Two operators / and % - under the context of integer division
\[ m \div n : m \text{ divided by } n \]
\[ m \mod n : \text{the remainder after } m \text{ is divided by } n \]

For example: 7 / 2 = 3, 7 % 2 = 1

Equality/Relational/Logical Operators – section 3.2 & section 3.4
1) Equality Operators: ==, !=
2) Relational Operators: <, <=, >, >=
3) Logical Operators: !, &&, ||

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>Logical NOT</td>
<td>! a</td>
<td>true if a is false and false if a is true</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>Logical AND</td>
<td>a &amp;&amp; b</td>
<td>true if a and b are both true and false otherwise</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Logical OR</td>
</tr>
</tbody>
</table>

For example
0 ≤ price ≤ 100: ((price >= 0) && (price <= 100))
num is a number less than 0 but not equal to –1: ((num < 0) && (num != -1))
price < 0 or price > 100: ((price < 0) || (price > 100))

IF Statement

- An if statement allows a program to choose whether or not to execute a particular statement.
  
  ```java
  if ( grade > 90) //enclose the Boolean expression, or condition, in parenthesis.
  num = num +1;
  ```

- An if-else statement allows a program to do one thing if a condition is true and a different thing if the condition is false.
  
  ```java
  if ( num1 < num2)
  min = num1;
  else
  min = num2;
  ```

- Using block statement to execute more than one statement as the result of evaluating a Boolean condition. (a block is a list of statements enclosed in braces).
  
  ```java
  if ( num1 < num2) {
  min = num1;
  max = num2;
  }
  else {
  min = num2;
  ```
max = num1;
}

- In a nested if statement, an else clause is matched to the closest unmatched if.

if (num1 < num2)
  if (num1 < num3)
    min = num1;
  else
    min = num3;
else
  if (num2 < num3)
    min = num2;
  else
    min = num3;

The WHILE Statement
- A while statement allows a program to execute the same statement multiple times.

Line 1:
Line 2: while (a condition)
Line 3: {
Line 4:
Line 5: }
Line 6:

The order of execution

Example 1 – calculate the total credits for all courses you take, input –1 to end the calculation
//initialize totalCredits to zero
totalCredits = 0;

//read the credit for the first course
str=stdin.readLine(); //Reads in string of characters from user
credit=Integer.parseInt(str); //Converts string to integer

//calculate the total credits for all courses, if credit==-1, that means there is no course left
while(credit != -1){
  //add the credit of this course to the total credits
  totalCredits = totalCredits + credit;

  //read the credit for the next course
  str=stdin.readLine();
  credit=Integer.parseInt(str);
}
// Display the total credits on the screen
System.out.println("The total credits is " + totalCredit);

**Example 2** – Print 10,9,8,7,6,5,4,3,2,1 on the screen.
count = 10;
while (count > 0) {
    System.out.println(count);
    count = count – 1;
}
System.out.println("end!");

Question: if delete the statement “count = count – 1”, what will happen?