CS 125 - Lecture 21

Objectives: Intro objects; objects as parameters in subroutines.

Up next: MP4 due 8PM tonight. MP5 out tomorrow - due in 14 days

1. **Objects in your life? Examples:**

Tangible…

Abstract…

What characteristics do objects have in common?

5. **Merge two sorted integer arrays together into a single output array**

```
.public static int[] merge(int[] A, int[] B) {
   int done = 0, countA = 0, countB = 0;
   int[] result = new int[______________________];
   while ((countA < A.length) ____ ___________________) {
      if (________________]) result[done++] = A[ countA++];
      else
         result[______] = B[ _________];
   }
while (countA < A.length)
   result[done++] = A[countA++];
```

2. Classes can be thought of as:

Templates or blueprints or factories for objects

3. **Instantiation: constructing an instance of a class. There's a lot going on here!**

```
Building dcl = new Building();
```

Declaration statement (left) and assignment statement (right), combined

Declare `dcl` object of type `Building` class. `dcl` is a _______________ variable.

Evaluate the right-hand side: construct an object/instance of type Building

- Allocate memory for the object
- Initialize instance variables - default values or
- Execute constructor method on Building class
- return memory address of object in heap

Assign `dcl` the reference (memory address) to the object

4. **Objects as parameters in subroutines.** 2D Array Demo:

```
public class ArrayPassingDemo {

   public static void main(String[] args) {
        // test the array passing in here...
   }

   public static void debug2DArrayInt(int[][] ptr) {
      // Handle the case if the reference is null
      if (ptr == null) {
         System.out.println("Ooops! Array is null");
         return;
      }

      // Handle the case if the reference points to an array
      System.out.print("{ ");
      for (int i = 0; i < ptr.length; i++) {
         System.out.print("{ ");
         for (int j = 0; j < ptr[i].length; j++) {
            if (j > 0) System.out.print(", ");
            System.out.print(ptr[i][j]);
         }  // end of i-loop
         System.out.print(" } ");
      }  // end of j-loop
      System.out.println(" }");
   } // end of class method debug2DArrayInt
}
```

6. Me: **Create a Label class, instances of the class, and diagnostics…**

```
public class Label {
   String text;
   int x,y;
   public void setText(String n) { this.text = n; }
   public void printMe() { System.out.println(this.text); }

   public static void main (String[] args) {
      Label labelptr1 = new Label ();
      labelptr1.setText("About");
      labelptr1.printMe();

      Label labelptr2 = new Label ();
      labelptr2.setText("Bass");
      labelptr2.printMe();

      Label labelptr3 = new Label ();
      labelptr3.setText("Treble");
      labelptr2.printMe();
   }
}
```

7. You: **Create a Bird class (of the 'Flappy' variety) and instance methods to initialize it. Use an instance method to make it move one unit to the right.**

8. **Complete the boolean instance method below:**

```
Dog d1 = new Dog();
Dog d2 = new Dog();

d1.x = 10; d1.y = 20;
// d2.x = ..., d2.y = ...

class Dog {
   int x,y;

   // returns true if this dog is at same
   // x,y location as the other dog
   public boolean canSniffButt(Dog other) {

      return ____________________________

   }
}
```

9. **Create the House class so that the following code compiles and runs correctly.**

```
House h = new House();
      h.setStreet("101 Main St");

      // returns true iff houses have same street string
      h.equals(otherHouse);
```

4. **Bucket sorting: explain the algorithm to your neighbor. Need a refresher?**

- Set up an array of initially empty "buckets".
- Scatter: Go over the original array, putting each object in its bucket.
- Sort each non-empty bucket.
- Gather: Visit the buckets in order and put all elements back into the original array.

7. You: **Create a Bird class (of the 'Flappy' variety) and instance methods to initialize it. Use an instance method to make it move one unit to the right.**

8. **Complete the boolean instance method below:**

```
Dog d1 = new Dog();
Dog d2 = new Dog();

d1.x = 10; d1.y = 20;
// d2.x = ..., d2.y = ...

class Dog {
   int x,y;

   // returns true if this dog is at same
   // x,y location as the other dog
   public boolean canSniffButt(Dog other) {

      return ____________________________

   }
}
```

9. **Create the House class so that the following code compiles and runs correctly.**

```
House h = new House();
      h.setStreet("101 Main St”);

      // returns true iff houses have same street string
      h.equals(otherHouse);
```