WESTERN UNIVERSITY
DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

ES1036B
PROGRAMMING FUNDAMENTALS FOR ENGINEERS
WINTER 2022

Lab05 – Conditional Statements – (ET2+KB4)

Instructor:
DR. QUAZI RAHMAN

Prepared by:
HIRA NADEEM

Lead TA: Hira Nadeem, hnadeem5@uwo.ca

---

1 Goal

In this lab, you will know how the conditional statement is used to create a decision structure, which allows a program to have more than one path of execution. The conditional statement causes one or more statements to execute only when a Boolean expression is true.

2 Resources

Lecture notes:	"Unit 5: Control Statements in Java."
Lecture notes:	"Unit 4: Class and Objects"
Pre-recorded videos:	Session 1: If-statements, IntelliJ tutorial: If-statements

3 Deliverables

One zip file folder containing your project folder. Name it username_Lab3.zip where username is the beginning part of your email (e.g., Dr. Rahman's email address is qrahman3@uwo.ca, and his UWO username is: qrahman3).

Submission deadline: Submit your code by the end of your lab session. Prepare to demonstrate your understanding during the lab session.

4 Good Programming Practices

Below are several programming practices that should be followed to write and maintain quality codes. **Please note that you will be marked on all these components:**

- Include meaningful comments in your program. This will help you remember what each part of your code does, especially after long breaks from your work. Your TA will also appreciate understanding your code by going over your comments.

- Choose meaningful and descriptive names for your variables. There is a balance between descriptive names and code readability but always err on the side of descriptive.

- It is recommended that you follow the naming strategy for variables, methods and class names, as outlined in your course handout (Unit 2). Since the identifiers cannot contain white-space characters (spaces or tabs), words in an identifier should be separated by uppercase letters (myNewFunction(), myNewVariable). For class names, capitalize the first letter of each word in the name (e.g. MyClass, MatrixCalculator). For any constant name, use uppercase letters, and if needed, concatenate two or more words with underscore (e.g., MINIMUM_DRIVING_AGE)

- Initialize variables when declaring them. This means giving them initial values which are easier to track in your program if logical errors are present with your output.

- Indent and properly format your code. You should write your codes so that your teaching assistant can read and understand your code easily.

- **Include a header in each of your java class files**. The header should include your full name, UWO ID number, date the code was written and a brief description of the program in that file. Use any print statement to print the Header information on the screen based on the format below:

```
/*******************************************
* Add your full name here*
* Add your student number*
* Add Date*
* Give a brief description of the task *
*******************************************/

public class MyClass
{
    public static void main(String args[])
    {
      // Your code here
    }
}
```

5 Conditional statement

A conditional statement is a construct that allows a program to perform different computations depending on a Boolean condition. If the condition is true, the program performs one computation; otherwise, the condition is false, and the program performs another computation. The condition can be any Boolean expression, for instance: `a > b`, `i - j == 1` and so on. There are three types of conditional statements. We will give examples of all of them**.**

5.1 A single if statement

The simplest form of the conditional statement is a single if statement. It follows the structure:

```
if (condition) {
    // do something
}
```

In this case, if the condition is true, the action inside the code block is executed. Otherwise, the program skips the action. See the following example. In this example, if the age is less than 3, the program prints "This person is too young". Otherwise, it does nothing.

```
int age = ...; // it has a value
if (age < 3) {
    System.out.println("This person is too young");
}
```

5.2 If-else statement

The if statement can also be extended to do something else if the condition is false. In this code, if the condition is "true", the first code block is executed. Otherwise, the second code block is executed.

```
if (condition) {
    // do something
} else {
    // do something else
}
```

In the example below, the program outputs different text depending on the value of num (even or odd). Note that a whole number is even if it can be divided evenly by 2; otherwise, it's odd. For example, if the value of num is 10, the program outputs "It's an even number". If the value is 11, it outputs "It's an odd number".

```
int num = ...; // num is assigned some value
if (num % 2 == 0) {
    System.out.println("It's an even number");
} else {
    System.out.println("It's an odd number");
}
```

5.3 If-else-if statements

The most general form of the if-else statement consists of multiple conditions and else-branches.

```
if (condition0) {
    // do something
} else if (condition1) {
    // do something else 1
// ...
} else if (conditionN) {
    // do something else N
}
```

The following code outputs recommendations as to what type of computer you should buy depending on your budget. This conditional statement has four branches: dollars < 1000, dollars < 2000, dollars < 100000 and dollars >= 100000. For example, if the value of dollars is 9000, it prints Buy a server.

```
long dollars = ...; // your budget
if (dollars < 1000) {
    System.out.println("Buy a laptop");
} else if (dollars < 2000) {
    System.out.println("Buy a personal computer");
} else if (dollars < 100000) {
    System.out.println("Buy a server");
} else {
    System.out.println("Buy a data center or a quantum computer");
}
```

6 Lab Assignment Questions

In each question's driver method, print a header using the **myHeader** method in the beginning, perform the task, and then print an exit message at the end using the **myFooter** method like you did in Lab 4.

