**Java Week 3 Notes**

* Programs were procedural
* GUI
  + Great for users
  + How do we write a program that runs from top to bottom that can handle all different events user will experience
* OOP
  + Object oriented programming
  + Most powerful, scalable, extensible model for programming that exists today
  + Solves our GUI problem and provides a way to reuse code from project to project
  + Objects are instances of classes
  + Classes exist at Design time
  + Objects exist at runtime
  + JVM uses class to create object
  + Building an object from a class is called instantiating
  + Methods let our objects do work
  + Methods let us break big problems into smaller chunks
  + Method headers always have brackets after method name
* Random
  + Not primitive
  + Used to instantiate an object
  + We must import the java.util.Random, this is a class that is a blueprint for us to instantiate an object
  + Random rnd = new Random(), creates new object, objects stored in the heap
  + Rnd holds the address of the object, its a local variable, but also a reference variable as it references an object
  + Int num = rnd.nextInt(100); ←, uses next int method to retrieve a random num from the object
* Primitive vars store values
* Reference variables refer to the address of an object in the heap
* Variables
  + Always either Local or Instance
  + Always either Primitive or Reference
* Syntactic Sugar
  + Shorter, easier to read and write syntax for certain lines of code
* Abstraction
  + When we narrow our perception of something down to just the things we need
* Variables have scope
  + Scope: part of code where the variable exists
  + A variable has scope within the code block it was declared in
  + Variables declared in a method are local
    - Stored in the stack frame
* Instance Variables
  + Declared inside the class
  + Has scope across entire class
  + Stored in the object
* Arguments of a method are local variables, not instance
* Void methods
  + If methods are void, information can be passed through them, but you cant return information
* Non void methods
  + Can return values
  + Must have a header including the data type it returns
  + Ex: public String translateToPirate(String phrase) {
* Paramaters are catching values sent to the method
* Arguments are the values you are sending to the method