Selection and Repetition, revisited

Selection and Repetition Revisited
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/

Selection structures in Java

• Conditional statement:
  \[
  \text{if } (n > 0) \\
  \quad \text{System.out.print(“positive”);} \\
  \quad \text{else} \\
  \quad \text{System.out.print(“negative”);} 
  \]

• Other selection structures (Chapter 6 in text)
  – the \textbf{conditional} operator
  – the \textbf{switch} statement

The Conditional Operator Syntax

\[ \text{condition} \ ? \ \text{expression1} \ : \ \text{expression2} \]

• If the \textbf{condition} is true, \textbf{expression1} is evaluated; if it is false, \textbf{expression2} is evaluated
• The value of the entire conditional operator is the value of the selected expression
• Example: Rewrite this →

\[
\text{if } (n > 0) \\
\quad \text{System.out.print(“positive”);} \\
\quad \text{else} \\
\quad \text{System.out.print(“negative”);} 
\]

\[ \text{(n > 0)} \ ? \ \text{System.out.print(“positive”)} : \ \text{System.out.print(“negative”)} \]

\textbf{The conditional operator is \textit{not a statement}}

Wrong!
More examples:

```java
int bit = (ans.equalsIgnoreCase("Yes") ? 1 : 0);
String status = (age < 18 ? "child" : "adult");
```

- The conditional operator requires three operands so it is sometimes called the ternary operator

Hands on: try this in the Person class!

Try this:

Using if/else:

```java
if (HAPPY)
    page.drawArc (x+15, y+30, 20, 10, 180, 180);
else
    page.drawArc (x+15, y+30, 20, 10, 0,   180);
```

- Rewrite using conditional operator:

Try this:

Rewrite this statement so that "Dime" is printed if `num` equals 1.

```java
System.out.println ("Your change is " +
    num + "Dimes");
```

use conditional operator:

The `switch` Statement: multi-way branches
Recall: Logic of an if-else statement
The switch Statement: multi-way branches

switch statement logic

<table>
<thead>
<tr>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statement 1</td>
<td>Statement 2</td>
<td>Statement 3</td>
</tr>
</tbody>
</table>

Note: this is a simplified flowchart of the logic of the switch statement.

The switch Statement in general

```
switch (expression)
{
    case value1 :
        statement-list1
    case value2 :
        statement-list2
    case value3 :
        statement-list3
    .
    .
    default:   
        statement-list-n
}
```

- Integer, char, or enumerated types
- (Java 7 also allows Strings)
- NO floating point values
- NO ranges of values (e.g., 0<x<10)

So... the logic of the switch is more like this:

```
expression evaluated
```

<table>
<thead>
<tr>
<th>Case 1</th>
<th>Case 2</th>
<th>Case 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statements 1</td>
<td>Statements 2</td>
<td>Statements 3</td>
</tr>
</tbody>
</table>

Note: this is still a simplified flowchart of the logic of the switch statement.

The switch Statement - example

```java
public String toString()
{
    String result = "";
    switch (faceValue)
    {
        case 1:
            result = "one";
            break;
        case 2:
            result = "two";
            break;
        case 3:
            result = "three";
            break;
    }
    return result;
}
```
The image contains a document discussing selection and repetition in programming, with code snippets in Java. The document includes examples of how to use a `switch` statement to grade a numeric input and another example demonstrating a `switch` statement without using `break` statements after each case. The text also includes sample runs for both examples, showing how the program behaves with different inputs. The document is authored by Dr. Papalaskari and is part of the CSC 1051 course at Villanova University.
Selection and Repetition, revisited

Loops revisited: do & for loops
Repetition structures in Java, so far:
while loop:
  int count = 0;
  while (count < 5)
  {
    System.out.println (count);
    count++;
  }
• Other repetition structures (Chapter 6 in text)
  – the do loop
  – the for loop

The do Statement in Java
• A do statement has the following syntax:
  
  do 
  { 
    statement-list;
  } while (condition); //end do
• The statement-list is executed once initially, and then the condition is evaluated
• The statement-list is executed repeatedly until the condition becomes false

The while and do loops are similar.

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

Similar – but not the same:

while Loop

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

do Loop

int count = 0;
do
{ 
  statement-list;
} while (condition); //end do

• The body of a do loop executes at least once

Dr. Papalaskari
Try this:
• Write a do loop to print the even numbers from 2 to 100.