The methods:

- **myHeader** -> Outputs a header with your name, lab number and question number. Recycle the **myHeader** method from Lab 4
- **myFooter** -> Outputs an exit message i.e. `*** Goodbye from YourFullName! ***`. Recycle the **myFooter** method from Lab 4

Use IntelliJ IDE to create a project named ***username*_Lab5.**

5.1 Question 1 (10 marks)

For this question, refer to the slides in "Unit 5: Control Statements", focusing on If-statements and if-else-if statements. You will also need to refer to "Unit 4: Class Fundamentals in Coding" to ensure you understand Classes, Objects, and Methods.

Objective: Write a program to display the scores and letter grades of students.

1. Create a package called **Q1.**
2. Create a Java class called **Student** according to the UML diagram.

Specifications for the Student class

- **Constructor** without arguments - will assign all field values based on the information pertaining to any fictional character of your choice.
- **Constructor** with 3 arguments - will assign all field values based on the information passed in the arguments. Here the arguments need to correspond to your personal information (your name, your student number, and any score of your choice).
- **printInfo()** method prints out the three field values with the help of printf() method that uses field width and justification as shown in the sample output.
  (**Note**: in the sample output, the field width of 15, 15, 3 and 2 have been used for presenting the name, student number, score, and letter grade, respectively, and all the data-items have used left justification.)
- The accessor/getter method **getName()** returns the student's full name (the first name followed by a space followed by the last name), while **getScore()** returns the score of a student.

Student
-name: String **-studentNumber: int** **-score: int**
+Student() **+Student(nm: String, sNum: int, sc: int)** **+printInfo(): void** **+getName() : String** **+getScore(): int** **+getLetterGrade() : String**

- Member method called **getLetterGrade()** will return the letter grade for the corresponding score according to the table given at the end (**Hint**: See Unit 5: Slide: 15-16 in the brief version).

Score	Letter Grade
90 or greater	A+
80-89	A-
70-79	B+
60-69	B-
50-59	C+
40-49	C-
30-39	D
29 or less	F

3. In the same Java package, create a new driver class with the name **StudentGradesBy*YourFirstName***

   Specifications for the driver class
   - Inside the main method, call **myHeader()** after recycling it from the previous lab.
   - Declare a **Student** type reference variable and instantiate it to create an object with the help of the constructor without argument.
   - Declare another **Student** type reference variable and instantiate it using the constructor with three arguments. Choose your full name, student number, and any score of your choice.
   - Now, print the information for both the students by calling the **printInfo()** method for each (see the sample output).
   - Compare the scores of these two students and display which student scored higher (see the sample output).
   - Call your **myFooter()** method to print your custom footer.

4. **Assumption**: score will always be between 0 and 100. No validation required.
   Note: You must use if-else-if statements. See resources.
   - Unit 5 (brief version) (slide 20-23)
   - IntelliJ: Working with If-statements and some Short-cut techniques.

5. Run the program three times as per the following cases. The output should be something like the following Sample Outputs:

Run 1:

```
*****************************************************
   Quazi Rahman
   Lab 5, Question 1
*****************************************************

Name           Student Number Score (Letter Grade)
-------------------------------------------------
Ron Weasley    112233         39 (Letter Grade: D )
Harry Potter   556677         89 (Letter Grade: A-)
-------------------------------------------------
Note: Harry Potter scored higher than Ron Weasley.

*** Goodbye from Quazi Rahman! ***
```

Run 2:

```
*****************************************************
   Quazi Rahman
   Lab 5, Question 1
*****************************************************

Name           Student Number Score (Letter Grade)
-------------------------------------------------
Ron Weasley    112233         91 (Letter Grade: A+)
Harry Potter   556677         89 (Letter Grade: A-)
-------------------------------------------------
Note: Ron Weasley scored higher than Harry Potter.

*** Goodbye from Quazi Rahman! ***
```

Run 3:

```
*****************************************************
   Quazi Rahman
   Lab 5, Question 1
*****************************************************

Name           Student Number Score (Letter Grade)
-------------------------------------------------
Ron Weasley    112233         71 (Letter Grade: B+)
Harry Potter   556677         71 (Letter Grade: B+)
-------------------------------------------------
Note: Their scores are equal!

*** Goodbye from Quazi Rahman! ***
```

5.2 Question 2 (15 marks)

For this question, refer to the slides in "Unit 5: Control Statements", focusing on If-statements and if-else-if statements. You will also need to refer to "Unit 4: Class Fundamentals in Coding" to ensure you understand Classes, Objects, and Methods. Also, review "Unit 2: Part 5" for a review of the Java Math class.

