COMP90041
Programming and Software Development
2020 - Semester 1
Lab 3 - Week 4

Yuhao(Beacon) Song

yuhsong1@unimelb.edu.au

Introduction

- Timetable
  - Tue(11) 14:15-15:15 (Melbourne Time) Join URL: https://unimelb.zoom.us/j/490084146
  - Tue(07) 16:15-17.15 (Melbourne Time) Join URL: https://unimelb.zoom.us/j/291505765

- Contact
  - yuhsong1@unimelb.edu.au
  - Github: https://github.com/Beaconsyh08/COMP90041_Programming_and_Software_Development_Tutorials.git

Outline

- Lecture Review
- Exercise & demo
- ProjectA

Pitfall: Using == with Strings

- The equality comparison operator (==) can correctly test two values of a **primitive type**
- However, when applied to two objects such as objects of the **String** class, == tests to see if they are stored in the same memory location, not whether or not they have the same value → (wk5)
- In order to test two strings to see if they have equal values, use the method equals, or equalsIgnoreCase

string1.equals(string2)

string1.equalsIgnoreCase(string2)

Primitive Types

Display 1.2 **Primitive Types**

TYPE NAME	KIND OF VALUE	MEMORY USED	SIZE RANGE
`boolean`	`true` or `false`	1 byte	not applicable
`char`	single character (Unicode)	2 bytes	all Unicode characters
`byte`	integer	1 byte	−128 to 127
`short`	integer	2 bytes	−32768 to 32767
`int`	integer	4 bytes	−2147483648 to 2147483647
`long`	integer	8 bytes	−9223372036854775808 to 9223372036854775807
`float`	floating-point number	4 bytes	$-3.40282347 \times 10^{+38}$ to $-1.40239846 \times 10^{-45}$
`double`	floating-point number	8 bytes	$\pm 1.76769313486231570 \times 10^{+308}$ to $\pm 4.94065645841246544 \times 10^{-324}$

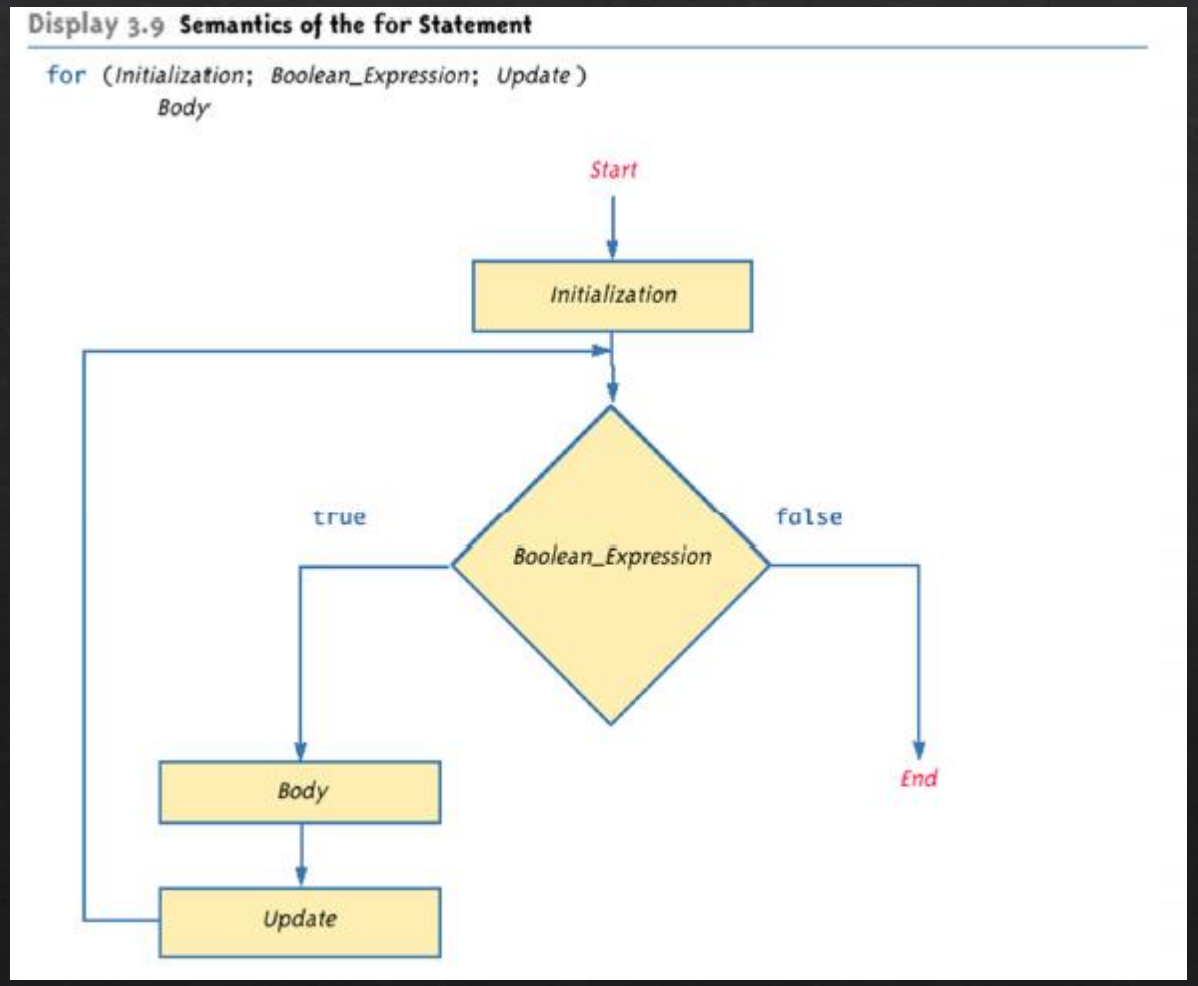
Branching with an if-else Statement

• An if-else statement chooses between two alternative statements based on the value of a Boolean expression

```
if (Boolean_Expression)
    Yes_Statement;
else
    No_Statement;
```

