Chapter 3 Selections

Introduction to Computer Science

Motivations

Scanner scanner = new Scanner(System.in);
// read in a radius
double radius = scanner.nextDouble();
// Compute area
double area = radius * radius * 3.14159;
// Display results
System.out.println("The area for the circle of radius "+
radius + " is "+ area);

What would happen if a negative value for radius was entered above?

Objectives

- Declare and use boolean types
- Compare values using relational operators
- Write Boolean expressions using logic operators
- Implement selection control using if statements
- Implement selection control using switch statements

boolean Type

- boolean type is used to represent one of the 2 truth values - true or false
- A logical value, or a truth value, is a value indicating the extent to which a proposition is true.
- The result of a comparison is a boolean type
  - 6/3 > 2
  - radius > 0
- Boolean algebra (or Boolean logic) is a logical calculus of truth values, developed by George Boole

The Boolean Expressions

- Use a boolean variable
  - boolean var; // declare a boolean variable
  - var = expression; // assign a boolean value
  - boolean var = expression; // declare and assign
- A Boolean expression is an expression that evaluates to a Boolean value
- Comparison operators: compare a pair of values (numbers, characters, boolean values)
  - boolean validInput = radius > 0;
- Boolean operators: perform logic operations (boolean values)
  - boolean validInput = (radius > 0) || (radius<10);

Comparison Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>less than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal to</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal to</td>
</tr>
<tr>
<td>==</td>
<td>equal to</td>
</tr>
<tr>
<td>!=</td>
<td>not equal to</td>
</tr>
</tbody>
</table>
Boolean Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>!</td>
<td>not</td>
</tr>
<tr>
<td>&amp;&amp;</td>
<td>and</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>^</td>
<td>exclusive or</td>
</tr>
</tbody>
</table>

Truth Table for Operator !

<table>
<thead>
<tr>
<th>p</th>
<th>!p</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
</tr>
</tbody>
</table>

Example

!(1 > 2) is true, because !(1 > 2) is false.
!(1 > 0) is false, because !(1 > 0) is true.

Truth Table for Operator &&

<table>
<thead>
<tr>
<th>p1</th>
<th>p2</th>
<th>p1 &amp;&amp; p2</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>true</td>
</tr>
</tbody>
</table>

Example

(3 > 2) && (5 > 5) is true, because (3 > 2) and (5 > 5) are both true.
(3 > 2) && (5 > 5) is false, because (5 > 5) is false.

Truth Table for Operator ||

| p1 | p2 | p1 || p2 |
|----|----|-----|-----|
| false | false | false |
| false | true | false |
| true | false | true |
| true | true | true |

Example

(2 > 3) || (6 > 5) is true, because (2 > 3) and (6 > 5) are both true.
(1 > 2) || (6 > 5) is true, because (1 > 2) is true.

Truth Table for Operator ^

<table>
<thead>
<tr>
<th>p1</th>
<th>p2</th>
<th>p1 ^ p2</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>false</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>false</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>false</td>
</tr>
</tbody>
</table>

Example

(2 > 3) ^ (6 > 5) is true, because (2 > 3) is false and (6 > 5) is true.
(1 > 2) ^ (6 > 5) is false, because both (1 > 2) and (6 > 5) are true.

Example – TestBoolean

Scanner input = new Scanner(System.in);
int number = input.nextInt();

System.out.println("Is " + number + " divisible by 2 and 3? |
|{(number % 2 == 0) && (number % 3 == 0)}|);
System.out.println("Is " + number + " divisible by 2 or 3? |
|{(number % 2 == 0) || (number % 3 == 0)}|);
System.out.println("Is " + number + " divisible by 2 or 3, but not both? " |
|{(number % 2 == 0) ^ (number % 3 == 0)}|);

TestBoolean.java
Boolean Expressions Evaluation

- Java uses shortcut evaluation - evaluation of boolean expression stops as soon as the result is known, which makes code execute faster than if all boolean operands were evaluated.

```java
int x = 0;
if (x > 0 && x++ < 5) x = x - 1;
```

Problem: Determining Leap Year?

- Write a program that determines if given year is a leap year
  1. Read input year
  2. Compute whether input year is a leap year - a year is a leap year if it is divisible by 4 but not by 100 or if it is divisible by 400
  3. Display the result

LeapYear.java

Problem: A Simple Math Learning Tool

- Write a program to let a first grader practice additions
  - The program should randomly generate two single-digit integers number1 and number2 and display a question such as “What is 7 + 9?” to the student
  - After the student types the answer, the program displays a message to indicate whether the answer is true or false.