Background: Complex Numbers are written in the form a + bi, where a is the real part, b is the imaginary part of the complex number, and i = $\sqrt{-1}$. When adding complex numbers, we add the real parts together and add the imaginary parts together. If there are two complex number x = 5 + 2i and y = 4 – 3i, then, x + y = (5+4) + (2 - 13)i = 9 – 11i.

Objective: Write a program to perform arithmetic operations (addition and multiplication) on complex numbers.

1. Create a package called **Q2.**
2. Create a java class called **ComplexNumber,** as shown in the UML diagram.

ComplexNumber
-real: double **-imaginary: double**
+ComplexNumber() **+ComplexNumber (re: double, im: double)** **+getMagnitude(): double** **+getAngle(): double** **+displayRecForm():void**

Specifications for the ComplexNumber Class:

- The **constructor** without argument will assign all field values to zero.
- The **constructor with parameter** should accept the real and imaginary values as arguments and assign these values to the fields (**real** and **imaginary**).
- No accessor/getter or mutator/setter method is required for this problem.
- The **getMagnitude()** method is a helper method which will return the magnitude of a complex number. Magnitude of a complex number: a+bi = $\sqrt{a^2 + b^2}$ (Use Java's Math class for **Math.pow()** and **Math.sqrt()** methods)
- The **getAngle()** method is a helper method which will return the angle of a complex number in degrees. Angle of a complex number: a+bi = $\tan^{-1}\frac{b}{a}$ (Here, **i** is a symbol for the imaginary part of the number. Use Java's Math class for **Math.atan2(b, a)** that returns the angle in radians, and use **Math.toDegrees(x)** method to convert the radians to degrees. See Unit 2, Part 5: slide 34. All the trigonometric methods in the Math class return the results in radians, and to get the result in degrees one must use **Math.toDegree(x)** method)
- The **displayRecForm()** method will display the complex number as shown in the sample output. For example, if the complex number is (a + bi), it should display a + bi with the corresponding a and b values. The displayed result should use up to 2 decimal places.

6. In the same Java package, create a new driver class with the name **ComplexNumberDemo*YourFirstName*.** If your first name is Quazi, the name of this class should be **ComplexNumberDemoQuazi.** This class will contain three public-static methods:
   - header method (recycle from the previous lab and revisit)
   - driver method
   - footer method (recycle from the previous lab and revisit)

   Specifications for the Driver Class:
   - Call **myHeader()** after recycling it from the previous lab.
   - Declare two **ComplexNumber** type reference variables **x** and **y**.
   - Ask the user to enter the real and imaginary values for each of the two numbers x and y. Get those inputs with the help of a Scanner reference variable.
   - Instantiate both the **ComplexNumber** type reference variables **x** and **y** with the above entered values.
   - Display the two complex numbers using the a + bi form as in the sample output.
   - Declare a 3rd complex reference variable called **addRes.**
   - Add the complex reference variables **x** and **y** and instantiate **addRes** with the resultant values as arguments.
     Note: If there are two complex numbers x = a + bi, and y = c + di, then, x + y = (a + c) + (b + d) i.
   - Declare a 4th complex reference variable called **mulRes.**
   - Multiply the complex reference variables **x** and **y** and instantiate **mulRes** with the resultant values as arguments.
     Note: If there are two complex numbers x = a + bi, and y = c + di, then, x * y = (ac - bd) + (bc + ad) i
   - Using proper method calls and appropriate print statements, display the results as shown in the sample output.
   - Call the **myFooter()** method and end the program.
   - See your instructor/TA if you have any issues with the code.

Note: To test your code with more data-points, use the following website (the result may vary slightly in the decimal points while you are comparing, but most of the cases your result will be exactly the same): https://www.omnicalculator.com/math/cartesian-to-polar

7. The output should be something like the following Sample Outputs:

```
*******************************************************
   Quazi Rahman
   Lab 5, Question 2
*******************************************************
Enter x's real value: -12.2
Enter x's imaginary value: 6.78
Enter y's real value: 3.4512
Enter y's imaginary value: -2.7
You have entered the following two complex numbers x and y:
x = -12.20 + 6.78i
y = 3.45 - 2.70i
Here are the results of the arithmetic operations:
x + y = -8.75 + 4.08i, Magnitude: 9.65, Angle: 155.00 degrees
x * y = -23.80 + 56.34i, Magnitude: 61.16, Angle: 112.90 degrees

*** Goodbye from Quazi Rahman! ***
```

7 Submission

Zip the project file, name it ***username_Lab5.zip*** and submit in OWL by the end of your lab session.

If you have any questions about the Lab Handout, please contact the Lead TA, Hira Nadeem at hnadeem5@uwo.ca