Quick Exercise: Increment Operator

What is the value of a and b at the end of each code fragment?

```java
int a = 1;
int b = a + a++;
```

```java
int a = 1;
int b = a + ++a;
```

```java
int a = 1;
int b = a++ + a;
```

```java
int a = 1;
int b = a*++a;
```

```java
int a = 1;
int b = (a++ + 2)*a;
```

```java
int a = 1;
int b = a++ + a + a++;
```
Iteration (Loops)

while (condition to continue)
{
    // repeat as long as condition = true
}

do
{
    // run once and repeat as long as condition = true
}
while (condition to continue);

for (initial statement, condition to continue, increment statement)
{
    // execute initial statement
    // repeat as long as condition = true
    // execute increment statement after each repetition
}
while vs. do...while

```java
double weight = 0; // Declaration and Initialization
while (weight < 40) {
    String s = JOptionPane.showInputDialog("Enter Weight");
    weight = Double.parseDouble(s);
}
```

```java
double weight; // Declaration only
do {
    do {
        String s = JOptionPane.showInputDialog("Enter Weight");
        weight = Double.parseDouble(s);
    }
    while (weight < 40);
```
**while vs. for**

*while* loops and *for* loops are often interchangeable.

What does the following loop do?

```java
int i = 0;
while (i < 5) {
    System.out.print(i + " ");
    i++;
}
```

How would you do the same thing with a *for* loop?
Common for loop errors
(Check these first if you get unexpected results)

for (int i = 0; i > 5; i++) {
    System.out.println(i + " ");
}  
What does this code do?

Common errors in declaring for loops:
1) Incorrect termination statement
   • Remember, the loop will only run as long as the termination is TRUE
   • Make your termination statement to be true as long as you want the loop to run, 
     not when you want it to stop

2) Semicolon after loop declaration
   • The for loop and all loops exist inside brackets – brackets are a way of grouping 
     bits of code for execution
   • The semicolon terminates a line. Java will move to the next line and won’t know 
     to associate the for loop with what is in the brackets
while vs. for

The method uses a for loop to raise x to the power of n.

```java
public static double power(double x, int n) {
    double result = 1;
    for (int i = 1; i <= n; i++)
        result *= x;
    return result;
}
```

Write a while loop version

```java
public static double power(double x, int n) {
    double result = 1;
    for (int i = 1; i <= n; i++)
        result *= x;
    return result;
}
```
Nested Loops

What does the following double loop print?

```java
int i = 0;
while (i < 3)
{
    i++;
    for (int j = 0; j <= i; j++)
    {
        System.out.println(10*i + j);
    }
}
```

Can you get rid of the some braces {} and keep the same output?
Methods

What is a method?

Where do you write methods?

How many methods per class?

How many `main()` method(s) per class?
Example: a **power** method that raises any number \( x \) to an integer power \( n \)

```java
public static double power(double x, int n)
```

- A method has a single return type: it can only return one "thing."
- If nothing is returned, the return type is **void**.
- The number of arguments is not limited.
Exercise: Income Tax Calculator

Using the following tax brackets, write a method to compute the tax applicable to any income.

<table>
<thead>
<tr>
<th>Income Bracket</th>
<th>Tax rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 – 5,000</td>
<td>0</td>
</tr>
<tr>
<td>5,000 – 12,500</td>
<td>7</td>
</tr>
<tr>
<td>&gt; 12,500</td>
<td>16</td>
</tr>
</tbody>
</table>

For example, if income = 13,000:
\[\text{tax} = 0.07 \times (12,500 - 5,000) + 0.16 \times (13,000 - 12,500)\]

Method signature?
Exercise: Income Tax Calculator

```java
public static double calcTax(double inc) {
    if (inc < 5000)
        return 0;
    else if (inc < 12500)
        return 0.07 * (inc - 5000);
    else
        return 0.07 * (12500 - 5000) + 0.16 * (inc - 12500);
}
```

What if you use an `else if` instead of an `else` for the third case?
Primitives are passed "by value".

```java
public static void main(String[] args) {
    double a = 2.0;
    int b = 3;
    double p = power(a, b);
}

public static double power(double x, int n) {
    double result = 1.0;
    while (n > 0) {
        result *= x;
        n--;
    }
    return result;
}
```
Exercise: Methods

You will write a method to compute binomial coefficients.

The binomial coefficient \( \binom{n}{k} \) can be computed as: \( \binom{n}{k} = \frac{n!}{(n-k)! \cdot k!} \)

Where \( x! \) is the factorial operator (e.g. \( 5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \))

Strategy?
Factorial Method

// for loop version

// while loop version
Binomial Method

\[ \binom{n}{k} = \frac{n!}{(n-k)! \cdot k!} \]

// putting it all together

Using methods avoids code repetition!
Homework 2: Throw ball in basket

Main method should:

• Prompt user for input
• Call the following methods:
  – Compute optimal angle
  – Compute smallest angle
  – Compute largest angle
  – Compute max height
  – Determine if ball hits ceiling
  – Adjust velocity incrementally until ball does not hit ceiling anymore
• Print all results to console

Math is an existing library in Java

Example methods in Math class:

# = Math.cos(∠)
∠ = Math.asin(#)
x^y = Math.pow(x, y)
π = Math.PI

Math methods are in radians!