Objectives

- Garbage collection
- Static methods, variables
- Parameter passing in Java
- Inheritance

Assignment 2 Review

private int oneVar;
public Assign2(int par) {
    oneVar = par;
}

- Is the above code correct?

Code Review: Good Use of switch Statement

public Birthday() {
    int x = random.nextInt(12);
    switch (x) {
        case 1:
            randDay = random.nextInt(29) + 1;
            break;
        case 3:
        case 5:
        case 8:
        case 10:
            randDay = random.nextInt(30) + 1;
            break;
        default:
            randDay = random.nextInt(31) + 1;
            break;
    }
    this.month = months[x];
    this.day = randDay;
}

What does this code do?

Code Review

public static void main(String[] args) {
    Birthday birthday = new Birthday("Sept", 25);
    System.out.println("My birthday is "+birthday.getMonth()+" "+birthday.getDay()+");
}...

public String getMonth() {
    return month + " ";
}

- Discuss this API and how it would be used

Review

- What method should we implement to allow pretty printing of objects?
  ➢ To determine if two objects are equivalent?

Testing

Honey Badger gets bad grade in CSCI209
Double.compare, Float.compare

- Above methods do a comparison of the doubles or floats, respectively

assertEquals(double expected, double actual, double delta)

- Asserts that two doubles or floats are equal to within a positive delta.

More on asserts later...

Memory Management

- In C++ and some other OOP languages, classes have explicit destructor methods that run when an object is no longer used
- Java does not support destructors because it provides automatic garbage collection
  - Waits until there are no references to an object
  - Reclaims memory allocated for the object that is no longer referenced

Do you know what Python does?

Garbage Collector

- Garbage collector is low-priority thread
- Or runs when available memory gets tight
- Before GC can clean up an object, the object may have opened resources
  - Ex: generated temp files or open network connections that should be deleted/closed first
- GC calls object's finalize() method
  - Object’s chance to clean up resources

Discussion: Benefits and costs of garbage collection?

Garbage Collection

Benefits
- Fewer memory leaks
  - Less buggy code
  - But, memory leaks are still possible
  - Code is easier to write

Costs
- Garbage collection may not be as efficient as explicit freeing memory

finalize()

- Inherited from java.lang.Object
- Called before garbage collector sweeps away an object and reclaim its memory
- Should not be used for reclaiming resources
  - i.e., close resources as soon as possible
  - Why?
    - When method is called is not deterministic or consistent
    - Only know it will run sometime before garbage collection
  - Clean up anything that cannot be atomically cleaned up by the garbage collector
    - Close file handles, network connections, database connections, etc.
- Note: no finalizer chaining
  - Must explicitly call parent object’s finalize method
Alternatives to finalize

- Recall: unknown when finalize will execute—or if it will execute
  - Also heavy performance cost
- Solution: create your own terminating method
  - User of class terminates when done using object
- Examples: File's or Window's close method
- May still want a finalize method as a safety net if user didn't call the terminate method
  - Log a warning message so user knows error in code

Static Methods/Fields

- For related functionality/data that isn't specific to any particular object
- `java.lang.Math`
  - No constructor (what does that mean?)
  - Static fields: PI, E
  - Static methods:
    - `static double sin(double a)`

Static Fields

- A static field is used when only one such field per class (not object!)
- All objects of a class share one copy of the static field

Static Methods

- Do not operate on objects
- Cannot access instance fields of their class
- Can access static fields of their class
- Similar to Python functions that are associated with the class

Constant Static Fields

- We used a static field to designate a class constant:
  ```java
  public class Converter {
      public static final double CM2IN = 2.54;
  }
  ```
- The `Math` class has a static constant, PI
  - Value can be accessed using the `Math` class:
    ```java
    area = Math.PI * r * r;
    ```
  - Do not need an object of the `Math` class to use this constant
Static Fields Example

Each Student object has an id field, but there is only one nextID field, shared among all instances of the class.

How could we use the nextID field to create unique IDs?

main()

Most common static method

main() does not operate on any objects

Runs when a program starts...there are no objects yet

main() executes and constructs the objects the program needs and will use

Like the driver function for the program

Analyzing java.lang.String

String toUpperCase()

Converts all of the characters in this String to upper case

static String valueOf(boolean b)

Returns the string representation of the boolean argument

Why can the second method be static?

Static Summary

Static fields and methods are part of a class and not an object

Do not require an object of their class to be created in order to use them

When would we make a method static?

When a method does not have to access an object's state (fields) because all needed data are passed into the method

When a method only needs to access static fields in the class

Review: Class Design/Organization

Fields

Chosen first

Placed at the beginning or end of class definition

Have an access modifier, data type, variable name, and some optional other modifiers

If no access modifier, defaults to package-private

Use this keyword to access the object

Constructors

Methods

Need to declare the return type

Have an access modifier (defaults to package-private if none specified)
PARAMETER PASSING

Method Parameters in Java

- Java always passes parameters into methods by value
  - Methods cannot change the variables used as input parameters
  - A subtle point, so we need to go through several examples

- Python is something that’s not quite pass-by-value—it depends on if the object is mutable or immutable
  - Pass-by-alias is one term used

What’s the Output?

```java
public static void main(String[] args) {
    int x = 10;
    System.out.println("The square of "+ x + " is "+ square(x));
}
public static int square(int num) {
    return num*num;
}
```

Pass by Value: Objects

- Primitive types are a little more obvious
  - Can’t change original variable
- For objects, passing a copy of the parameter looks like

```java
public void methodName(Chicken c)
    methodName(chicken);
```

Pass Chicken object to methodName

```
chicken = new Chicken();
chicken.name = "Fred"
chicken.weight = 3.0

methodName(chicken);
```
Pass by Value: Objects

What happens in this case?

```
public void methodName(Chicken c) {
    if (c.getWeight() < MIN) {
        c.feed();
    }
}
```

```
methodName(chicken);
```

- chicken = c
- height = 38
  - name = "Fred"
- weight = 3.0

Does chicken change in calling method?

**YES!** Both chicken and c are pointing to the same object

What's the Output?

```
Farm farm = new Farm("OldMac");
Chicken sal = new Chicken("Sallie Mae", 50, 10);
farm.feedChicken(sal);
System.out.println(sal.getWeight());
```

// From Farm class
```
void feedChicken(Chicken c) {
    c = new Chicken(c.getName(), c.getWeight(), c.getHeight());
    c.setWeight(c.getWeight() + .5);
}
```

What's the Difference?

```
Farm farm = new Farm("OldMac");
Chicken sal = new Chicken("Sallie Mae", 50, 10);
System.out.println(sal.getWeight());
```

// From Farm class
```
void feedChicken(Chicken c) {
    c = new Chicken(c.getName(), c.getWeight(), c.getHeight());
    c.setWeight(c.getWeight() + .5);
}
```

What's the Difference?
Summary of Method Parameters

• Everything is passed *by value* in Java

• An *object variable* (not an object) is passed into a method
  ➢ Changing the state of an object in a method changes the state of object outside the method
  ➢ Method does not see a copy of the original object

To Do

• Assignment 3b