

Loops

- o Which situations would use a loop?
 - o A game that you play until you decide to quit
 - o A roll of a pair of dice
 - o Playing a list of music
 - o Displaying an image
 - o Drawing an image
 - o Asking a user to input "yes" or "no"

Differences?

- o What are the differences between classes and objects?
 - o <students please answer this>

Classes

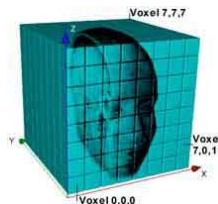
- o Represent templates from which objects are created.
 - o Name some classes from games?

Making Coding Easier?

- o Example from the expressions lab using Beanshell:
 - o <http://www.beanshell.org>

Location in a Game

- o Coordinates are x, y, and z in 3D
- o x and y in 2D



Location: a Class?

- o Is a location just a bunch of variables or does it really deserve to be a class?
 - o One can compare locations
 - o One can translate/rotate locations

Stubs

- o Definition: the methods and returns are all written out with dummy information inside of them. Useful for beginning a class and filling it in.
- o Stubs can be written for any class.

Let's write a Location class

- o Look at the Javadocs up at:
 - o <http://www.cs.rit.edu/~cs1/lectureNotes/location/>
- o **Write just the stubs: 10 minutes**
 - o If the method returns an integer: `return 0;`
 - o If the method returns a string: `return "";`
 - o Cut/paste method comments only if you have time - do comments last.
 - o Test your stub by compiling
- o **When you comment, test javadocs by compiling:**
 - o javadoc Location.java
 - o Bring the index.html file up in a web browser

Fill in the stubs

- o Fill in the constructors
 - o Use this!
- o Fill in the accessors:
 - o These return a variable/value from the class/object
- o Fill in the mutators:
 - o These change a variable in a class/object.
- o Fill in the other method!

Testing and Debugging

- o Testing: the process of finding errors in the system implementation
- o Debugging: the process of finding the source of errors and fixing such errors
 - o Testing is done before debugging

Why Test?

- o The purpose of testing is to identify implementation errors before the product is shipped
- o The errors may be:
 - o Actual bugs in the code
 - o Incorrect implementation of the requirements or functional specifications
 - o Misunderstandings
 - o Incomplete requirements or functional specifications

Why you can't prove there are no errors

```
public int test(int a, int b)
{
    return a / (a+b);
}
```

- o Say that each integer can be a value from 0 to 999
- o In theory, this 3 line function has 1 million logical states, 999,999 of which will work correctly and 1 of which will not
- o "The sun will be a cold hard lump before you can test all states in your program."
 - o — comment from the *Pragmatic Programmer*

What Testing is Not

- o Testing is not a random process
- o Testing is not debugging
 - o Testing identifies the problem
 - o Debugging finds the location of a problem and fixes it.

Write a Test Method

- o It's often convenient to use the main method as a test method in any class
 - o All classes can have main - the one that gets called is the one you type in at the command line prompt
 - o If the class later gets changed then you just add/change the test code

Java's class: Point

- o <http://www.cs.rit.edu/usr/local/jdk/docs/api/index.html>
- o Point has several types of methods in it:
 - o Mutators: methods that change variable values within the Point class
 - o Accessors: methods that access variable values within the Point class
 - o Core methods: translate, move, equals, toString

Useful?

- o What kinds of things are location points helpful in?

Games

- o Performance intensive
- o Often "CPU bound"
- o With managed languages like Java, it becomes important to get the most out of the language
 - o Not all Java implementations of classes are the best implementation for your game

Data Structures

- o We've looked mostly at what we do to data – functions, calculations, etc.
- o But what about the data?
 - o How do we structure data to make it easy to "do things" with it?
 - o How do we structure data so that we get the most performance in our game?

Easy?

- o How can we make it easy/fast to access data?
- o How can we make it easy/quick to put more data in?
- o How do we make it easy/quick to get data out?
- o How do we make it easy/quick to find a specific data element?

