Terms that came up in Unit 4 – we refer to Monkey class on back.

class – an idea of something. A template for a data type. A data-type.
Monkey is a data-type. (a class file is never executed)

object – the physical manifestation of a class. If we construct a Monkey like so :
Monkey bob = new Monkey(“Bob”); bob is an object of type Monkey.
The header of a constructor : public class-name (parameter list)

instance variables – the state or present condition of the object is hekd in its instance Variables. The defining characteristics of a class or object. PRIVATE!

instance – an object created. bob from above is an instance of type Monkey.
it is a creation of type Monkey.

constructor – what you call to create an object. Type variable name = new Type(args)
Monkey bob = new Monkey(“Bob”); ← this is a call to a 1-arg constructor

static – refers to any method or variable not called using an object. The variable or method belongs to the class.

instance method – a method created within a class to be called by an instance of that class. bob.getName() calls the instance method getName. An object must be created or instantiated prior to calling any of its methods. (behavior)

state – the instance variables hold the present state of an object. So a monkey object has a name, a number of toys, a message, etc. what they contain at any given moment is the Monkey object’s state.

behavior – the methods that can be called for an object. You define any things that the object can do or have done to it via its methods.

mutator – a mutator method refers to an instance method that can change the state of the objects instance variables. No = sign is needed. Void is type!
bob.addToys(3) will add three toys to the current numberOfToys.

accessor – an accessor method refers to an instance method that <u>returns a certain data type</u>, usually an instance variable which is part of the objects state.
boolean h = bob.isHungry() ← need = sign to retrieve returned variable!

encapsulation – the computer science principle that says we will not allow access to the state of an object except via the public methods. (So all instance variables should be private to uphold encapsulation) visibility = private!

overloaded method – the ability to define more than one method with the same name (especially since it performs the same basic function) As long as the signature(parameter type & number of parameters) is different then the JVM knows which method to call.

overloaded constructor – the ability to have multiple constructors (remember constructors always have the same name as the class but don’t have a type). The number of parameters or types may vary.

no-args constructor- a constructor with no parameter list. This means that the JVM will be giving default values to each instance variable or you may have defined basic values for this constructor. If and only if no constructor is provided by the class, java creates a no-args constructor for the class.

```
public class Monkey
{
   private String mName;
   private String message;
   private int numToys;
   private boolean isHungry;
   private int numBananas;


   public Monkey(String name)
   {
      mName=name;
      isHungry=true;    //monkeys are always hungry!!
   }


   public Monkey(int numT, String mName) {
      this.mName=mName;
      isHungry = true;
      numToys = numT;
      message = "ooh ooh ooh oooh ooh!";
   }


   public void setMessage(String msg)
   {
      message = msg;
      //note: if I put msg=msg; then the instance veriable is
never set
      //msg which is a local variable is set to itself. nothing
happens!!
   }


   public void speak() {
     System.out.println(message);
   }


   public void addToys(int n) {
     numToys += n;
     //note: we did not set the numToys instance variable
but the JVM will
     //default int variables to 0. win-win!
   }


   public boolean isHungry( ) {
     if (numBananas > 5)
       isHungry = false;
     else
       isHungry = true;

     return isHungry;

   }
public void eatBananas(int n){
      // note: total is a local variable. only accessed in this
method!
      int total = n + numBananas;
      numBananas = total;
   }

   public String getMsg() {
      return message;
   }

   public String getName() {
     return mName;
   }

     public int getBananas(){
               return numBananas;
     }

   public int howManyToys( ) {
     return numToys;

   }



   public String toString() {
      return ("Monkey: "+getName()+"\nHungry?
"+isHungry()+"\nHas how many toys: "+
      howManyToys()+"\nMessage: "+getMsg()+"\n");
   }
}
```