Classes, Arrays, Lists, and Iterators

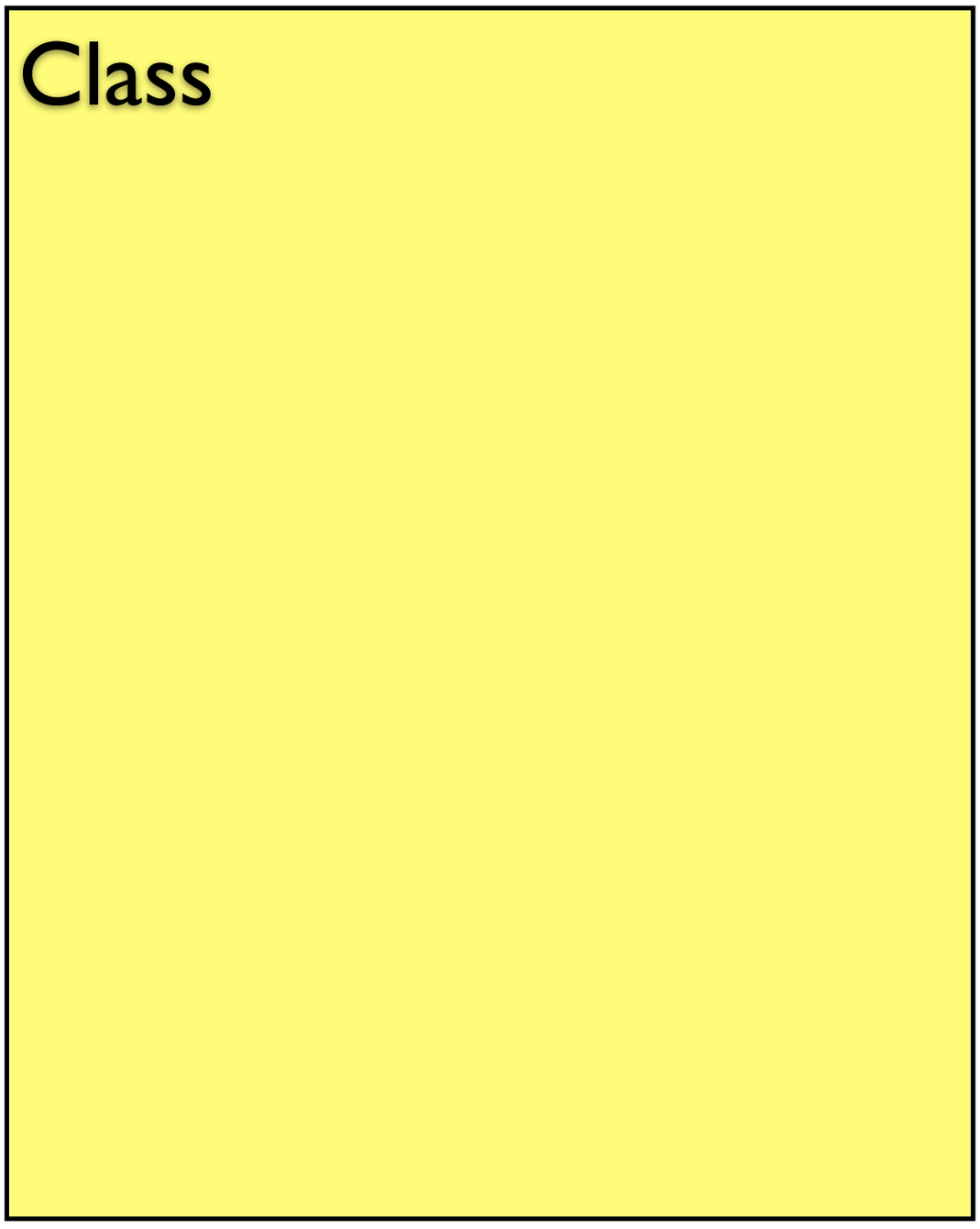
September, 7 2012

At the end of class

- You should know about:
  - Classes and objects
  - Primitives and objects
  - Cloning
  - Iterators

