Decision Structures

Outline

- Scanner Class
- if-else Statement
- Logical Operators
- Comparing String Objects
- Conditional Operator
- **`switch`** Statement

`Scanner` Class (1)

- To read input from the keyboard we can use the **`Scanner`** class.
- The **`Scanner`** class is defined in **`java.util`**, so we will use the following statement at the top of our programs:

```
import java.util.Scanner;
```

`Scanner` Class (2)

- Scanner objects work with **`System.in`**
- To create a **`Scanner`** object:

```
Scanner keyboard = new Scanner (System.in);
```

- Popular **`Scanner`** class methods
  - nextDouble()
  - nextInt()
  - nextLine()
- Example

```
int hourlyWage;
hourlyWage = keyboard.nextInt();
```

UNLV

`If-else` Statement (1)

- The `if` statement uses a **`boolean`** to decide whether the next statement or block of statements executes.

```
if (boolean expression is true) {
    execute statement1;
    execute statement2;
    execute statement3;
}
```

- Note that when the curly braces are not used, then only the next statement after the `if` condition will be executed conditionally.

`If-else` Statement (2)

- The **`if-else`** statement adds the ability to conditionally execute code when the **`if`** condition is false.

```
if (expression) {
  statementOrBlockIfTrue1;
  statementOrBlockIfTrue2;
}
else {
  statementOrBlockIfFalse1;
  statementOrBlockIfFalse2;
  }
```

`If-else` Statement (3)

- A *boolean expression* is any variable or calculation that results in a *true* or *false* condition.

Expression	Meaning
`x > y`	Is x greater than y?
`x < y`	Is x less than y?
`x >= y`	Is x greater than or equal to y?
`x <= y`	Is x less than or equal to y.
`x == y`	Is x equal to y?
`x != y`	Is x not equal to y?

`If-else` Statement (4): Example

```
if (x > y)
   System.out.println("X is greater than Y");

if(x == y)
   System.out.println("X is equal to Y");

if(x != y){
   System.out.println("X is not equal to Y");
   x = y;
   System.out.println("However, now it is.");
}
```

`If-else` Statement (5): Example

```
if(x != y)
   System.out.println("X is not equal to Y");
   x = y;
   System.out.println("However, now it is.");
```

Only the first statement is executed conditionally!!

UNLV

Lab (1)

- Employee
- EmployeeDemo2

UNLV

Exercise (1)

- Please revise the calSalary() method in the Employee class
  - If the employee's working hours is more than the regular hours, calculate the overtime pay for the overtime hours as 1.8 of the base pay.

`if-else-if` Statements (1)

- The **`if-else-if`** statement makes certain types of nested decision logic simpler to write.
- Care must be used since **`else`** statements match up with the immediately preceding unmatched **`if`** statement.

`if-else-if` Statements (2)

```
if (expression_1){
    statement;
    statement;
    etc.
}
else if (expression_2){
    statement;
    statement;
    etc.
}
```

If `expression_1` is true these statements are executed, and the rest of the structure is ignored.

Otherwise, if `expression_2` is true these statements are executed, and the rest of the structure is ignored.

Insert as many else if clauses as necessary

```
else{
    statement;
    statement;
    etc.
}
```

These statements are executed if none of the expressions above are true.

Lab (2)

- GradePointCalculator.java
  - The user enters either A, B, or C, and the program prints out the grade point accordingly.

Logical Operators (1)

- Java provides two binary *logical operators* (`&&` and `||`) that are used to combine **`boolean`** expressions.
- Java also provides one *unary* (`!`) logical operator to reverse the truth of a **`boolean`** expression.

Logical Operators (2)

Operator	Meaning	Effect
&&	**AND**	Connects two **`boolean`** expressions into one. Both expressions must be true for the overall expression to be true.
||	**OR**	Connects two **`boolean`** expressions into one. One or both expressions must be true for the overall expression to be true. It is only necessary for one to be true, and it does not matter which one.
!	**NOT**	The ! operator reverses the truth of a **`boolean`** expression. If it is applied to an expression that is true, the operator returns false. If it is applied to an expression that is false, the operator returns true.

Lab (3)

- LogicalAnd.java
  - In this program, we ask the user to input the number of hours an employee worked, and the points earned las week. If the number of hours is greater than the regular hour AND the current exp is greater than 200, than the employee is qualified for the bonus.

Exercise (2)

- Revise GradePointCalculator.java
  - Please revise the program so that it will handle both uppercase and lowercase letter grade.

Logical Operators (3)

- Order of Precedence
  - The `!` operator has a higher order of precedence than the `&&` and `||` operators.
  - The `&&` and `||` operators have a lower precedence than relational operators like `<` and `>`.
- Parenthesis can be used to force the precedence to be changed.

Logical Operators (4)

Order of Precedence	Operators	Description
1	**(unary negation) !**	Unary negation, logical NOT
2	*** / %**	Multiplication, Division, Modulus
3	**+ -**	Addition, Subtraction
4	**< > <= >=**	Less-than, Greater-than, Less-than or equal to, Greater-than or equal to
5	**== !=**	Is equal to, Is not equal to
6	**&&**	Logical AND
7	**||**	Logical NOT
8	**= += -= *= /= %=**	Assignment and combined assignment operators.

