topics:
more operators
loops:
for, while, do-while
exit conditions
discuss lab 4 - handout

1. **operators and precedence**
   **highest to lowest:**
   - ()
   - +, - unary
   - ++, --, (unary increment and decrement, prefix)
   - *, /, %
   - +, - binary
   - ++, --, unary increment and decrement postfix
   - <, <=, >=, > (relational)
   - ==, != (equality)
   - =, +=, -=, /=, *=, %= (assignment)

2. postfix vs. prefix
determines when the number is incremented or decremented – before or after it is used in the evaluation of the expression. The value is permanently changed.
ex. int nTries = 0;
   System.out.println(nTries++); //displays 0, and then nTries is incremented to 1
   System.out.println (nTries); // displays 1

   int nFirstNum = 7, nSecondNum;
   nSecondNum = ++nFirstNum – 3; // nFirstNum gets 8; nSecondNum gets 5

3. Repetition structure: While

Use **WHILE** when you can state a condition under which you want to perform one or more statements. The test is at the top of the loop, so it's possible that no iterations will be performed. The exit condition is the logical opposite of the loop condition.
while (condition) [a condition is a boolean expression]
    Statement;

eexample:
while (nItems < MINIMUM)
    nItems = Integer.parseInt(JOptionPane.showInputDialog
                             ("Enter minimum order");

========
while (condition) {
    statement;
    statement;
}

eexample:
int nTotal = 0, nNum = 1;
// add all integers from 1 to 10
while (nNum <= 10) {       // what is the exit condition?
    nTotal += nNum;
    nNum++;
}

4. DO-WHILE
Use DO-WHILE when you want at least one iteration, because the test of the
condition is done at the bottom of the loop:

do {
    Statements
} while (condition)

// using a do-while to error-check input
// for the range nLower to nUpper
int nLower = 10, nUpper = 20;
String sVal;
do {
    sVal = JOptionPane.showInputDialog(null,
    " Please enter a number between 10 and 20: ");
    nVal = Integer.parseInt(sVal);
} while (nVal < nLower || nVal > nUpper);
// now we use nVal in some way
5. For loops
use FOR when you know, or can calculate, exactly how many iterations of the loop
you want to execute.

**syntax:** for (init; condition, update)

**order of execution:**
1. init (first time only)
2. test condition
3. if true, execute body of loop; if false, exit the loop immediately
4. if still in loop - second(and subsequent iterations) – update, then test

**first time:** init, test
**all other times:** update, test

**example:**
//adds up all integers from 1 to 100
int nTotal;
for (int i = 1; i <= 100; i++)
    nTotal += i;

**example:**
// prints 15 stars
for (int i = 1; i <= 15; i++)
    System.out.print("*");

// could also be written this way:
for (int i = 0; i < 15; ++i)
    System.out.print("*");

**example:**
// prints numbers in reverse order – 20 19 18 17 ….2 1
for (int i = 20; i > 0; i--)
    System.out.print( i + " ");

**example:**
// adds up even numbers from 2 to 20
int nTotal = 0;
for (int a = 2; a <= 20; a += 2)
    nTotal += a;
You can set more than one initial value, test more than 1 condition (using boolean operators such as && and ||, and increment more than one variable:

```java
for (int i = 1, j = 10; i <= j; i++, j--)
    System.out.print(i + "   " + j);
```

**output**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

5. Discuss Lab 4 due next week MailOrder

Look at example in book: for sentinel-controlled loop, p 219, #4-9 SoccerPoints

Draw Logic Diagram for lab