

The background of the slide is a grayscale image of a circuit board. It features a complex network of black lines representing traces and several solid black circles representing vias or components. The circuitry is arranged in a somewhat symmetrical, horizontal pattern. A solid black horizontal band runs across the middle of the image, partially obscuring the circuit board design. In the center of this dark band, the text 'CMSC 131' is displayed in a large, white, sans-serif font. Below it, the text 'Fall 2018' is written in a smaller, green, monospace-style font.

# CMSC 131

Fall 2018

# Announcements

- Project #5 has been posted

# Mutability

What does it mean for a class to be **mutable**?

**Immutable**?

Can we look at a class and tell?

Always document whether your class is mutable or immutable!

Why is immutable “preferred”?

# String vs. StringBuffer

Strings are immutable.

- What's good about this?
- What's bad about this?

There is another class, StringBuffer, that is mutable.

- What's good about this?
- What's bad about this?

Example: [StringBufferExample.java](#)

# Making Copies

Take a look at the class `Pirate.java`.

Let's draw memory diagrams for two kinds of copies:

- Shallow Copy
- Deep Copy

Now consider some code:

```
1. Pirate x = new Pirate();
```

```
    Pirate y = x;
```

“reference” copy

```
2. ... (next slide)
```

# Deep Copy vs. Shallow Copy

```
2. Pirate x = new Pirate();  
   Pirate y = new Pirate(x);
```

- Is this deep or shallow?
- How could we implement it each way?
- Which way is better?  
(Depends on whether or not eyepatch and parrot are mutable or immutable. Why?)
- What if one is mutable and the other is immutable?  
(Hybrid is best.)

# Privacy Leaks

- Assume that the bird class is **mutable**. (Perhaps there is an instance method called “changeColor”.)
- Assume that we don’t want the state of the bird changed by anyone other than the Pirate who has the bird.

Consider the getter (getBird) of the Pirate class. Any problems?

This is called a “privacy leak”.  
How can we fix it?

Assume that the EyePatch is **immutable**.  
Is the getPatch method OK?

## Complex Example

Take a look at the CD and RewritableCD classes.  
What property distinguishes them?

Take a look at the two getters in the CDOwner class.  
Let's draw memory diagrams.  
Does either generate a memory leak?  
Which one is best?

Now let's do the same analysis for the  
RewritableCDOwner class.