Selection Statements
Outline

- Block statement

- Branching Statements
  - Simple if statement
  - if-else statement
  - if-else-elseif statement
  - switch statement

- Nested if statements
- Block Statement

- A block statement consists of one or more Java statements enclosed in braces.

- Example of a block statement:

```
{
    statement 1;
    statement 2;
    ...
    statement n;
}
```

- Blocks can be nested.

- A block statement can be used anywhere that a single statement can be used.
- Branching Statement

- A branching statement consists of one or more block statements.

- The execution of a block statement in a branching statement is controlled by a boolean expression which we call a condition.

- There are mainly the following four types of branching statement.
  - Simple if
  - If-else
  - If-elseif-else
  - switch
Simple if statement has the following structure:

```java
if ( <boolean_expression> ) {
    <then_block>
}
```

- The `boolean_expression` must be enclosed in parentheses.
- If the `boolean_expression` is true, then the `then_block` is executed. Otherwise it will NOT execute.

Example:

```java
if ( testScore >= 95 ) {
    System.out.println("You are a good student");
}
```
-- Simple if Statement ...

**Control Flow of if:**

- `testScore >= 95?`
  - **true**
    - `System.out.println("You are a good student");`
  - **false**
    - 

March 23, 2008
ICS102: The course Chapter 6 - 6
Example: Design and write a Java program prints the absolute value of a number.

```java
import java.util.Scanner;
class absolute {
    public static void main(String [] args) {
        Scanner kb = new Scanner(System.in);
        System.out.print("Enter a number: ");
        double x = kb.nextDouble();
        double y = x;
        if( y < 0) {
            y = -y;
        }
        System.out.print("The absolute value of " + x + " is " + y);
    }
}
```
-- if-else Statement ...

- An if-else statement chooses between two alternative statements based on the value of a Boolean expression.
- If the Boolean expression is true, then the then block is executed, otherwise the else block is executed.

```java
if (boolean_expression) {
    then_block
} else {
    else_block
}
```

```java
if (testScore < 50) {
    System.out.println("You did not pass");
} else {
    System.out.println("You did pass");
}
```
-- if-else Statement ...

```
if (testScore < 50)
    System.out.println("You did not pass");
else
    System.out.println("You did pass");
```
Compound Statements

- You have to use braces if the <then> or <else> block has multiple statements.
- If only one statement is there, braces are optional but it is advisable to always use them to enhance readability.

```java
if (testScore < 70) {
    System.out.println("You did not pass");
    System.out.println("Try harder next time");
} else {
    System.out.println("You did pass");
    System.out.println("Keep up the good work");
}
```
Design and write a Java program which prints the difference of two numbers.

```java
Scanner kb = new Scanner(System.in);
System.out.println("Enter the value of the first number: ");
double first = kb.nextDouble();
System.out.println("Enter the value of the second number: ");
double second = kb.nextDouble();
if (first > second)
{
    double diff = first - second;
    System.out.println(diff);
}
else
{
    double diff = second - first;
    System.out.println(diff);
}
```
Nested if Statements...

- One of the block statements of a branching statement can be another if statement.
- The inner if statement is executed when the enclosing block statement is executed.
- If statements can be nested to many levels.

If(<boolean_expression_1>)
{
    Statement_1;
    <block_statement_2>;
    Statement_3;
}

if(<boolean_expression_2>)
{
    Statement_2A;
    Statement_2B;
}

> Statement_2A and 2B are only executed if boolean_expression_1 and 2 are true.
if (testScore >= 50) {
    if (studentAge < 10) {
        System.out.println("You did a great job");
    } else {
        System.out.println("You did pass");
    }
} else { //test score < 50
    System.out.println("You did not pass");
}
Nested if Statements ...

- Nested if Statements ...

false

System.out.println("You did not pass");

true

testScore >= 50 ?

false

System.out.println("You did not pass");

true

System.out.println("You did pass");

false

studentAge < 10 ?

false

System.out.println("You did a great job");

true

System.out.println("You did a great job");
The multiway if-else statement is simply a normal if-else statement that nests another if-else statement at every else branch.