The Answer

- o We create different **data structures** for our data
- o You will learn about:
 - o Arrays, Linked Lists, Queues, Stacks, Trees, Hashing, Graphs, etc.
 - o Maybe even BSP trees

Arrays

- o Good:
 - o for *random access*
 - o when you know what you want and where it exists
- o Bad:
 - o If you need to reorder things to be in a certain order often
 - o Capacity of the structure changes often

Array Example: Command Line Input

```
class UnixAcctGen {
  public static void main(String[] args) {
    if (args.length != 4) {
      System.out.println("Please enter first name, " +
        "middle name, last name, and SS# to run " +
        "the program.");
    } else {
      String firstInitial = args[0].substring(0,1);
      String middleInitial = args[1].substring(0,1);
      String lastInitial = args[2].substring(0,1);
      String lastFourSS= args[3].substring(args[3].length()-4, args[3].length());
      System.out.println(firstInitial + middleInitial + lastInitial + lastFourSS );
    }
  }
}
```

Design and Test

- o Coffee.java
 - o a small program with problems
- o At:
<http://www.cs.rit.edu/~cs1/lectureNotes/Coffee.java>

Real Programmers Refactor

- o When you take a program and redesign ruthlessly to make future maintenance easier.
- o Why does Coffee.java need to be refactored?

Reduce Redundancy

- o How do we reduce redundancy?
 - o What happens if we change the price of Coffee?
- o How do we split the program up to make it more flexible?

- o NOTE: This could be combat stats for characters instead of the price of coffee.

One way of refactoring

- o <http://www.cs.rit.edu/~cs1/CoffeeExtra.java>

Memory diagrams?

- o Can you draw a memory diagram for the Reference and Basic types here:

```
class Test {  
    public static void main(String[] args) {  
        char h = 'h';  
        char h2 = 'h';  
        char h3 = 'H';  
        String h4 = new String("Hello");  
        String h5 = h;  
        String h6 = new String("Hello");  
        String h7 = "Hello";  
        String h8 = "Hello";  
    }  
}
```

Pass-by-value

- o Basic types are passed into a method by their value
 - o This means that the method receives a *copy* of the variable.

- o Can the method change the value of the original variable?
 - o No! It can only change the copy.

Pass-by-value for references

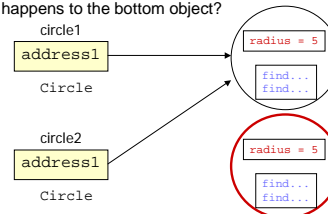
- o Reference variables are passed by their reference or address
 - o This means that the method gets a copy of the address to the object

- o Can the address change within the method?
 - o No! Only the copy can change.

- o Can the object itself change?
 - o Yes (if the object is not final/immutable)!

Garbage Collection

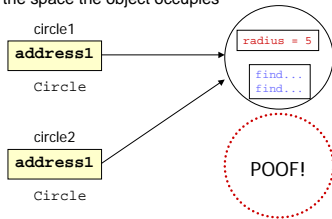
- o What happens to the bottom object?



- o Objects occupy memory, so there must be a way to free the space this object occupies if it is not used anymore

Garbage Collection

- o When there are no references to an object, that object is free to be destroyed
- o The Java runtime system detects garbage and automatically reclaims the space the object occupies



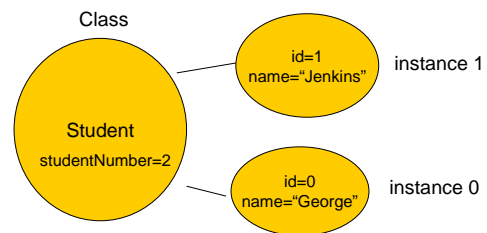
Static vs. Instance

- o Methods and variables declared static mean that no matter how many objects are created, only one method/variable will exist for the class
- o Instance methods/variables are NOT declared static and each object will get its own copy of the method/variable

Example

```
public class Student {  
    private static int studentNumber = 0;  
    private int id = 0;  
    private String name="";  
    public Student(String name) {  
        id = studentNumber;  
        studentNumber++;  
        this.name = name;  
    }  
    public Student() {  
        this("Bob");  
    }  
    public int getId() {  
        return id;  
    }  
}
```

What this looks like



this

- o Can be used with instance items:
 - o this.name refers to the instance version of name rather than the local copy in the class above
- o For constructor use:
 - o this("Bob");
 - o The above line would refer to the instance constructor taking a String as a parameter

static?

- o Cannot use the keyword "this" with static variables/methods.
- o Instead, refer to them by class name since they exist only at the class level:
 - o Math.PI
 - o Math.abs(-5)

Java Array Example

o String class