This lecture borrows heavily from Joel Castellanos’s CS-152 lecture on arrays.
An array is a data structure

In Java, an **array** is a collection of data items allocated in sequential memory locations that can be selected by indices computed at run-time.

```java
int[] a = {1, 1, 2, 3, 5};
int[] b = new int[10000000];
b[32767] = 9;
System.out.println(a[4]);
System.out.println(b[100]);
System.out.println(b[32767]);
```
Arrays allow constant time access and storage.

```java
int[] a = {1, 1, 2, 3, 5};
int[] b = new int[10000000];
b[32767] = 9;
System.out.println(a[4]);
System.out.println(b[100]);
System.out.println(b[32767]);
```

No matter how large an array, a particular element can be loaded or stored in constant time.
A world without arrays

String student0 = "Sam";
String student1 = "Jesse";
String student2 = "Mike";
String student3 = "Claire";
String student4 = "Teresa";
System.out.println(student0 +", " + student1 +
", " + student2 +", " + student3 +", " + student4);
With arrays

```java
String[] students = {"Sam","Jesse", "Mike",
                   "Claire", "Teresa"};
for(int i=0; i<students.length; i++){
    System.out.print(students[i]+", ");
}
```
Fibonacci series without arrays

```java
//Example without an array
int f0 = 1;
int f1 = 1;
int f2 = f1 + f0;
int f3 = f2 + f1;
int f4 = f3 + f2;
int f5 = f4 + f3;
int f6 = f5 + f4;

System.out.println(f0 + " " + f1 + " " + f2 + " " + f3 + " " + f4 + " " + f5 + " " + f6);

//1 1 2 3 5 8 13
```
Fibonacci series with arrays

//Initialize the array.
int[] f = new int[14];
f[0] = 1;
f[1] = 1;

//Populate the array with values
for (int i=2; i<f.length; i++){
    f[i] = f[i-1] + f[i-2];
}

//Print out the contents of the array
for (int i=0; i<f.length; i++){
    System.out.print(f[i] + " ");
}
System.out.println();
//1 1 2 3 5 8 13 21 34 55 89 144 233 377
int[] f = new int[14];

int[] g;
g = new int[14];

int[] f declares f as a reference to an int array. f = new int[14] reserves memory for 14 int variables and sets f to reference that block of memory.
Fibonacci series with arrays

```java
//Populate the array with values
for (int i=2; i<f.length; i++){
    f[i] = f[i-1] + f[i-2];
}
```

In Java, arrays have a field .length not a method .length(). By contrast, a String has a method .length() not a field .length.
Strings are variable length.

```java
String t = "hello";
t = t + " world";
System.out.println(t);
```

Arrays are fixed length.
Quiz

In the Java code below, which lines can be swapped (have their positions interchanged) without causing an error or changing the output?

```java
1) int [] f = new int [16];
2) f [0] = 1;
3) f [1] = 1;
4) for (int i=2; i<f.length; i++){
5)   System.out.println(f[i-2]);
6)   f[i] = f[i-1] + f[i-2];
7) }
```
Quiz

1) `int[] f = new int[16];`
2) `f[0] = 1;`
3) `f[1] = 1;`
4) `for (int i=2; i<f.length; i++){`
5) `System.out.println(f[i-2]);`
6) `f[i] = f[i-1] + f[i-2];`
7) `}

(a) 1 and 2 3 and 4
(b) 4 and 5 4 and 6
(c) 6 and 7
printArray

Let’s write a method that prints the contents of an array.
Go to Eclipse.
getMax

Let’s write a method that returns the maximum value stored in an int array. Go to Eclipse.
Copying by reference versus copying by value

CopyReferenceValue.java
Object equality

ObjectEquality.java