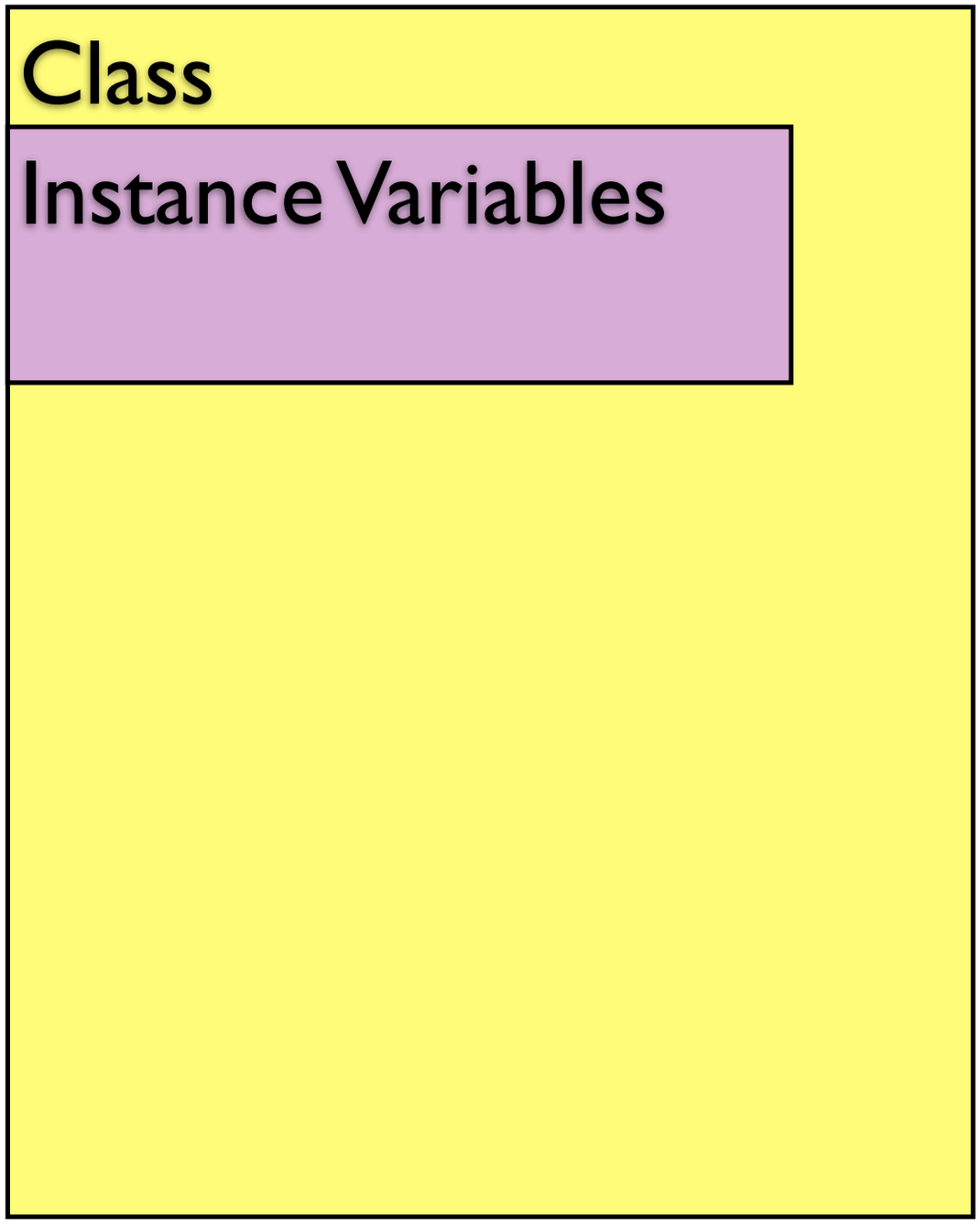
Class

- Blueprint for individual object



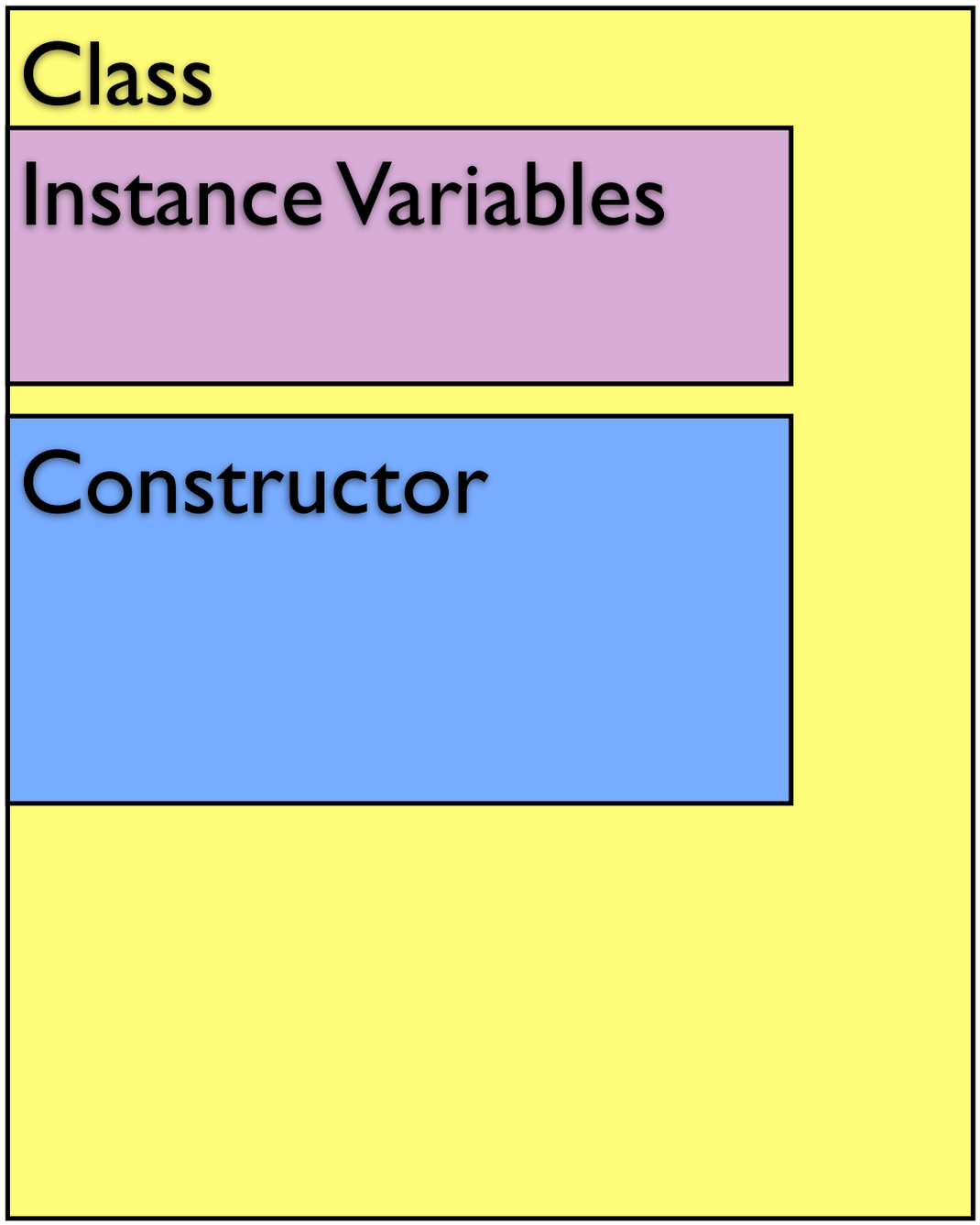
