Nested Loops
Nested Loops

- Just as one selection structure can be nested within another (or within a loop), one loop can be nested inside another
- When one loop is nested within another, several iterations of the inner loop are performed for every single iteration of the outer loop
Example – multiplication table

- Suppose you wanted to print a multiplication table of the sort your instructor was forced to memorize in 2nd grade (you’ve seen the picture!)
- Each row and column of the table is tagged with a numeric heading, and each entry in the table is the product of the two headings
## Multiplication table output – an excerpt

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>9</td>
<td>12</td>
<td>15</td>
<td>18</td>
<td>21</td>
<td>24</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>12</td>
<td>16</td>
<td>20</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>36</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>30</td>
<td>35</td>
<td>40</td>
<td>45</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
<td>18</td>
<td>24</td>
<td>30</td>
<td>36</td>
<td>42</td>
<td>48</td>
<td>54</td>
</tr>
<tr>
<td>7</td>
<td>14</td>
<td>21</td>
<td>28</td>
<td>35</td>
<td>42</td>
<td>49</td>
<td>56</td>
<td>63</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>24</td>
<td>32</td>
<td>40</td>
<td>48</td>
<td>56</td>
<td>64</td>
<td>72</td>
</tr>
<tr>
<td>9</td>
<td>18</td>
<td>27</td>
<td>36</td>
<td>45</td>
<td>54</td>
<td>63</td>
<td>72</td>
<td>81</td>
</tr>
</tbody>
</table>
Multiplication table - headings

Print the numbers between 2 and 15, spaced evenly
Print a series of hyphens in a single line
Place an end of line character after each of the lines above

```java
public void printHeadings () {
    System.out.printf ("%8s", "");
    for (int x=2; x<=15; x++)
        System.out.printf ("%5d", x);
    System.out.print("\n");
    for (int y=0; y<80; y++)
        System.out.print("-");
    System.out.print("\n");
}
```
Multiplication table

• Outer loop controls the number of lines to be printed; contains:
  – Inner loop
  – Line to print a newline character
• Inner loop controls the contents of each line
  – Row heading
  – Product of current row & column headings
public void drawTable () {
    for (int x = start; x <= size; x++)
    {
        for (int y = start; y <= size; y++)
        {
            if (y==start)
                System.out.printf("%7d%s", x, "|");
            System.out.printf("%5d", (x * y));
        }
        System.out.printf("\n");
    }
}
Tracing nested loops

- Write down value of each loop counter as it changes during loop execution
- If any output or change in other variable occurs, write this down next to the tally of loop counters
## Example – multiplication table

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>4 6 8 ... 30</td>
</tr>
<tr>
<td>3</td>
<td>2 3 4 ... 15 16</td>
<td>6 9 12 ... 45</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>8 ... 60</td>
</tr>
<tr>
<td>...</td>
<td>... 15 16</td>
<td>... 225</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>... 30 ... 225</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Pattern of a Nested Loop

initialize outer loop
while ( outer loop condition )
{
    ...

    initialize inner loop
    while ( inner loop condition )
    {
        inner loop processing and update
    }

    ...

}
Example Problem

Suppose we have data in the form below, involving several ID strings. For each ID string, a variable number of readings have been recorded; the number of readings for each ID is shown in the howMany column.

<table>
<thead>
<tr>
<th>ID</th>
<th>howMany</th>
<th>Readings</th>
</tr>
</thead>
<tbody>
<tr>
<td>4567</td>
<td>5</td>
<td>180 140 150 170 120</td>
</tr>
<tr>
<td>2318</td>
<td>2</td>
<td>170 210</td>
</tr>
<tr>
<td>5232</td>
<td>3</td>
<td>150 151 151</td>
</tr>
</tbody>
</table>
Our goal: read in the data and display a summary chart like the one shown below:

