Repetition Statements

CSC 1051 – Data Structures and Algorithms I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
www.csc.villanova.edu/~map/1051/

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus
Repetition structures in Java

**while** loop:

```java
int count = 0;
while (count < 5)
{
    count++;
    System.out.println (count);
}
```

- Other repetition structures (Chapter 6 in text)
  - the **do** loop
  - the **for** loop
The **`while`** and **`do`** loops are similar.

**while Loop**

- **Condition evaluated**
  - True
  - **Statement**
  - True
  - **Condition evaluated**
  - False

**do Loop**

- **Statement**
  - True
  - **Condition evaluated**
  - False
**for**: a loop with built in “counter”

- **Initialization**
- **Condition evaluated**
  - **true**
  - **Statement**
  - **Increment**
  - **false**
An example of a do loop:

```java
int count = 0;
do{
    count++;
    System.out.println (count);
} while (count < 5);
```

- The body of a do loop executes at least once
The **do** Statement

• A **do** *statement* has the following syntax:

```java
do
{
    statement-list;
} while (condition); //end do
```

• The **statement-list** is executed once initially, and then the **condition** is evaluated.

• The statement is executed **repeatedly** until the condition becomes **false**.

• See example [GradeReportLoopy.java](#)
import java.util.Scanner;

public class GradeReportLoopy
{
    public static void main (String[] args)
    {
        int grade;
        String ans;

        Scanner scan = new Scanner (System.in);
        do
        {
            System.out.print ("Enter a numeric grade (0 to 100): ");
            grade = scan.nextInt();
            printReport(grade);
            System.out.print ("Would you like to do another grade report? ");
            ans = scan.next();
        } while (ans.charAt(0) == 'y' || ans.charAt(0) == 'Y') ;
    }

public static void printReport (int grade) {
    int category = grade / 10;

    System.out.print ("That grade is ");

    switch (category) {
    case 10:
        System.out.println ("a perfect score. Well done.");
        break;
    case 9:
        System.out.println ("great!");
    case 8:
        System.out.println ("well above average. Excellent.");
        break;
    case 7:
        System.out.println ("above average. Nice job.");
        break;
    case 6
        System.out.println ("below average.");
        break;
    default:
        System.out.println ("not passing.");
    }
}
import java.util.Scanner;

public class GradeReportLoopy {
    public static void main(String[] args) {
        int grade;
        String ans;

        Scanner scan = new Scanner(System.in);
        do {
            System.out.print("Enter a numeric grade (0 to 100): ");
            grade = scan.nextInt();
            printReport(grade);
            System.out.print("Would you like to do another grade report? ");
            ans = scan.next();
        } while (ans.charAt(0) == 'y' || ans.charAt(0) == 'Y');
    }
}

Output
Enter a numeric grade (0 to 100): 84
That grade is well above average. Excellent.
Would you like to do another grade report? yes
Enter a numeric grade (0 to 100): 99
That grade is great!
well above average. Excellent.
Would you like to do another grade report? Y
Enter a numeric grade (0 to 100): 100
That grade is a perfect score. Well done.
Would you like to do another grade report? no
Try this:

- Write a do loop to print the even numbers from 2 to 100.

- Write a loop to input a grade, ensuring that it is in the range 1-100 (ask them to try again and again until they input a number in that range)
The **for** Statement

- A **for** loop is functionally equivalent to the following **while** loop structure:

```java
initialization;
while ( condition )
{
    statement;
    increment;
}
```
The **for** Statement - Example

• A **while** loop:

```java
int count=1;
while (count <= 5) {
    System.out.println (count);
    count++;
}
```

• Equivalent **for** loop:

```java
for (int count=1; count <= 5; count++)
    System.out.println (count);
```
The for Statement

- A *for statement* has the following syntax:

```
for ( initialization ; condition ; increment )
 statement;
```

- The *initialization* is executed once before the loop begins.
- The *statement* is executed until the *condition* becomes false.
- The *increment* portion is executed at the end of each iteration.
The for Statement

- A *for statement* has the following syntax:

```java
for (int count=1; count <= 5; count++)
    System.out.println (count);
```

- The *initialization* is executed once before the loop begins.
- The *statement* is executed until the *condition* becomes false.
- The *increment* portion is executed at the end of each iteration.
Try this:

• Write a for loop to print the even numbers from 2 to 100.

• Write a method that uses a for loop to print n stars, where n is an integer parameter.