Class

- Blueprint for individual object



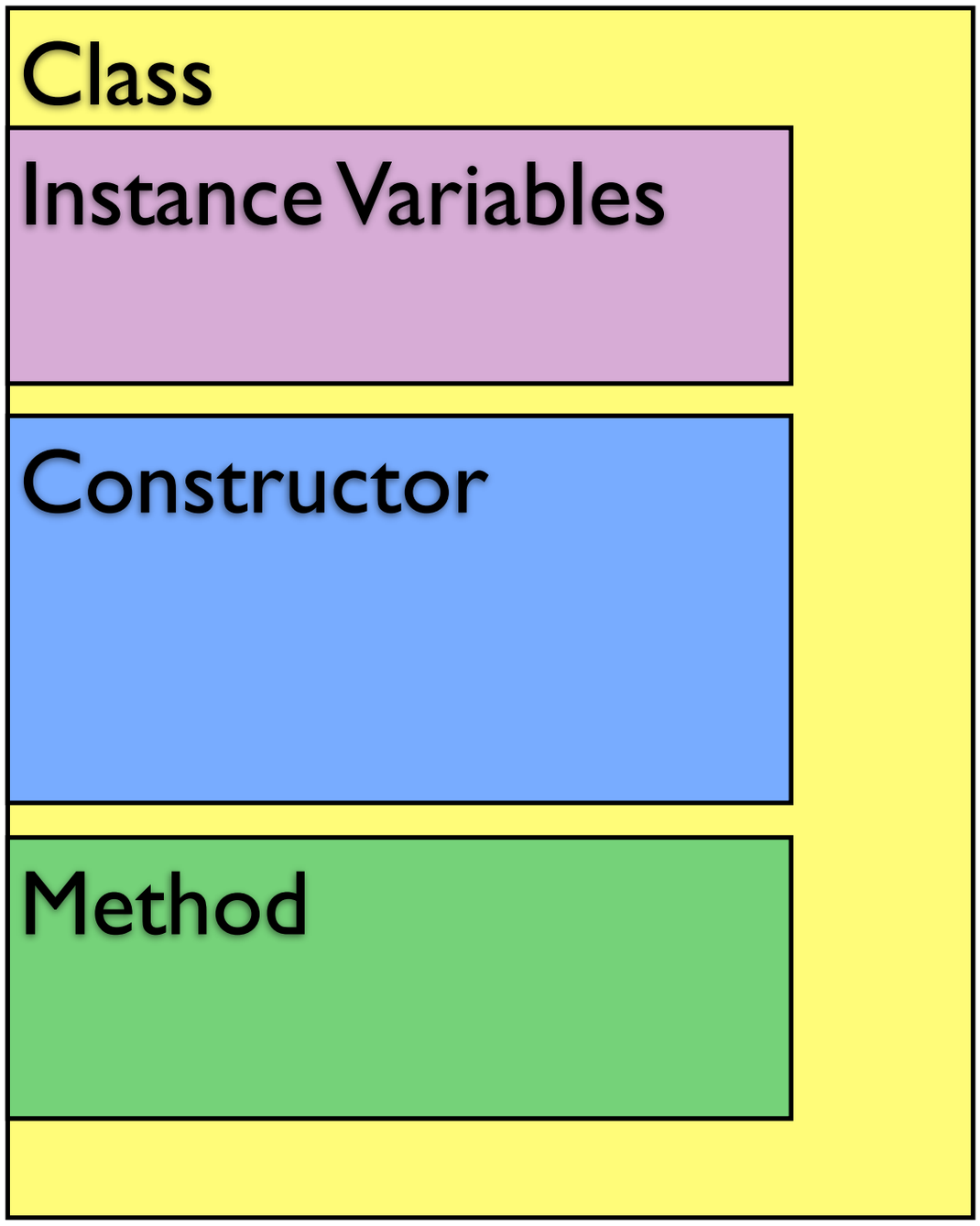
Class

