News

- Assignment 2
- Corrections to ASCIIArtiste.java posted
- Definitely read WebCT bboards!

Reading

- This week: 8.1, 8.5-8.7, topics 6.3 and 6.4

Recap: While Loop Example

```java
public class WhileDemo
{
    public static void main (String[] args)
    {
        int limit = 3;
        int counter = 1;
        while (counter <= limit)
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 1;
        }
        System.out.println("End of demonstration");
    }
}
```

- While version

Recap: For Loop Example

```java
public class ForDemo
{
    public static void main (String[] args)
    {
        for (int counter = 1; counter <= 3; counter = counter + 1)
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
        }
        System.out.println("End of demonstration");
    }
}
```

- For version

Recap: Do Loop Example

```java
public class DoDemo
{
    public static void main (String[] args)
    {
        int limit = 3;
        int counter = 1;
        do
        {
            System.out.println("The square of " + counter + " is " + (counter * counter));
            counter = counter + 1;
        } while (counter <= limit);
        System.out.println("End of demonstration");
    }
}
```

- Do version
Recap: For Statement

for (initialization; boolean expression; increment) 
body

- **Body** of loop can be
  - single statement
  - whole block of many statements in curly braces
- Control flow
  - first time through: initialization
  - boolean expression evaluated
  - if expression true, body executed; if false, end
  - increment processed
  - boolean expression evaluated
  - if true, body executed; if false, end
  - ....

Recap: For Versus While Statement

how for statement works
- initialization
- boolean expression
  - true
  - false
- statement
- increment

how while statement works
- boolean expression
  - true
  - false
- statement

- flowcharts can be somewhat deceptive
  - need initialization and incrementing/modifying in while loop too
  - although syntax does not require it in specific spot

Recap: Do Statement

- Body always executed at least once

order of four things can change, but need them all

Objectives

- More practice with loops
- Understand when and how to use arrays
  - and loops over arrays

Flipping Coins

- Did **while** version last time
- Let's try **for** version now

Keeping Track of Things

Cans of pop sold this month
185
92
370
485
209
128
84
151
32
563

What's the gross income?
What's the net profit?
Is Bubba stealing loonies?
In other words, how can I organize the data above in my computer so that I can access it easily and do the computations I need to do?

**Answer: Arrays**

- **use arrays**: common programming language construct
  - grouping related data items together
  - meaningful organization such that each individual data item can be easily retrieved or updated

- **collection of variables**
  - all of same type
  - share common name
  - each variable holds single value

Using Arrays

- **Collection of variables has single name**
  - how do we access individual values?

- Each value stored at unique numbered position
  - number called index of array element
    - aka subscript

- **cansSold** name of this array
  - holds 10 values

Using Arrays

- **To access individual value in array**
  - use array name followed by pair of square brackets
  - inside brackets, place index of array element we want to access

- **Reference to array element allowed anywhere that variables can be used**

Example:

```java
System.out.println(cansSold[4]);
```

- Prints value 209
Array Declaration and Types

- Just like ordinary variable, must declare array before we use it
- give array a type
- Since cansSold contains integers, make integer array:
  ```java
  int[] cansSold = new int[10]
  ```
- Looks like variable declaration, except:
  - empty brackets on the left tell Java that cansSold is an array...
  - the number in the brackets on the right tell Java that array should have room for 10 elements when it's created

```java
    // do useful stuff here    System.out.println("Element 4 is "+ cansSold[4]);  }
}
```
Array Declaration and Types

```java
public class ArrayTest2 {
    public static void main(String[] args) {
        int[] cansSold = {185, 92, 370, 485, 209,
                          128, 84, 151, 32, 563};
        // do useful stuff here
        System.out.println("Element 4 is "+cansSold[4]);
    }
}
```

- Can also use initializer list.
- Right side of declaration does not include type or size.
- Java figures out size by itself.
- Types of values on right must match type declared on left.
- Initializer list may only be used when array is first declared.

Using Arrays and Loops

- Write program to create array.
- Find total number of cans sold.
- Print result.