Selection Statements

- if statement
- switch statement
- Conditional operators

Simple if Statements

```java
if (booleanExpression) {
    statement(s);
}
```

Flow Chart

- A flowchart is a schematic representation of an algorithm or a process.
  - Start and end symbols, represented as rounded rectangles usually containing the word “Start” or “End”
  - Arrows, showing “flow of control”.
  - Processing steps, represented as rectangles.
  - Input/Output, represented as a parallelogram.
  - Conditional (or decision), represented as a diamond.
    - Typically contain a Yes/No question or True/False test.
    - Has two arrows coming out of it, usually from the bottom point and right point, one corresponding to Yes or True, and one corresponding to No or False.
Flow Chart Example

Testing whether a number is even or odd

if (number % 2 == 0) {
    System.out.println(number + " is even");
} else {
    System.out.println(number + " is odd");
}

Caution – if condition

if (i > 0) && (i < 10) {
    int a = 10;
    a = a / 3;
    System.out.println("i is between 0 and 10");
}

if ((i > 0) && (i < 10)) {
    System.out.println("i is between 0 and 10");
}

if ((i > 0) && (i < 10)) {
    int a = 10;
    a = a / 3;
    System.out.println("i is between 0 and 10");
}

Caution – Block statement

Note – Block statement

Caution – Extra semicolon

Adding a semicolon at the end of an if clause is a mistake.

if (radius > 0) {
    area = radius*radius*PI;
    System.out.println("The area for the circle of radius " + radius + " is " + area);
}

if (radius >= 0); {
    area = radius*radius*PI;
    System.out.println("The area for the circle of radius " + radius + " is " + area);
}
The if...else Statement

```java
if (booleanExpression) {
    statement(s)-for-the-true-case;
} else {
    statement(s)-for-the-false-case;
}
```

if...else Example

```java
if (radius >= 0) {
    area = radius * radius * 3.14159;
    System.out.println("The area for the circle " +
            "of radius " + radius + " is " + area);
} else {
    System.out.println("Negative input");
}
```

Multiple Alternative if Statements

```java
if (score >= 90.0)
    grade = 'A';
else if (score >= 80.0)
    grade = 'B';
else if (score >= 70.0)
    grade = 'C';
else if (score >= 60.0)
    grade = 'D';
else
    grade = 'F';
```

Trace if-else statement

```java
if (score >= 90.0)
    grade = 'A';
else if (score >= 80.0)
    grade = 'B';
else if (score >= 70.0)
    grade = 'C';
else if (score >= 60.0)
    grade = 'D';
else
    grade = 'F';
```

Trace if-else statement

```java
if (score >= 90.0)
    grade = 'A';
else if (score >= 80.0)
    grade = 'B';
else if (score >= 70.0)
    grade = 'C';
else if (score >= 60.0)
    grade = 'D';
else
    grade = 'F';
```

Trace if-else statement

```java
if (score >= 90.0)
    grade = 'A';
else if (score >= 80.0)
    grade = 'B';
else if (score >= 70.0)
    grade = 'C';
else if (score >= 60.0)
    grade = 'D';
else
    grade = 'F';
```

Trace if-else statement

```java
if (score >= 90.0)
    grade = 'A';
else if (score >= 80.0)
    grade = 'B';
else if (score >= 70.0)
    grade = 'C';
else if (score >= 60.0)
    grade = 'D';
else
    grade = 'F';
```
Trace if-else statement

Suppose score is 70.0

if (score >= 90.0)
grade = 'A';
else if (score >= 80.0)
grade = 'B';
else if (score >= 70.0)
grade = 'C';
else if (score >= 60.0)
grade = 'D';
else
grade = 'F';

grade is C

Suppose score is 70.0
Exit the if statement

Note, cont.

- To force the else clause to match the first if clause, you must add a pair of braces

```java
int i = 1;
int j = 2;
int k = 3;
if (i > j) {
    if (i > k)
        System.out.println("A");
    else
        System.out.println("B");
} else
    System.out.println("D");
```

This statement prints B.

TIP – Assigning value to a boolean variable

```java
boolean even;
if (number % 2 == 0)
even = true;
else
    even = false;
```

boolean even = number % 2 == 0;

TIP – testing a boolean variable

```java
if (even)
System.out.println("It is even.");
```

(a) Equivalent

```java
if (even == true)
System.out.println("It is even.");
```

(b)
Conditional Operator

if (booleanExp)
    var = exp1;
else
    var = exp2;

if (x > 0)
    y = 1
else
    y = -1;

---

switch Statement

switch (dayOfWeek) {
    case 0:  System.out.println("Sunday");
             break;
    case 1:  System.out.println("Monday");
             break;
    case 2:  System.out.println("Tuesday");
             break;
    case 3:  System.out.println("Wednesday");
             break;
    ...
    default: System.out.println("Errors: invalid day");
}

---

switch Statement Flow Chart

switch (dayOfWeek) {
    case 0:  System.out.println("Sunday");
             break;
    case 1:  System.out.println("Monday");
             break;
    case 2:  System.out.println("Tuesday");
             break;
    case 3:  System.out.println("Wednesday");
             break;
    ...
    default: System.out.println("Errors: invalid day");
}

---

Caution: switch statement

Suppose ch is 'a':

Don't forget the break statement!