- Blueprint for individual object



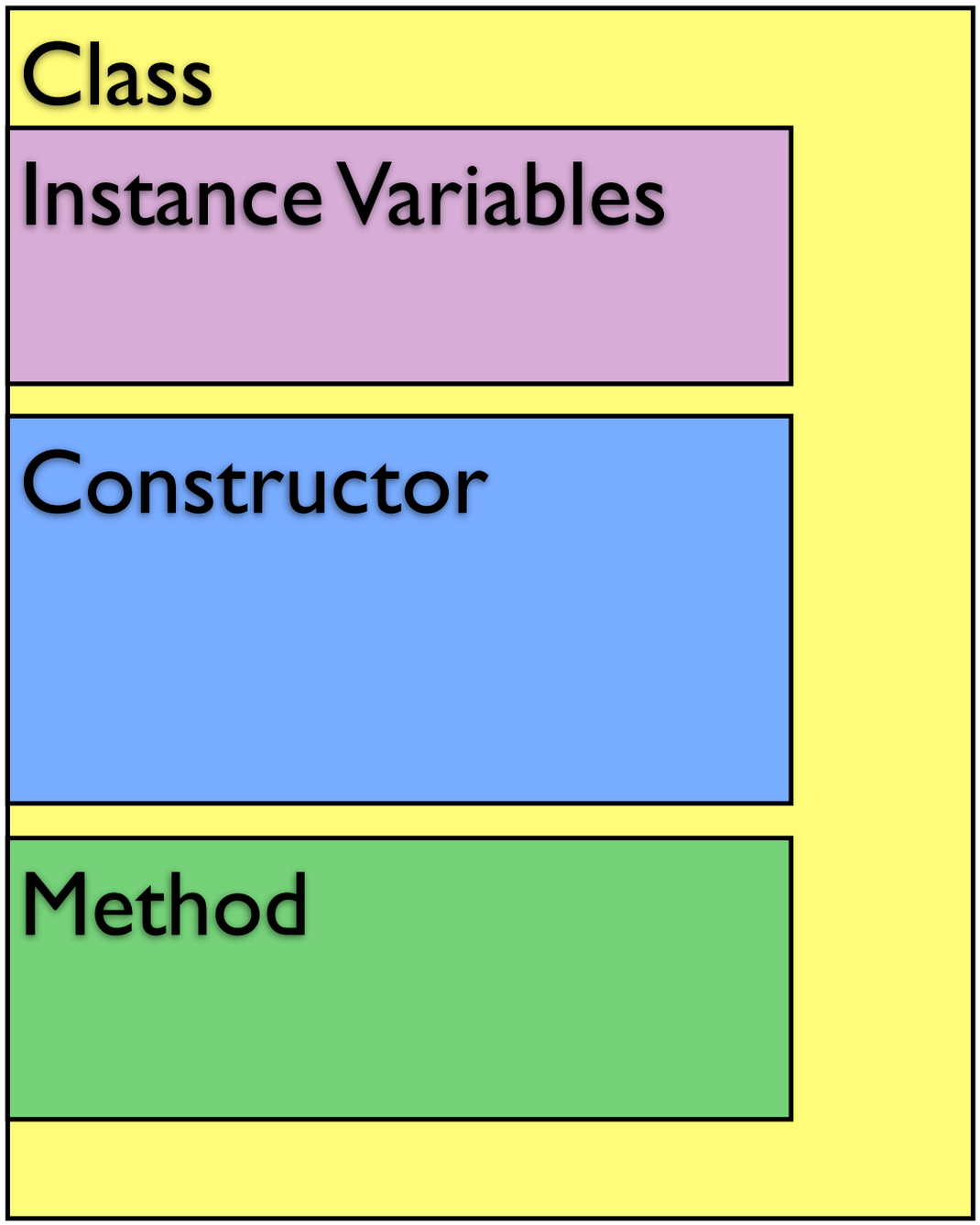
Class

- Blueprint for individual object