<table>
<thead>
<tr>
<th>ID</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>4567</td>
<td>152</td>
</tr>
<tr>
<td>2318</td>
<td>190</td>
</tr>
<tr>
<td>5232</td>
<td>151</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>There were 15 data sets on file</td>
<td></td>
</tr>
</tbody>
</table>
Algorithm

• initialize count to 0
• read first ID and howMany
• while not at end of data
  - increment count
  - display ID
  - use a count-controlled loop to read and sum up this ID’s howMany readings
  - calculate and display average for ID
  - read next ID and howMany
• display count
import java.util.*;

public class NestLoop {
    public static void main (String [] args) {
        int total = 0;    // total for all IDs
        int thisID,      // current ID number
                        howMany, // number of readings for current ID
                        reading, // current reading
                        idTotal, // total for current ID number
                        idCount, // counter for inner loop
                        again;   // outer loop control variable
        double average;  // average for current ID
        Scanner kb = new Scanner (System.in);
System.out.printf("%s%13s\n", "ID Number", "Average");

do {  // start of outer loop
    System.out.print("Enter ID number ");
    thisID = kb.nextInt();
    System.out.print(
        "How many readings for this ID?"");
    howMany = kb.nextInt();
    idTotal = 0;
    idCount = 0;
    total++;
}
// inner loop – process all readings for this ID

while (idCount < howMany) {
    System.out.print("Enter reading");
    reading = kb.nextInt();
    idTotal += reading;
    idCount++;
}

// continuation of outer loop

average = (double)idTotal / howMany;
System.out.print("" + thisID);
System.out.printf("%17.2f\n", average);
System.out.print("Enter 1 to continue, 0 to quit: ");
again = kb.nextInt();
} while (again == 1);
System.out.println ("Total of " + total +
        " records were processed.");
Using nested loops to draw figures (ASCII art)

- Drawing figures can illustrate how nested loops work
- Keep in mind the principle: outer loop controls number of lines, inner loop controls content of lines
Trace the following loop

```java
int x, y;
for(x=0; x<5; x++)
{
    for(y=5; y>0; y--)
        System.out.print("* ");
    System.out.print("\n");
}
```
Trace the following loop

```java
import java.util.*;

public class triangle {
    public static void main (String [] args) {
        int x, y, z, height;
        Scanner kb = new Scanner(System.in);
        System.out.print("Enter height: ");
        height = kb.nextInt();
        for (x=0; x<height; x++)
        {
            for (y=height; y>x; y--)
                System.out.print(" ");
            for (z=0; z<=x; z++)
                System.out.print("* ");
            System.out.print("\n");
        }
    }
}
```

```
height = 4
x  y  z
0  4  3  2  1  0  0  1
1  4  3  2  1  0  1  2
2  4  3  2  0  1  2  3
3  4  3  0  1  2  3  4
Output:
```
```markdown
*  
  *  *
  *  *  *
  *  *  *  *
```

- y loop prints spaces
- z loop prints stars
Loop example with break statement

```java
int x, y;
for (x=1; x<5; x++)
{
    for (y=1; y<5; y++)
    {
        if (y > x)
            break;
        System.out.print("* ");
    }
    System.out.print("\n");
}
```
Continue statement

- is valid only within loops
- terminates the current loop iteration, but not the entire loop
- in a For or While, continue causes the rest of the body statement to be skipped--in a For statement, the update is done
- in a Do-While, the exit condition is tested, and if true, the next loop iteration is begun
Loop example with continue

```java
int x, y;
for (x = 1; x < 5; x++) {
    for (y = 1; y < 5; y++) {
        if (y > x)
            break;
        System.out.print("* ");
    }
    if (x % 2)
        continue;
    System.out.print("\n");
}
```
Loop Testing and Debugging

- test data should test all sections of program
- beware of infinite loops -- program doesn’t stop
- check loop termination condition, and watch for “off-by-1” problem
- use get function for loops controlled by detection of ‘\n’ character
- trace execution of loop by hand with code walkthrough
- use a debugger to run program in “slow motion” or use debug output statements