• Write a method that uses a for loop to print a square of n by n stars, where n is an integer parameter. Here is an example of the 3 by 3:
  
  * * *
  
  * * *
  
  * * *
  
  * * *
The for Statement

• The increment section can perform any calculation:

```java
for (int num=100; num > 0; num -= 5)
   System.out.println (num);
```

• A `for` loop is well suited for executing statements a specific number of times that can be calculated or determined in advance

• See [Stars.java](http://example.com/Stars.java)
public class Stars
{
    // Prints a triangle shape using asterisk (star) characters.
    public static void main (String[] args)
    {
        final int MAX_ROWS = 10;

        for (int row = 1; row <= MAX_ROWS; row++)
        {
            for (int star = 1; star <= row; star++)
                System.out.print ("*");

            System.out.println();
        }
    }
}
public class Stars {
    // Prints a triangle using asterisk (star) characters.
    public static void main(String[] args) {
        final int MAX_ROWS = 10;
        for (int row = 1; row <= MAX_ROWS; row++) {
            for (int star = 1; star <= row; star++)
                System.out.print("*");
            System.out.println();
        }
    }
}
The for Statement

• Each expression in the header of a for loop is optional

• If the initialization is left out, no initialization is performed

• If the condition is left out, it is always considered to be true, and therefore creates an infinite loop

• If the increment is left out, no increment operation is performed
More examples using for loops

• Conditionals and loops enhance our ability to generate interesting graphics

• See Bullseye.java
• See BullseyePanel.java
• See Boxes.java
• See BoxesPanel.java
import javax.swing.JFrame;

public class Bullseye {
    // Creates the main frame of the program.
    public static void main (String[] args) {
        JFrame frame = new JFrame("Bullseye");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        BullseyePanel panel = new BullseyePanel();

        frame.getContentPane().add(panel);
        frame.pack();
        frame.setVisible(true);
    }
}
import javax.swing.JFrame;

public class Bullseye {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Bullseye");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        BullseyePanel panel = new BullseyePanel();
        frame.getContentPane().add(panel);
        frame.pack();
        frame.setVisible(true);
    }
}
import javax.swing.JPanel;
import java.awt.*;

public class BullseyePanel extends JPanel
{
    private final int MAX_WIDTH = 300, NUM_RINGS = 5, RING_WIDTH = 25;

    //-------------------------------
    //  Sets up the bullseye panel.
    //-------------------------------
    public BullseyePanel ()
    {
        setBackground (Color.cyan);
        setPreferredSize (new Dimension(300,300));
    }
}
continue

// Paints a bullseye target.
public void paintComponent (Graphics page)
{
    super.paintComponent (page);
    int x = 0, y = 0, diameter = MAX_WIDTH;
    page.setColor (Color.white);

    for (int count = 0; count < NUM_RINGS; count++)
    {
        if (page.getColor() == Color.black)  // alternate colors
            page.setColor (Color.white);
        else
            page.setColor (Color.black);

        page.fillOval (x, y, diameter, diameter);

        diameter -= (2 * RING_WIDTH);
        x += RING_WIDTH;
        y += RING_WIDTH;
    }

    // Draw the red bullseye in the center
    page.setColor (Color.red);
    page.fillOval (x, y, diameter, diameter);
}
import javax.swing.JFrame;

public class Boxes {
    public static void main (String[] args) {
        JFrame frame = new JFrame("Boxes");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        BoxesPanel panel = new BoxesPanel();

        frame.getContentPane().add(panel);
        frame.pack();
        frame.setVisible(true);
    }
}
import javax.swing.JFrame;

public class Boxes {

    public static void main(String[] args) {
        JFrame frame = new JFrame("Boxes");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        BoxesPanel panel = new BoxesPanel();
        frame.getContentPane().add(panel);
        frame.pack();
        frame.setVisible(true);
    }
}
import javax.swing.JPanel;
import java.awt.*;
import java.util.Random;

public class BoxesPanel extends JPanel {

    private final int NUM_BOXES = 50, THICKNESS = 5, MAX_SIDE = 50;
    private final int MAX_X = 350, MAX_Y = 250;
    private Random generator;

    // Sets up the drawing panel.
    public BoxesPanel () {
        generator = new Random();

        setBackground (Color.black);
        setPreferredSize (new Dimension(400, 300));
    }

    continue
public void paintComponent(Graphics page)
{
    super.paintComponent(page);

    int x, y, width, height;

    for (int count = 0; count < NUM_BOXES; count++)
    {
        x = generator.nextInt(MAX_X) + 1;
        y = generator.nextInt(MAX_Y) + 1;

        width = generator.nextInt(MAX_SIDE) + 1;
        height = generator.nextInt(MAX_SIDE) + 1;

        continue
    }
}
continue

if (width <= THICKNESS) // check for narrow box
{
    page.setColor (Color.yellow);
    page.fillRect (x, y, width, height);
}
else
    if (height <= THICKNESS) // check for short box
{
    page.setColor (Color.green);
    page.fillRect (x, y, width, height);
}
    else
{
    page.setColor (Color.white);
    page.drawRect (x, y, width, height);
    }
}
Homework

