



# CMSC 131

Fall 2018

# Arrays of References

Suppose I have a class called Cat and I want to store a sequence of Cats.

Let's draw the memory diagram for this:

```
Cat[] x;
```

```
x = new Cat[4];    // how many Cats have I  
made?
```

```
x[0] = new Cat("Fluffy");
```

```
x[1] = new Cat("Princess");
```

```
x[2] = new Cat("Spot");
```

```
x[3] = new Cat("Steve");
```

# Crazy Example

Creating something complex from something simple.

Example: Word.Java, Sentence.Java,  
Paragraph.java, Driver.java

## Initializing an array When Constructed

```
char[] arr = {'x', '@', 'A', '!'};
```

```
double[] values = {3.1, 62.79, 5.88, 6.1, 7.55};
```

```
Cat[] kitties = {new Cat("Felix"), new Cat("Tom"),  
                 new Cat("Sylvia"), new Cat("Oscar")};
```

# 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 `CopiesOfArraysExamples.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.)