For some things the do loop is more appropriate:

```java
System.out.println("input a number >5");
int num = scan.nextInt();
while (num <= 5)
{
    System.out.println("type a number >5");
    num = scan.nextInt();
}
do {
    System.out.println("type a number >5");
    num = scan.nextInt();
} while (num <= 5);
```

Another example: ReverseNumber.java

```java
public class ReverseNumber {
    public static void main(String[] args)
    {
        int number, lastDigit, reverse = 0;
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter a positive integer: ");
        number = scan.nextInt();
        do
        {
            lastDigit = number % 10;
            reverse = (reverse * 10) + lastDigit;
            System.out.print("(Again? 1=yes,0=no ");
            answer = scan.nextInt();
        } while (1 == scan.nextInt());
        System.out.println("Thank you. GoodBye.");
    }
}
```

Sample Run
Enter a positive integer: 2986
That number reversed is 6892
Selection and Repetition, revisited

Another example: EvenOdd.java

```java
import javax.swing.JOptionPane;

public class EvenOdd
{
    // Demonstrates the use of the JOptionPane class.
    public static void main (String[] args)
    {
        String numStr, result;
        int num, again;
        do
        {
            numStr = JOptionPane.showInputDialog("Enter an integer: ");
            num = Integer.parseInt(numStr);
            result = "That number is " + ((num%2 == 0) ? "even" : "odd");
            JOptionPane.showMessageDialog(null, result);
            again = JOptionPane.showConfirmDialog(null, "Do Another?");
        }
        while (again == JOptionPane.YES_OPTION);
    }
}
```

Another example: EvenOdd.java

```java
import javax.swing.JOptionPane;

public class EvenOdd
{
    // Demonstrates the use of the JOptionPane class.
    public static void main (String[] args)
    {
        String numStr, result;
        int num, again;
        do
        {
            numStr = JOptionPane.showInputDialog("Enter an integer: ");
            num = Integer.parseInt(numStr);
            result = "That number is " + ((num%2 == 0) ? "even" : "odd");
            JOptionPane.showMessageDialog(null, result);
            again = JOptionPane.showConfirmDialog(null, "Do Another?");
        }
        while (again == JOptionPane.YES_OPTION);
    }
}
```

**for**: a loop with built in “counter”

```
for (int count = 0; count < 5; count++)
{
    System.out.println(count);
    count++;  
}
```
Selection and Repetition, revisited

**for**: a loop with built in “counter”

- **Initialization**: `int count = 0;`
- **Condition**: `count < 5`
- **Increment**: `count++`

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

The **for** Statement

- A **for** loop is functionally equivalent to the following while loop structure:
  ```java
  int count = 0;
  while (count < 5)
      {
        System.out.println (count);
        count++;
      }
  ```

The **for** Statement - Example

- A **for statement** has the following syntax:
  ```java
  for (initialization; condition; increment)
      statement;
  ```

- The **initialization** is executed once before the loop begins.
- The **condition** is tested after the initialization and again before each iteration.
- The **increment** is executed at the end of each iteration.

- The **statement** is executed while the **condition** is true.

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

Try this:

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

```java
for (int i=2; i<=100; i+=2)
    System.out.println (i);
```

**Example: ReverseNumberAsString.java**

```java
//********************************************************************
// ReverseNumberAsString.java  Author: MAP
// Demonstrates the use of a for loop.
//********************************************************************
import java.util.Scanner;
public class ReverseNumberAsString
{
    //-----------------------------------------------------------------
    // Reverses the digits of an integer viewed as a String.
    //-----------------------------------------------------------------
    public static void main (String[] args)
    {
        int number;
        String reverse = "";
        Scanner scan = new Scanner (System.in);
        System.out.print ("Enter a positive integer: ");
        number = scan.nextInt();
        String original = Integer.toString(number);
        for (int i=0; i<original.length(); i++)
            reverse = original.charAt(i) + reverse;
        System.out.println ("That number reversed is " + reverse);
    }
}
```

Sample Run

Enter a positive integer: 2896
That number reversed is 6982

**Example: Stars.java**

```java
//********************************************************************
// Stars.java  Author: Lewis/Loftus
// Demonstrates the use of nested for loops.
//********************************************************************
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 (" ");
        }
    }
}
```

Output

```
* 
** ***
**** ****
***** *****
****** ******
```

Exercise: can you make it print the row number in (1, 2, 3 …) at the beginning of each line?