– The Boolean_Expression must be enclosed in **parentheses**
– If the Boolean_Expression is true, then the Yes_Statement is executed
– If the Boolean_Expression is false, then the No_Statement is executed

The for Statement Syntax



for (Initializing; Boolean_Expression; Update)
Body

- The Body may consist of a single statement or a list of statements enclosed in a pair of braces ({ })

- Note that the three control expressions are separated by **two**, not three, semicolons

- Note that there is **no semicolon** after the closing parentheses at the beginning of the loop

While Statement

```
while (Boolean_Expression)
    Statement
```

Or

```
while (Boolean_Expression)
{
    Statement_1
    Statement_2
        .
    Statement_Last
}
```

- A **while statement** is used to repeat a portion of code (i.e., the loop body) based on the evaluation of a Boolean expression
  - The **Boolean expression** is checked **before** the loop body is executed

- When false, the loop body is not executed at all

- If true, the loop body is executed again
- If false, the loop statement ends

→ The loop body can consist of a single statement, or multiple statements enclosed in a pair of braces ({ })

The break and continue Statements

- The **break statement** consists of the keyword **break** followed by a semicolon
  - When executed, the break statement ends the nearest enclosing switch or loop statement

- The **continue statement** consists of the keyword **continue** followed by a semicolon
  - When executed, the continue statement ends the current loop body iteration of the nearest enclosing loop statement
  - Note that in a **for loop**, the continue statement transfers control to the update expression

- When loop statements are nested, remember that any break or continue statement applies to the **innermost**, containing loop statement

Tutorial Q1

Exercise 1a: Histogram of temperatures

Write a program that reads in temperatures (in Celsius) for five days, that is, from Monday to Friday and plots a histogram showing the temperatures. The name of your class should be `Temperatures`. Given below is a sample run of the program.

```
Please enter temperature for Monday: 25
Please enter temperature for Tuesday: 33
Please enter temperature for Wednesday: 26
Please enter temperature for Thursday: 28
Please enter temperature for Friday: 20

Histogram of Temperatures
-------------------------
Monday      | **************************
Tuesday     | *********************************
Wednesday   | **************************
Thursday    | ****************************
Friday      | ******************
```

Exercise 1b: Input and Output Redirection

You will be given a sample test input file `test0.txt` and the corresponding sample output file `test0-output.txt`. When you run your program by the following command in a terminal (or Windows command line):

```
java Temperatures < test0.txt > my-output.txt
```

your program should produce a file name `my-output.txt`which should be exactly the same as `test0-output.txt`. In this command, "`< test0.txt`" and "`> my-output.txt`" are called "input redirection" and "output redirection." They use the content in `test0.txt`as the command line input, and print the program output into `my-output.txt`.

Exercise 1c: Submission

In this subject, you will be submitting your projects via the engineering servers. To get familiar with the process of project submission, we will do a practice submission as part of this lab.

Your program should be contained within a single Java class. You must call this Java class `Temperatures.java`and store it in a directory under your home directory on the Engineering School server. Then, you can submit your work using the following command:

```
submit COMP90041 wk4 Temperatures.java
```

You should then verify your submission using the following command. This will store the verification information in the file "feedback.txt", which you can then view:

```
verify COMP90041 wk4 > feedback.txt
```

You should issue the above commands from within the same directory as where the file is stored (to get there you may need to use the cd "Change Directory" command). Note that you can submit as many times as you like to test your code.

How you edit, compile and run your Java program is up to you. You are free to use any editor or development environment. However, you need to ensure that your program compiles and runs correctly on the Engineering School servers, using build 1.8.0 of Oracle's (as Sun Microsystems has been acquired by Oracle in 2010) Java Compiler and Runtime Environment, i.e., javac and java programs.

Tutorial Q2

Exercise 2: Traffic Infringements

The traffic section of a Police Department wishes to automate the writing of warnings, fines etc. to motorists who exceed the 60km/hr speed limit and whether doing it under influence of liquor or not. Your task is to implement the following warning and fines in the program based on the corresponding conditions:

Condition	*Message(s)*
> 60 and <65	Warning
>60 and <65 and drunk	Warning + Take a shower
65 to <= 70	$5 fine for each km/hr over 60 km/hr
65 to <= 70 and drunk	$7 fine for each km/hr over 60 km/hr + Take a shower
> 70	$10 fine for each km/hr over 60 km/hr
> 70 and drunk	$15 fine for each km/hr over 60 km/hr Spend the day/night in cell until become sober

The program should ask the traffic officer to type in the km/hr speed of the offending driver. It should then ask whether driver is drunk or not. (The officer answers with a ‘y’ or ‘n’ and the appropriate message is then given.) The program should then display the appropriate message and where any fine is applicable, the program should compute and display the fine.

Sample Run 1

```
Please enter speed: 64
Is the driver drunk? ('Y' for drunk, 'N' otherwise): N

**************************************************
Warning

--------------------------------------------------
You have a fine of $0.0
**************************************************
```

Sample Run 2

```
Please enter speed: 64
Is the driver drunk? ('Y' for drunk, 'N' otherwise): Y

**************************************************
Warning + Take a shower

--------------------------------------------------
You have a fine of $0.0
**************************************************
```

Sample Run 3

```
Please enter speed: 85
Is the driver drunk? ('Y' for drunk, 'N' otherwise): Y

**************************************************
$15.0 fine for each km/hr over 60 km/hr

Spend the day/night in cell until become sober.

--------------------------------------------------
You have a fine of $375.0
**************************************************
```