Class

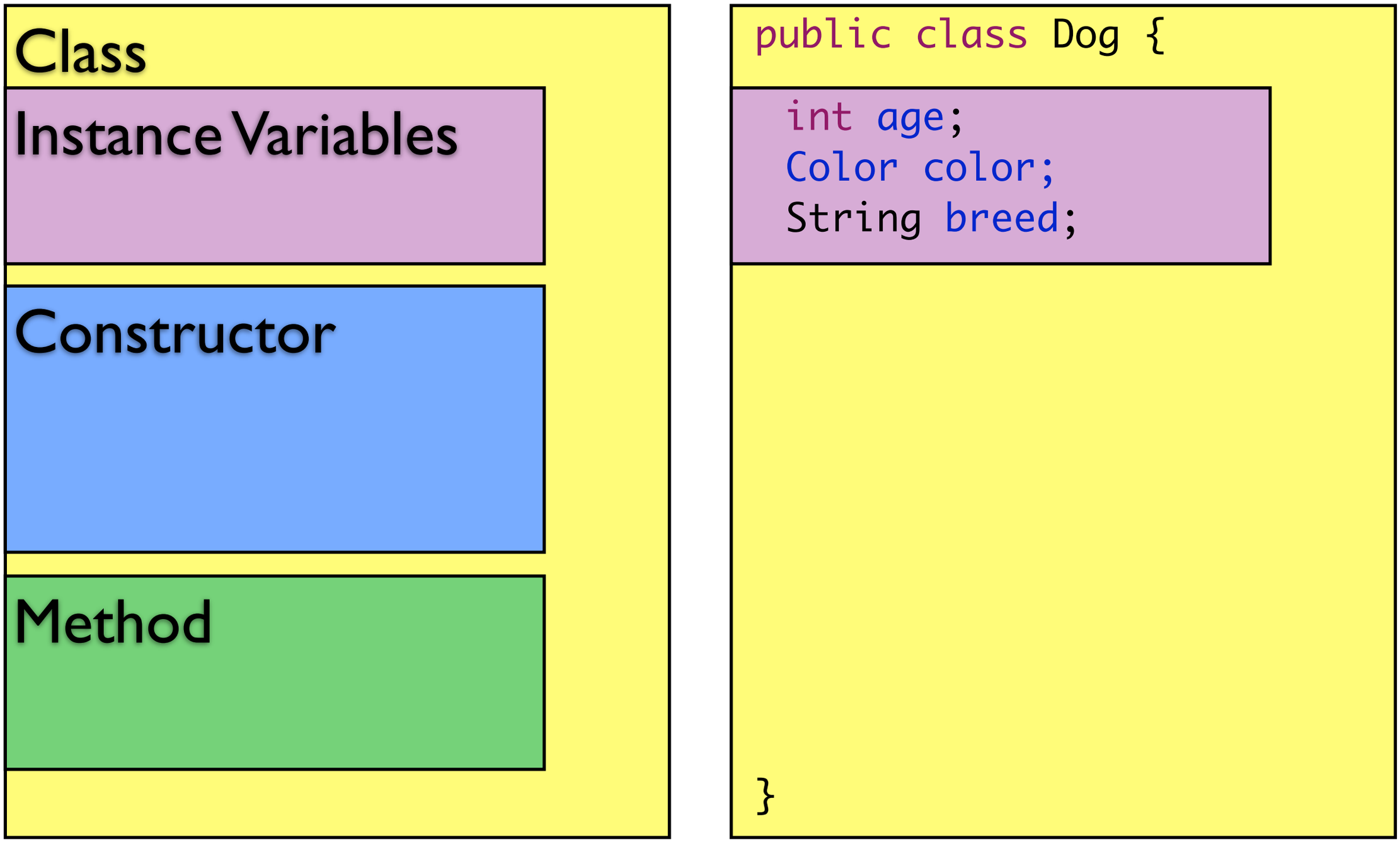
- Blueprint for individual object




```
public class Dog {

}
```

