm travelling by car – I can create one Car class, and then have the specific car extend the Car class.
  - A Ford Mustang and a Chevrolet Camaro both are fairly similar and might be able to use properties from the Car class such as number of wheels, engine location, type of car, etc. with minor differences.
- A class can only extend one other class but can be extended by many classes.
- It's also possible to declare the class abstract and leave some methods abstract (without any implementation).
  - But you can still only extend one class (that includes abstract ones, too!)

When to use what

Interface

- When the implementation of things are different for one kind of behavior, but you want to support that behavior without worrying about the specifics.
- public *List* demoMethod();
- class SLList *implements* List …

Extends

- When classes are very similar with similar implementation.
- class Car *extends* Vehicle …

Overriding and Overloading

- When a class has a method with the same name but different signatures, that is overloading.
  - public int abs(int a)
  - public double abs(int a)
- When a subclass has the exact same signature as the superclass, the subclass method overrides the superclass method.
  - class Vehicle, contains method: public void printInfo()
  - class Car extends Vehicle, contains method: public void printInfo()
- For dynamic method selection (was brought up briefly in lecture but questions haven’t yet come on worksheets), you must check for overridden methods, not overloaded methods.