Note

• HW2 is already out, due February 26 at 5pm
• HW1 is graded.

What we are covering today

• Quick review from lab 3
  – If...else
  – Iteration/Looping
• Switch Statements
• Methods
• Variable scope
if...else...if

```java
if (condition1) {
    <statement1>
    <statement2>
} else if (condition2) {
    <statement3>
    <statement4>
} else {
    <statement5>
}
```

else

```java
else if (condition) {
    <statement3>
    <statement4>
} else {
    <statement5>
}
```

switch

```java
switch (variable) {
    case value1:
        <statement1>
        <statement2>
        break;
    case value2:
        <statement3>
        <statement4>
        break;
    default:
        <statement5>
}
```

switch (args.length) {
```java
    case 0:
        System.out.println("no input entered");
        break;
    case 1:
        System.out.println("one input entered");
        break;
    default:
        System.out.println(">1 input entered");
```}

Methods

- A method groups together statements in a logical manner
- So far we have seen a single method in any given Java program

```java
public static void main(String[] args) {
    // method body (statements) goes here
}
```

Components of a method declaration

- `public`: other Java classes could hypothetically call this method
- `static`: 
- `void`: return type
- `main`: identifier – name of method
- `()`: delimits the input variables
- `String[] args`: the input variable TYPE and NAME (>1 variable are comma separated)

Methods

- There can be more than one method in a program. The way to jump from method to method is by calling the method

```java
class Example{
    public float checkValidDiv(int a, int b) {
        double div = 0;
        if (b == 0) {
            return -1;
        } else {
            div = (float) a / b;
        }
        return div; // if the return value is not specified as void, you must 'return' a value
    }
}
```

```java
public static void main(String[] args) {
    System.out.println("program starts here");
    int returnVal = checkValidDiv(4,0);
    // this is the method call!
    if (returnVal == -1)
        System.out.println("you tried to divide by 0");
}
```

Components to a method call:
- input values, return value
main method is static

- The main method declaration will always look like that
- It must always be declared static
- Therefore it is not an ideal place to write the body of your program

Constructors

- Every class has a special method called a constructor.
- Like main, the constructor has a special syntax. No return value etc, only needs an identifier. The identifier must match the class name.

```java
class HelloWorld{
    HelloWorld(){
        System.out.println("Hello World");
    }
    public static void main(String[] args){
        new HelloWorld();
    }
}
```

- In the main method, we will be calling the constructor for the class. To call the constructor method, we use the keyword “new” before its identifier

Constructors can have input

```java
class HelloWorld{
    HelloWorld(String[] printme){
        System.out.println(printme[0]);
    }
    public static void main(String[] args){
        new HelloWorld(args);
    }
}
```

Difference between a regular method call and a constructor method call

* new keyword
* never a return value
Example.java with a constructor

class Example
{
    Example(String[] args)
    {
        int returnVal = checkValidDiv(4,0);
        // this is the method call!
        if (returnVal == -1)
        {
            System.out.println("you tried to divide by 0");
        }
    }

    public float checkValidDiv(int a, int b)
    {
        double div = 0;
        if (b == 0)
        {
            return -1;
        }
        else
        {
            div = (float)a/b;
        }
        return div;
    }

    public static void main(String[] args)
    {
        System.out.println("program starts here");
        new Example(args);
    }
}

HW2

• IMPORTANT! Name your class Palindrome
  – (your file should be called Palindrome.java)

• Write a method called isPalindrome that takes one parameter
  (a string) and returns a boolean (Java) indicating whether or
  not the supplied String is a palindrome.

• Modify the palindrome checking procedure so that it's case-
  insensitive. Hint: use java's Character.toLowerCase method

• Modify the palindrome checking procedure so that it ignores
  whitespace and punctuation. In particular, handle spaces (),
  periods (.), commas (,), and apostrophes (')

The Java API

• The java API contains information about all of java's methods

    http://java.sun.com/j2se/1.4.2/docs/api/