Passing Arrays
Ch 7
Array review

Classes store information about a similar abstract idea (e.g. BankAccount's name and balance)

Arrays store information of similar type (e.g. `char` arrays is how we've been thinking about strings)

Both of these are actually stored as references
Array review

```
double[] x = new double[10];
```

Type, [] means array

variable name

length of array

Alternative declaration-initialization:

```
int[] x = {1, 2};
```

(See: Max.java)
Each element of an array is the same as an object of that type

For example:

```
int[] x = {1, 2};
```

`x[0]` is an `int`, and we can use it identical as if we said:

```
int x0 = 1;
```

(See: MaxPassInt.java)
Array - array looping

Arrays often involve loops (since arrays store multiple elements)

Classes had a toString() method that allowed you to display relevant information

java.util.Arrays has this method as well

(See: MaxDisplayArray.java)
Array - passing arrays

Arrays are references like Classes

This means we can pass the reference as an argument in a method

Then the method can see the whole array

(See: MaxPassArray.java)
Array - passing arrays

But wait! If arrays are references, we can leak information like with classes.

To do this, we must modify the original reference (not reassign the reference).
(See: ReverseLeak.java)
Arrays can also be returned by methods (just like Classes)

Again, similar to passing arrays (references) information can leak when arrays are returned

Return a copy if you do not want someone editing your reference!
Arrays do not need to be of primitive types

You can have an array of a Class type, but you will need to run a constructor on every element (loop)

Otherwise, the references will be null

(See: ClassArray.java)
main(String[] args)

When running your Java program, any words from the command line are put into String[] args in main()

```
java HelloWorld hi 2.3456 bye
```

Class name    args[0]    args[1]    args[2]

In NetBeans: Run -> Set Project Config.
-> Customize -> Arguments
(See: MainArgs.java)