• Review Sections 6.3, 6.4, and 6.5
  • **Always** do all self-review exercises when you review
• Exercises EX6.1 - 6.17 (important to get lots of practice with writing loops!)
• Think about the following exercise from Lab 4 and determine which is the best kind of loop for each of the cases:
  We need to write a program that finds the largest of some numbers input through the keyboard. Here are some questions:
   a) Do we know in advance how many numbers there will be?
   b) Do we want to prompt for each number?
   c) If we do not prompt for each number, how do we signal the end of input?
• Look at some additional examples of programs from the text:
  • ReverseNumber.java
  • Multiples.java

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus
Another example of do loop:

• Reverse a number

• See ReverseNumber.java

• This can be done numerically or by viewing the number as a String – the example in the textbook does it numerically.

• Using strings it is a bit easier (EX 6.17). Try it!
```java
//********************************************************************
//     ReverseNumber.java     Author: Lewis/Loftus
//
//     Demonstrates the use of a do loop.
//********************************************************************

import java.util.Scanner;

public class ReverseNumber {
    //---                                                     
    //     Reverses the digits of an integer mathematically. 
    //---                                                     
    public static void main (String[] args) {
        int number, lastDigit, reverse = 0;

        Scanner scan = new Scanner (System.in);

        continue
    }
}
```
System.out.print("Enter a positive integer: ");
number = scan.nextInt();

do
{
    lastDigit = number % 10;
    reverse = (reverse * 10) + lastDigit;
    number = number / 10;
}
while (number > 0);

System.out.println("That number reversed is "+reverse);
```java
System.out.print("Enter a positive integer: ");
number = scan.nextInt();
do {
    lastDigit = number % 10;
    reverse = (reverse * 10) + lastDigit;
    number = number / 10;
} while (number > 0);
System.out.println("That number reversed is "+ reverse);
```
Another example of for loop:

- Print multiples of a number up to a limit.
- See [Multiples.java](Multiples.java)
- Can you make it print 10 per line?
- Can you change it so that it prints them backwards?

### Sample Run

Enter a positive value: 7
Enter an upper limit: 400

The multiples of 7 between 7 and 400 (inclusive) are:

<table>
<thead>
<tr>
<th></th>
<th>14</th>
<th>21</th>
<th>28</th>
<th>35</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>42</td>
<td>56</td>
<td>63</td>
<td>70</td>
</tr>
<tr>
<td>77</td>
<td>112</td>
<td>126</td>
<td>133</td>
<td>140</td>
</tr>
<tr>
<td>147</td>
<td>154</td>
<td>161</td>
<td>168</td>
<td>175</td>
</tr>
<tr>
<td>182</td>
<td>189</td>
<td>196</td>
<td>203</td>
<td>210</td>
</tr>
<tr>
<td>217</td>
<td>224</td>
<td>231</td>
<td>238</td>
<td>245</td>
</tr>
<tr>
<td>252</td>
<td>259</td>
<td>266</td>
<td>273</td>
<td>280</td>
</tr>
<tr>
<td>287</td>
<td>294</td>
<td>301</td>
<td>308</td>
<td>315</td>
</tr>
<tr>
<td>322</td>
<td>329</td>
<td>336</td>
<td>343</td>
<td>350</td>
</tr>
<tr>
<td>357</td>
<td>364</td>
<td>371</td>
<td>378</td>
<td>385</td>
</tr>
<tr>
<td>392</td>
<td>399</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
import java.util.Scanner;

public class Multiples {

    public static void main (String[] args) {
        final int PER_LINE = 5;
        int value, limit, mult, count = 0;

        Scanner scan = new Scanner (System.in);

        System.out.print ("Enter a positive value: ");
        value = scan.nextInt();

        continue
System.out.print ("Enter an upper limit: ");
limit = scan.nextInt();

System.out.println();
System.out.println ("The multiples of " + value + " between " + value + " and " + limit + " (inclusive) are:");

for (mult = value; mult <= limit; mult += value)
{
    System.out.print (mult + "\t");

    // Print a specific number of values per line of output
    count++;
    if (count % PER_LINE == 0)
        System.out.println();
}
}