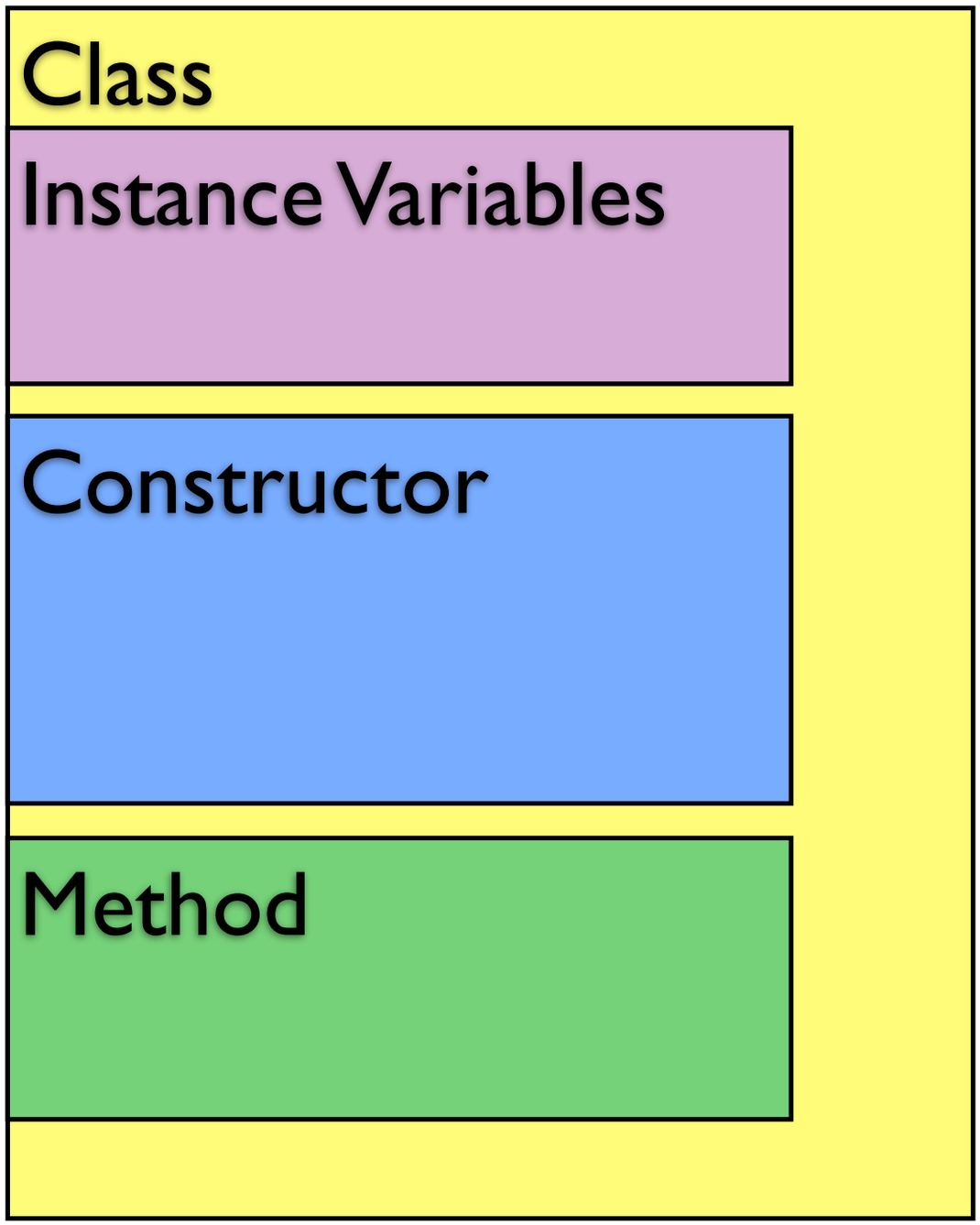
Class

- Blueprint for individual object



Class

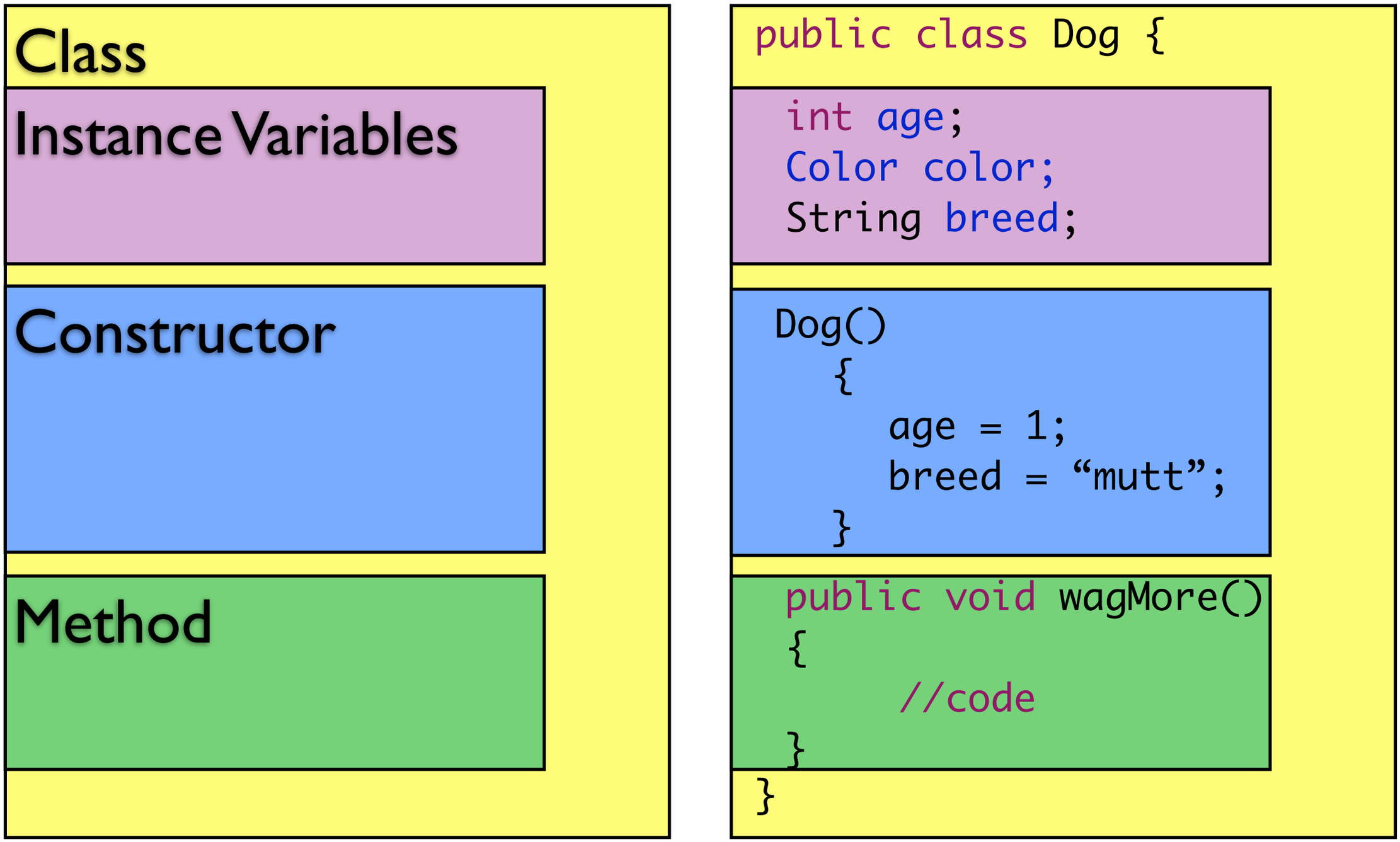
- Blueprint for individual object




```
public class Dog {

  int age;
  Color color;
  String breed;

  Dog()
     {
        age = 1;
        breed = “mutt”;
     }

}
```