Exercise: Logic Operator

- Given x = 30, y = 25, what is the result of following expression?

```
!((y - x == 5) && (x > 0))
```

Comparing `String` Objects

- In most cases, you cannot use the logical operators to compare two **`String`** objects.
- In the **`String`** class, the **`equals`** method is case sensitive.
- In order to compare two **`String`** objects that might have different case, use:
  - **`equalsIgnoreCase`**
- Example

```
if (name.equalsIgnoreCase("france"))
    ……
```

Other String Methods

```
3  public class StringMethods {
4
5      public static void main(String[] args) {
6          String message ="Java is great fun!";
7
8          String lowercaseMesassge=message.toLowerCase();
9          String uppercaseMessage=message.toUpperCase();
10         char letter = message.charAt(2);
11         int stringSize = message.length();
12
13         System.out.println(lowercaseMesassge);
14         System.out.println(uppercaseMessage);
15         System.out.println(letter);
16         System.out.println(stringSize);
17     }
18 }
19
```

Problems @ Javadoc Declaration Console

<terminated> StringMethods [Java Application] C:\eclipse\plugins\org.eclip

```
java is great fun!
JAVA IS GREAT FUN!
v
18
```

UNLV

Lab (4)

- SecretWord.java
  - When the user enters the secret word “professional,” the program prints out a success message; otherwise, print out an error message.

Conditional Operator (1)

- The conditional operator is a ternary (three operand) operator.
  - You can use the conditional operator to write a simple statement that works like an **`if-else`** statement.
- Format

```
BooleanExpression ? Value1 : Value2
```

  - If ***`BooleanExpression`*** is true, the value of the conditional expression is ***`Value1`***.
  - If ***`BooleanExpression`*** is false, the value of the conditional expression is ***`Value2`***.

Conditional Operator (2)

```
z = x > y ? 10 : 5;
```

This line is functionally equivalent to:

```
if(x > y)
  z = 10;
else
  z = 5;
```

- Oftentime the conditional operator is used to supply a value.

UNLV

Conditional Operator Example

```
7   public static void main(String[] args) {
8       // constants
9       final double BASE_FEE = 4.5; // regular charge per pound
10      final double OVERWEIGHT_FEE = 7; // charge for overweight package
11      final double WEIGHT_LIMIT = 20; // the threshold of overweight package
12
13      // variables used in the program
14      double weight; // weight of the package. To be entered by user
15      double charge; // charge of the package to be calculated.
16
17      // keyboard input
18      Scanner kb = new Scanner(System.in);
19
20      // get user input for the weight
21      System.out.print("Please enter the weight of the package: ");
22      weight = kb.nextDouble();
23
24      // calculate the charge based on the weight
25      // use the conditional operator to determine which fee to use in calculation
26      charge= weight * (weight<=WEIGHT_LIMIT? BASE_FEE: OVERWEIGHT_FEE);
27
28      // show result
29      System.out.print("The total charge is: $"+charge);
30  }
```

Problems @ Javadoc Declaration Console

```
<terminated> PackageFee [Java Application] C:\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.
Please enter the weight of the package: 25
The total charge is: $175.0
```

`switch` Statement (1)

- The **`if-else`** statement allows you to make true / false branches.
- The **`switch`** statement allows you to use an ordinal value to determine how a program will branch.

`switch` Statement (2)

```
switch (SwitchExpression){
  case CaseExpression:
    // place one or more statements here
    break;
  case CaseExpression:
    // place one or more statements here
    break;

    // case statements may be repeated
    //as many times as necessary
  default:
    // place one or more statements here
}
```

`switch` Statement (3)

```
switch (SwitchExpression){

  ...

}
```

- The **`switch`** statement will evaluate the *SwitchExpression*, which can be a **`byte`**, **`short`**, **`int`**, **`long`**, **`char, or String`**

UNLV

`switch` Statement (4)

- Each **`case`** statement will have a corresponding *CaseExpression* that must be unique.

```
case CaseExpression:
    // place one or more statements here
    break;
```

- If the *SwitchExpression* matches the *CaseExpression*, the Java statements between the colon and the **`break`** statement will be executed.

`switch` Statement (5)

- **`break`** statement
  - It ends the **`case`** statement.
  - **`break`** statement is optional.
- If a **`case`** does not contain a **`break`**, then program execution continues into the next **`case`**.
- The **`default`** section is optional and will be executed if no *CaseExpression* matches the *SwitchExpression*.

Lab (5)

- GradePointCalculatorWithSwitch.java
  - The user enters either A, B, or C, and the program prints out the grade point accordingly.

Exercise

- Create a program that displays the price of a concert ticket based on users' input. The ticket price is based on the code entered, as shown below. If the user enters a number other than 1 to 4, the application should indicate the problem.

Class	Ticket Price
1	$15
2	$15
3	$25
4	$35
Other	Invalid