Flow of Control: Loops

Chapter 4
Part 6 – Nested Loops
Objectives

• Design a loop
• Use **while**, **do**, and **for** in a program
• Use the **for-each** with enumerations
• Use assertion checks
• Use repetition in a graphics program
• Use **drawString** to display text in a graphics program
Nested Loops

• The body of a loop can contain any kind of statements, including another loop
Nested Loops

- The body of a loop can contain any kind of statements, including another loop.
- Exercise
  - Write the code snippet that creates the following output:
    1  2  3  4  5  6  7  8  9  10
Nested Loops

• The body of a loop can contain any kind of statements, including another loop

• Exercise
  ▪ Write the code snippet that creates the following output
    1 2 3 4 5 6 7 8 9 10

• Solution:
  ```java
  for (int i = 0; i <= 10; ++i)
      System.out.print(i + " ");
  System.out.println(" ");
  ```
Nested Loops

• Exercise
  ▪ Write the code snippet that creates the following output

```
* * * * *
```
Nested Loops

• Exercise
  • Write the code snippet that creates the following output

• Solution:

```java
for (int i = 0; i <= 5; ++i)
    System.out.print("*");
```

* * * * * *
Nested Loops

• Exercise
  ▪ Write the code snippet that creates the following output
  ▪ This will require nested loops
Nested Loops

• Exercise
  ▪ Write the code snippet that creates the following output

• Solution:

```java
for (int i = 0; i <= 4; ++i) {
    for (int j = 0; j <= i; ++j) {
        System.out.print ("*" + " ");
    }
    System.out.println (" ");
}
```
Nested Loops

• Exercise
  ▪ Write the code snippet that creates the following output

• Solution:

```java
for (int i = 0; i <= 10; ++i)
    System.out.print(i + "  ");
System.out.println(" ");
```
Nested Loops

• Exercise
  ▪ Trace the following code snippet and find out what is the output

1. for (int i = 0; i <= 3; ++i) {
   2. for (int j = 4; j >0; --j) {
      3. System.out.print((i+j)+" ");
   }
   4. System.out.println("");
}
Nested Loops

1. for (int i = 0; i <= 3; ++i)
   {
2.  for (int j = 4; j >0; --j)
   {
3.  System.out.print((i+j)+" ");
   
4.  System.out.println(" ");
   
   }
 }

<table>
<thead>
<tr>
<th>Line #</th>
<th>i</th>
<th>j</th>
<th>cond1</th>
<th>cond2</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>4</td>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>2</td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td></td>
<td>T</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td></td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>\n</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Nested Loops

• Exercise
  - Final output

4 3 2 1
5 4 3 2
6 5 4 3
7 6 5 4

• Solution:

```java
for (int i = 0; i <= 3; ++i)
{
    for (int j = 4; j >0; --j)
    {
        System.out.print((i+j)+" ");
    }
    System.out.println(" ");
}
```
Nested Loops Example: ExamAverager

Exercise: Trace it

Sample Screen Output

This program computes the average of a list of (nonnegative) exam scores. Enter all the scores to be averaged. Enter a negative number after you have entered all the scores.

100
90
100
90
-1

The average is 95.0
Want to average another exam? Enter yes or no.

yes

Enter all the scores to be averaged. Enter a negative number after you have entered all the scores.

90
80

The average is 80.0
Want to average another exam? Enter yes or no.

no

LISTING 4.4 Nested Loops

import java.util.Scanner;
/**
 * Computes the average of a list of (nonnegative) exam scores. Repeats computation for more exams until the user says to stop.
 */
public class ExamAverager {
    public static void main(String[] args) {
        System.out.println("This program computes the average of");
        System.out.println("a list of (nonnegative) exam scores.");
        double sum;
        int numberOfStudents;
        double next;
        String answer;
        Scanner keyboard = new Scanner(System.in);
        do {
            System.out.println();
            System.out.println("Enter all the scores to be averaged.");
            System.out.println("Enter a negative number after you have entered all the scores.");
            90
            80
            -1
            The average is 80.0
            Want to average another exam? Enter yes or no.
        } while (answer.equalsIgnoreCase("yes"));
    }
}
Nested Loops Notes

• In the previous example
  - The average score was computed using a **while** loop
  - This **while** loop was placed inside a **do-while** loop so the process could be repeated for other sets of exam scores
Using BlueJ Debugger

• Some Video Tutorials
  – Irv Kalb: Using the Debugger in BlueJ with Java (7:50)
  – Scott Oliver: BlueJ Debugger (4:45)
  – BlueJ team: How to use the Debugger in BlueJ (3:42)
Summary

• A loop is a programming construct that repeats an action
• Java has the **while**, the **do-while**, and the **for** statements
• The **while** and **do-while** repeat the loop while a condition is true
• The logic of a **for** statement is identical to the while
• Loops may be ended using a sentinel value or a boolean value
Summary

• Typical loop bugs include infinite loops or loops which are off by 1 iteration

• Variables may be traced by including temporary output statements or a debugging utility