Class

- Blueprint for individual object



Object

- an instance of a class

```
public static void main(String[] args)
{
      Dog myDog = new Dog();

      myDog.setAge(6);

      myDog.setName(“Mad Max”);

      myDog.setColor(Color.orange);

      myDog.wagMore();



 }
```

```
public class Dog {

   int age;
   Color color;
   String breed;

   Dog()
      {
         age = 1;
         breed = “mutt”;
      }

   public void wagMore()
   {
         //code
   }
}
```

Object

- an instance of a class

```
public static void main(String[] args)
{
      Dog myDog = new Dog();

      myDog.setAge(6);

      myDog.setName(“Mad Max”);

      myDog.setColor(Color.orange);

      myDog.wagMore();

}
```

```
public class Dog {

   int age;
   Color color;
   String breed;

   Dog()
      {
         age = 1;
         breed = “mutt”;
      }

   public void wagMore()
   {
         //code
   }
}
```

Primitives vs. Objects

```
String myString = new String("hello");
String yourString = new String("hello");

      if(myString == yourString)
      {
         System.out.println("hello == hello");
      }
       else
         System.out.println("hello != hello");
```

What is the output from the code?

A. “hello == hello”

B. “hello != hello”

Primitives vs. Objects

- Primitives (int, float, boolean, char)
  - Store the value in memory
- Objects
  - Store the location!

  - (This is known as a pointer)

How to compare objects

- myString.equals(yourString)

- don't use "==" because you will be comparing memory addresses!

Clone

- How to make a copy of an object
  - DON'T
    - newObject = oldObject
  - DO
    - newObject = oldObject.clone();

Arrays

- int [] anArray = new int [10];

- fixed number of elements
- single type
- each element is accessed by index
  - 0 to (n-1)

ArrayList

- ArrayList<type> myList = new ArrayList<type>();

- resizable array
  - myList.add(something);
  - myList.remove(something);

Iterator

- Iterator<String> itr = myList.listIterator();

```
while(itr.hasNext())
{
   System.out.println(itr.next());
}
```

Let's practice

- Snarf the code
  - SimpleSort.java in Recitation 2
- Get coding
- Help your friends
- Submit your code at the end of class
  - Even if you are not done!

It is the end of class

- You should know about:
  - Classes and objects
  - Primitives and objects
  - Cloning
  - Iterators