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209,
                          128, 84, 151, 32, 563};
        // do useful stuff here
        System.out.println("Element 4 is "+cansSold[4]);
    }
}
```

- Write program to create array.
- Find total number of cans sold.
- Print result.

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209,
                          128, 84, 151, 32, 563};
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209,
                          128, 84, 151, 32, 563};
        // do useful stuff here
        System.out.println("Element 4 is "+cansSold[4]);
    }
}
```
Using Arrays and Loops

- Write program to
  - create array
  - find total number of cans sold
  - print result

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
        for (int i = 0; i < cansSold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold "+totalCans+" cans of pop");
    }
}
```

Using Arrays and Loops

- Write program to
  - create array
  - find total number of cans sold
  - print result

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = (185, 92, 370, 485, 209, 128, 84, 151, 32, 563);
        for (int i = 0; i < cansSold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold "+totalCans+" cans of pop");
    }
}
```

Using Arrays and Loops

- Write program to
  - create array
  - find total number of cans sold
  - print result

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = (185, 92, 370, 485, 209, 128, 84, 151, 32, 563);
        for (int i = 0; i < cansSold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold "+totalCans+" cans of pop");
    }
}
```

Using Arrays and Loops

- Write program to
  - create array
  - find total number of cans sold
  - print result

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = (185, 92, 370, 485, 209, 128, 84, 151, 32, 563);
        for (int i = 0; i < cans Sold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold "+totalCans+" cans of pop");
    }
}
```

Using Arrays and Loops

- Write program to
  - create array
  - find total number of cans sold
  - print result

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = (185, 92, 370, 485, 209, 128, 84, 151, 32, 563);
        for (int i = 0; i < cansSold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold "+totalCans+" cans of pop");
    }
}
```

Tracing Arrays and Loops

```java
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = (185, 92, 370, 485, 209, 128, 84, 151, 32, 563);
        for (int i = 0; i < cansSold.length; i++) {
            System.out.println("We've sold "+totalCans+" cans of pop");
            totalCans = totalCans + cansSold[i];
        }
    }
}
```
public class ArrayTest3 {
    public static void main(String[] args) {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
        for (int i = 0; i < cansSold.length; i++) {
            totalCans = totalCans + cansSold[i];
        }
        System.out.println("We've sold " + totalCans + " cans of pop");
        totalCans = 0;
        cansSold.length = 10;
        System.out.println("We've sold " + totalCans + " cans of pop");
    }
}

Is i < 10?
  yes, 0 < 10

i = 0

i = 1

i = 2

...
public class ArrayTest3{
  public static void main(String[] args)
  {
    int totalCans = 0;
    int[] cansSold = {185, 92, 370, 485, 209,
                     128, 84, 151, 32, 563};
    for (int i = 0; i < cansSold.length; i++)
    {
      totalCans = totalCans + cansSold[i];
    }
    System.out.println("We've sold " + totalCans + " cans of pop");
  }
}

int[] cansSold = {185, 92, 370, 485, 209,
                  128, 84, 151, 32, 563};
for (int i = 0; i < cansSold.length; i++)
{
  totalCans = totalCans + cansSold[i];
}
System.out.println("We've sold " + totalCans + " cans of pop");
public class ArrayTest3
{
    public static void main(String[] args)
    {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
        for (int i = 0; i < cansSold.length; i++)
        {
            totalCans += cansSold[i];
        }
        System.out.println("We've sold " + totalCans + " cans of pop");
    }
}

And so on...
public class ArrayTest3{
  public static void main(String[] args) {
    int totalCans = 0;
    int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
    for (int i = 0; i < cansSold.length; i++) {
      totalCans = totalCans + cansSold[i];
      System.out.println("We've sold "+totalCans + " cans of pop");
    }
  }
}
cansSold.length 10
6
2299
And so on...
**Initializing Array With Keyboard Input**

```java
import java.util.Scanner;

public class ArrayTest3b {
    public static void main(String[] args)
    {
        final int ARRAYSIZE = 10;
        Scanner scan = new Scanner(System.in);

        for (int i = 0; i < cansSold.length; i++)
        {
            System.out.print("Enter machine "+
                                 "'s sales: ");
            cansSold[i] = scan.nextInt();
        }

        // do useful stuff here
        System.out.println("Element 4 is " +
                           cansSold[4]);
    }
}
```

**Averaging Loop Example**

Let's say we want to write a program that prints average of values in some arbitrarily large array

- like the one to the left called numbers

- Will require loop
- Simple task for looping in the context of an array
- how will we make this happen?

**Tracing Arrays and Loops**

