Assignment 2 is due Monday at 11:59pm

Reading for next lecture is the remainder of Chapter 7 and all of Chapter 8

int[] A;       // declaration
A = new int[5]; // memory allocation

int[] arrayB = new int[5]; // both
char[] lettersArray = new char[5];
String[] words = new String[3];

int[] arrayC = {1, 2, 3, 4, 5}; // +initialization
char[] letterGrades = {'A', 'B', 'C', 'D', 'F'};
String[] wordArray = {"CS230", "Data", "Struct"};
- Arrays are an **indexed** and **mutable** collection.
  - We can directly access and change an element at *any* index.
- Arrays are **homogeneous** collections.
  - All the elements of a Java array must have the same type.
- Arrays have a **fixed length**.
  - Once an array is created, its length cannot be changed.
- For array `a` its length is given by `a.length`

```java
int[] arrayB = new int[5];
for (int i = 0; i < 5; i++) {
    arrayB[i] = 2*i;
}
```

```java
int[] arrayC = {1, 2, 3, 4, 5};
for (int i = 0; i < arrayC.length; i++) {
    arrayC[i]++;
    System.out.println(arrayC[i]);
}
```

```java
String[] wordArray = {"CS230", "Data", "Struct"};
wordArray[1] = "Silly ";
System.out.println(wordArray[1] + wordArray[2]);
```

*Given a sentence (as a String), how do you count how many times each letter appears?*

* (Hint: You can access any char using `charAt(i)`)

* Where would you keep track of all the counters?*
When you manipulate an array, you access it through a reference!

What happens here?
arrayA = arrayB;

What is printed here?
int[] arr1 = {1, 2, 3, 4, 5};
int[] arr2 = {1, 2, 3, 4, 5};
if (arr1 == arr2)
    System.out.println("same");
else
    System.out.println("different");

How do we copy the contents of arrayA into arrayB?
How do we check if two arrays contain the same info?

- The elements of an array can be object references. E.g. references to String objects
  String[] words = new String[5];
- Initially an array of objects holds null references
  System.out.println(words[0]);
- At this point, the above line would throw a NullPointerException
- Each object must be instantiated separately
  words[1] = "loyalty";
  words[0] = "friendship";
  words[2] = "honor";
The `String[] args` input parameter in the `main()` method is Java's way to communicate with the outside world at the time of invocation.

The arguments to the `main()` method are called command-line arguments and are provided when an application is run.

```java
public class PlayGame {
    public static void main(String[] args) {
        String player1 = args[0];
        String player2 = args[1];
        System.out.print("Welcome to the game ");
        System.out.println(player1 + " and " + player2);
    }
}
```

```
>java PlayGame Jack Jill
```

An entire array can be passed as a parameter to a method.

Like any other object, the `reference` to the array is passed, making the formal and actual parameters aliases of each other.

Therefore, changing an array element within the method changes the original (called “by reference”).

This can also be a source of errors – be careful!

```java
public static int sumElements (int[] numArray) {
    int sum = 0;
    for (int i = 0; i<numArray.length; i++)
        sum = sum + numArray[i];
    return sum;
}
```

```java
public static int[] createNumArray (int size) {
    int[] newArray = new int[size];
    for (int i = 0; i<size; i++)
        newArray[i] = i;
    return newArray;
}
```

```java
int[] myData = {1, 2, 3, 4, 5};
int result = sumElements(myData);
```

```java
int[] arrayC = createNumArray(20);
```
A UML diagram for the Tunes program

```java
/**
 * Demonstrates the use of an array of objects
 * @author Java Foundations
 */
public class Tunes {

  // toString method omitted

  public static void main (String[] args) {
    CDCollection music = new CDCollection();
    music.addCD("Storm Front", "Billy Joel", 14.95, 10);
    music.addCD("Come On Over", "Shania Twain", 14.95, 16);
    music.addCD("Soundtrack", "Les Miserables", 17.95, 33);
    music.addCD("Graceland", "Paul Simon", 13.90, 11);
    System.out.println (music);
    music.addCD("Double Live", "Garth Brooks", 19.99, 26);
    music.addCD("Greatest Hits", "Jimmy Buffet", 15.95, 13);
    System.out.println (music);
  }
}
```

```java
import java.text.NumberFormat;
/**
 * Represents a compact disc (an ancient technology for recording sound)
 * @author Java Foundations
 */
public class CD {
  private String title, artist;
  private double cost;
  private int tracks;

  /**
   * Create a new CD with the specified information
   * @param name album title
   * @param singer artist name
   * @param price cost of CD in floating point
   * @param numTracks number of tracks on CD
   */
  public CD (String name, String singer, double price, int numTracks) {
    title = name;
    artist = singer;
    cost = price;
    tracks = numTracks;
  }

  toString() method omitted...
```
A two-dimensional array can be thought of as a table of elements, with rows and columns.

In Java a two-dimensional array is an array of arrays.

It is declared by specifying the size of each dimension separately:
```
int[][] scores = new int[12][50];
```

An array element is referenced using two index values:
```
value = scores[3][6]
```

The array stored in any one row can be specified using one index!
Demonstrates the use of a two-dimensional array.
* Author: JAVA Foundations

```java
public class TwoArray {
    /**
     * Creates a 2D array of integers, fills it with increasing
     * integer values, then prints them out.
     *
     */
    public static void main(String[] args) {
        int[][] table = new int[5][10];
        // Load the table with values
        for (int row=0; row < table.length; row++)
            for (int col=0; col < table[row].length; col++)
                table[row][col] = row * 10 + col;

        // Print the table
        for (int row=0; row < table.length; row++) {
            for (int col=0; col < table[row].length; col++)
                System.out.print(table[row][col] + " ");
            System.out.println();
        }
    }
}
```