CS1073 Assignment # 5 - Fall 2020

Submission Deadline: Friday, October 16th before 12:00 NOON (Atlantic Daylight Time Zone) in the Assignment 5 dropbox in Desire2Learn. (Read the submission instructions at the end of this document carefully).

The purpose of this assignment is:

- to introduce the `boolean` data type and decision statements

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor.

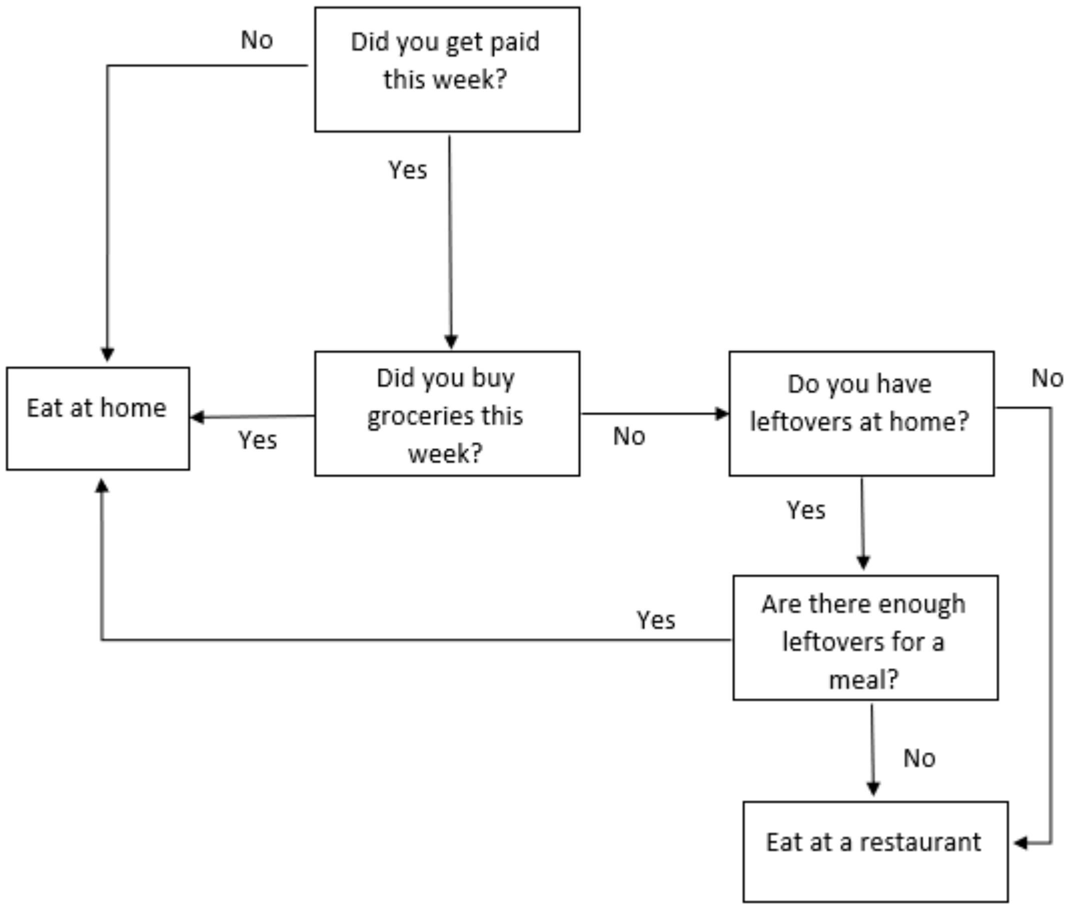
1. Making a Decision

Write a program that will help the user to help determine whether they should eat at a restaurant or at home.

Your program should be interactive and should be based on the flow chart below (see page 2). The boxes with questions are what you will ask the user. They can answer with either "yes" or "no" to each question. Eventually they will reach one of the two final boxes, where the program will be able to recommend that they should eat at a restaurant or eat at home.