Once a case is matched, the statements are executed until a break statement or the end of switch statement is reached

switch (ch) {
    case 'a': System.out.println(ch);
    case 'b': System.out.println(ch);
    case 'c': System.out.println(ch);
}
Formatting Output

- Problem: display floating point number with two digits after decimal point
- Previous solution:
  ```java
double x = 12.2223345322;
System.out.println("Number is "+ (int) (x * 100) / 100.0);
```
- Good solution: use printf and specify a format
  ```java
double x = 12.2223345322;
System.out.printf("Number is %.2f", x);
```

## Operator Precedence (operators order)

- What is the value of the expression:
  ```java
  3 + 4 * 4 > 5 * (4 + 3) - 1
  ```
  - The expression in parenthesis is evaluated first
    ```java
    (2 % 13 + 4) > 3 + 2
    ```
    - Evaluate subexpressions in parenthesis
      ```java
      2 % 13 + 4
      ```
    - When expression does not have parenthesis, certain rules are applied during evaluation

## Syntax:

```java
System.out.printf(format, item1, item2, ... , itemk);
```

### Examples:

<table>
<thead>
<tr>
<th>int n = 100;</th>
</tr>
</thead>
<tbody>
<tr>
<td>int i = 0;</td>
</tr>
<tr>
<td>. . . Change value of i . . .</td>
</tr>
<tr>
<td>System.out.printf(&quot;You completed %d out of %d tasks&quot;, i, n);</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>String name = &quot;Steve&quot;;</th>
</tr>
</thead>
<tbody>
<tr>
<td>double balance;</td>
</tr>
<tr>
<td>. . . Change value of balance . . .</td>
</tr>
<tr>
<td>System.out.printf(&quot;Hello %s, your balance is %.2f&quot;, name, balance);</td>
</tr>
</tbody>
</table>

### Frequent-Used Format Specifiers

A format specifier consists of a percent sign and a conversion code

<table>
<thead>
<tr>
<th>Specifier</th>
<th>Output</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>%b</td>
<td>a boolean value</td>
<td>true or false</td>
</tr>
<tr>
<td>%c</td>
<td>a character</td>
<td>'a'</td>
</tr>
<tr>
<td>%d</td>
<td>a decimal integer</td>
<td>200</td>
</tr>
<tr>
<td>%f</td>
<td>a floating-point number</td>
<td>45.460000</td>
</tr>
<tr>
<td>%e</td>
<td>a number in standard scientific notation</td>
<td>4.556000e+01</td>
</tr>
<tr>
<td>%s</td>
<td>a string</td>
<td>&quot;Java is cool&quot;</td>
</tr>
</tbody>
</table>

### Operator Precedence

- `var++`, `var--`
- `+,- (Unary plus and minus), ++var,--var`
- `(type) Casting`
- `! (Not)`
- `*`, `/` (Multiplication, division, and remainder)
- `+,- (Binary addition and subtraction)`
- `<, <=, >, >= (Comparison)`
- `==, !=` (Equality)
- `^ (Exclusive OR)`
- `|| (Conditional OR) Short-circuit OR`
- `&& (Conditional AND) Short-circuit AND`
- `=, +=, -=, *=, /=, %= (Assignment operator)`

### Operator Associativity

- When two operators with the same precedence are evaluated, the associativity of the operators determines the order of evaluation
- All binary operators except assignment operators are left-associative
  ```java
  ((a - b) + c) - d = a - b + c - d
  ```
- Assignment operators are right-associative
  ```java
  a = (b += (c = 5)) = a = b += c = 5
  ```
Example

\[3 + 4 \times 4 > 5 \times (4 + 3) - 1\]

(1) inside parentheses first
(2) multiplication
(3) multiplication
(4) addition
(5) subtraction
(6) greater than

\[3 + 16 > 35 - 1\]

\[19 > 34\]

false

HINT

- If not sure what gets evaluated first, ALWAYS use parenthesis to make the expression to be evaluated as you want

\[
\text{double value} = i++ + x \times 4 \ % \ n;
\]

VS

\[
\text{double value} = (i++) + (x \times 4) \ % \ n;
\]

Problem: Lottery

Write a program that implements a lottery

- Generate a random number between 0-99
- Prompt the user to enter a guess between 0-99
- Check the guess and give out prize using the following rule:
  - If user’s number has the same digits as the generated number, the prize is $3,000
  - If user’s number has one digit that appears in the generated number, the prize is $1,000
  - Otherwise there is no prize

Lottery.java

Problem: A Subtraction Quiz Tool

Write a program to teach a first grade child to learn subtractions

- Randomly generates two single-digit integers number1 and number2
- If number 1 is smaller than number 2, swap number1 with number2
- Displays a question for number1 – number 2 and reads the student’s answer
- Checks the answer and displays a message to indicate whether the answer is correct.

SubtractionQuiz.java

Problem: Guessing Birth Day

Write a program that can guess your birth day using the following algorithm:

GuessBirthDate.java

(GUI) Confirmation Dialogs

int option = JOptionPane.showConfirmDialog(null, "Continue?");

GuessBirthDateUsingConfirmationDialog.java