```
cansSold.length: 10

public class ArrayTest3
{
    public static void main(String[] args)
    {
        int totalCans = 0;
        int[] cansSold = {185, 92, 370, 485, 209,
                          128, 84, 151, 32, 563};

        for (int i = 0; i < cansSold.length; i++)
        {
            totalCans = totalCans + cansSold[i];
            System.out.println("We've sold "+
                                 " cans of pop");
        }
        System.out.println("We've sold "+
                                 " cans of pop");
        System.out.println("Element 4 is " +
                           cansSold[4]);
    }
}
```

**Something To Remember!**

Array `cansSold` created with 10 elements

- Indices (plural of index) are 0 through 9

- In general, array of size n will have indices ranging from 0 through n-1

- When you number things, you're used to beginning with 1

- Computer folks begin with 0

- leads to “off by one” errors, even among computer veterans

**Computer folks begin with 0**

- leads to “off by one” errors, even among computer veterans

- To begin with 1, arrays are numbered beginning with 1!

- leads to “off by one” errors, even among computer veterans

- Computer folks begin with 0!

- leads to “off by one” errors, even among computer veterans

- To begin with 1, arrays are numbered beginning with 1!

- Conventions for counting the indices of an array

- Consider the indices of an array named `cansSold` with 10 elements:

- For an array in computer science, it begins with 0!

- For an array in computer science, it begins with 0!

- Indices (plural of index) are 0 through 9

- In general, array of size n will have indices ranging from 0 through n-1

- When you number things, you're used to beginning with 1

- Computer folks begin with 0

- leads to “off by one” errors, even among computer veterans

- To begin with 1, arrays are numbered beginning with 1!

- leads to “off by one” errors, even among computer veterans

- Computer folks begin with 0!

- leads to “off by one” errors, even among computer veterans

- To begin with 1, arrays are numbered beginning with 1!

- leads to “off by one” errors, even among computer veterans
PrintMax Loop Example

- Now instead of average, we want to find and print maximum value from some arbitrarily large array
- Similar loop, but with some extra tweaks.

Histogram Loop Example

- Now use same data as basis for histogram
- Write one loop to look at value associated with each row of array
- For example, if program reads value 6 from the array, should print line of 6 asterisks
  - Program then reads the value 8, prints a line of 8 asterisks, and so on.
- Need outer loop to read individual values in the array
- Need inner loop to print asterisks for each value

Storing Different Data Types

- Write program to compare what’s been collected from each machine vs. how much should have been collected?

```java
class ArrayTest4
{
    public static void main(String[] args)
    {
        double expected;
        int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
        double[] cashIn = {201.25, 100.50, 412.75, 555.25, 195.00, 160.00, 105.00, 188.75, 40.00, 703.75};
        for (int i = 0; i < cansSold.length; i++)
        {
            expected = cansSold[i] * 1.25;
            System.out.println("Machine " + (i+1) + " off by $" + (expected - cashIn[i]));
        }
    }
}
```

Could use two arrays of same size but with different types

Storing Different Data Types

```java
class ArrayTest4
{
    public static void main(String[] args)
    {
        double expected;
        int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
        double[] cashIn = {201.25, 100.50, 412.75, 555.25, 195.00, 160.00, 105.00, 188.75, 40.00, 703.75};
        for (int i = 0; i < cansSold.length; i++)
        {
            expected = cansSold[i] * 1.25;
            System.out.println("Machine " + (i+1) + " off by $" + (expected - cashIn[i]));
        }
    }
}
```

Could use two arrays of same size but with different types
Storing Different Data Types

Arrays With Non-Primitive Types

Arrays Of Objects


Arrays of Objects

- Now we can put references to Strings in our String array.

```java
location[0] = "Chan Centre";
location[1] = "Law School";
location[2] = "Main Library";
```

- Or we could have done this:

```java
String[] location = {
    "Chan Centre", 
    "Law School", 
    "Main Library", 
    ... 
};
```

- Each individual String object in array of course has all String methods available.
- For example, what would this return?

```java
location[2].length()
```

- Think about a cleaner way to do all this...