- It is indented differently from other nested statements.
- All of the Boolean_Expressions are aligned with one another, and their corresponding actions are also aligned with one another.
- The Boolean_Expressions are evaluated in order until one that evaluates to true is found.
- The final else is optional.
... -- if-elseif-else Statement ...

if (Boolean_Expression_1)
  Block_Statement_1
else if (Boolean_Expression_2)
  Block_statement_2
    ...
    ...
  ...
else if (Boolean_Expression_n)
  Block_statement_n
else
  Block_statement_For_All_Other_Possibilities
if - else- if

```java
if (score >= 85) {
    System.out.println("Grade is A");
} else {
    if (score >= 75) {
        System.out.println("Grade is B");
    } else {
        if (score >= 65) {
            System.out.println("Grade is C");
        } else {
            if (score >= 50) {
                System.out.println("Grade is D");
            } else {
                System.out.println("Grade is N");
            }
        }
    }
}
```

<table>
<thead>
<tr>
<th>Test Score</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>85 \leq \text{score}</td>
<td>A</td>
</tr>
<tr>
<td>75 \leq \text{score} &lt; 85</td>
<td>B</td>
</tr>
<tr>
<td>65 \leq \text{score} &lt; 75</td>
<td>C</td>
</tr>
<tr>
<td>50 \leq \text{score} &lt; 65</td>
<td>D</td>
</tr>
<tr>
<td>score &lt; 50</td>
<td>N</td>
</tr>
</tbody>
</table>
if (score >= 85) {
    System.out.println("Grade is A");
} else if (score >= 75) {
    System.out.println("Grade is B");
} else if (score >= 65) {
    System.out.println("Grade is C");
} else if (score >= 50) {
    System.out.println("Grade is D");
} else {
    System.out.println("Grade is N");
}
Matching else

```java
if (x < y)
    if (x < z)
        System.out.println("Hello");
    else
        System.out.println("Good bye");
```

really means

```java
if (x < y) {
    if (x < z) {
        System.out.println("Hello");
    } else {
        System.out.println("Good bye");
    }
}
```
Matching else

```java
if (x < y) {
    if (x < z) {
        System.out.println("Hello");
    } else {
        System.out.println("Good bye");
    }
} else {
    System.out.println("Good bye");
}
```

means

```java
if (x < y) {
    if (x < z) {
        System.out.println("Hello");
    }
} else {
    System.out.println("Good bye");
}
```
- The **switch** Statement ...

- The **switch** statement is the only other kind of Java statement that implements *multiway* branching
  
  - When a **switch** statement is evaluated, one of a number of different branches is executed

  - The choice of which branch to execute is determined by a *controlling expression* enclosed in parentheses after the keyword **switch**
    
    - The controlling expression must evaluate to a **char**, **int**, **short**, or **byte**
Syntax for the switch Statement

switch ( <arithmetic expression> ) {
    <case label 1> : <case body 1>
    ... 
    <case label n> : <case body n>
}

switch ( fanSpeed ) {
    case 1:
        System.out.println("That's low");
        break;
    case 2:
        System.out.println("That's medium");
        break;
    case 3:
        System.out.println("That's high");
        break;
}
switch ( N ) {
    case 1: x = 10;
            break;
    case 2: x = 20;
            break;
    case 3: x = 30;
            break;
}
The switch Statement with default

```java
switch ( binaryDigit ) {
    case 0:
        System.out.println("zero");
        break;
    case 1:
        System.out.println("one");
        break;
    default:
        System.out.println("That's not a binary digit");
        break;
}
```

```
switch ( <arithmetic expression> ) {
    <case label 1> : <case body 1>
        ...
    <case label n> : <case body n>
    default: <default body>
}
```
switch ( N ) {
    case 1: x = 10;
    case 2: x = 20;
    case 3: x = 30;
}
double y = 30;
double z = 20;
Scanner kb = new Scanner(System.in);
System.out.println("1. add ");
System.out.println("2. Subtract ");
System.out.println("3. Multiply ");
System.out.println("Enter a value: between 1 and 3 ");
int x = kb.nextInt();
switch (x) {
    case 1:  System.out.println(z + y);
             break;
    case 2:  System.out.println(z - y);
             break;
    case 3:  System.out.println(z * y);
             break;
    default: System.out.println("Wrong Choice.");
             break;
}

March 23, 2008
THE END