HINT: Start by doing the first question only, assuming "no" means they should eat at home, and "yes" means they should eat at a restaurant. If you can get that working, how can you add another question? (and so on).

Continued on the next page…


Did you get paid this week?
No
Yes
Eat at home
Did you buy groceries this week?
Yes
No
Do you have leftovers at home?
No
Yes
Are there enough leftovers for a meal?
Yes
No
Eat at a restaurant

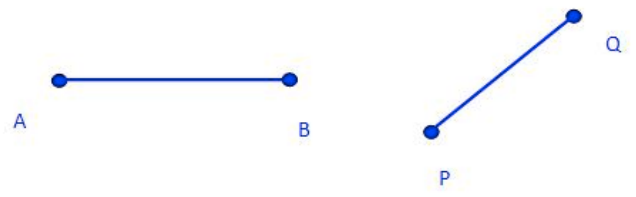
This program can be written completely in a main method. You should include a Javadoc comment block for your class (with the `@author` tag and your name & student number). Other non-Javadoc comments (using `/* … */` and/or `//`) could also be included within your main method to explain different sections of your code.

Once this first question is complete...

After you have tested your application and you're sure that it works properly, save sample output showing at least 3 runs of your program. For this question, take a snapshot of the terminal (or Command Prompt) window and save that as your output. That way, the markers will be able to see the input values as well.

2. Enhancing a LineSegment Class

In question 2 you will add methods to a `LineSegment` class to perform some basic geometry calculations.



Part A. In Desire2Learn you will find a `Part2.zip` archive containing two Java classes called `CartesianPoint.java` and `LineSegment.java`. Download and unzip the archive file. Study and compile both classes.

Part B. Add the following 2 methods to the `LineSegment` class:

- `public boolean containsPoint(CartesianPoint p)` – returns true if the line contains the specified point, false otherwise.
- `public boolean isVertical()` – returns true if the line segment is vertical, i.e. the line is straight up and down, false otherwise.

Consult math texts or online sources if you forget the definitions of a "vertical line segment" or how to easily determine if a line segment contains a point.

IMPORTANT NOTE regarding equality comparisons[1]: When working with floating-point numbers in Java (or any programming language), calculations don't work out exactly as we might expect. For instance, we may perform some calculations and end up with an answer like: 3.999999... instead of the expected answer of 4. This happens because of the limited precision used to store/represent floating–point numbers in a computer.

Therefore, rather than comparing two double values using:

```
if (num1 == num2) { ...
```

you should instead calculate the difference between the two numbers and then compare that to some really, really small value. If the difference is less than that really small value, then the two numbers are, essentially, equal.

For example, use:

```
if (Math.abs(num1-num2) < TOLERANCE) { ...
```

where `double TOLERANCE = 1E-14` (or some other really small number).

Aside: `Math.abs(num1-num2)` simply gives the absolute value of the difference between `num1` and `num2`.

[1] See also: Section 5.3 of the course textbook.

Part C. In a separate file, create a test driver that exercises the two new methods in your `LineSegment` class; name this class `TestLine`. In your test driver, create at least 2 `LineSegment` objects and 2 additional points (one that is on a line segment, one that is not on a line segment). One of the `LineSegment` objects should also be vertical. Call both methods on each of these objects and print out the results in a meaningful way, such as "The point p1 is not on the line segment line1." or "The line segment line1 is not vertical". Compile and run this program. Examine the output and return to Part B if you notice any problems. Once it is running correctly, please save sample output.

Part D. Add a third `@author` tag to the javadoc comment at the top of the `LineSegment` class and add your name & student number (since you have also now written parts of this class). In addition, add a javadoc comment for each of the two new methods that you added; be sure to include `@param` and/or `@return` tags where appropriate. You do not need to write javadoc comments for the other methods and instance variables; you only need to document the two methods that you added.

Add a javadoc comment to the top of your `TestLine` class; include the `@author` tag (with your name & student number).

You are not required to generate and hand in the html documentation.

Submission instructions are on the next page…

Your electronic assignment submission (submitted via Desire2Learn) will consist of two files:

i. a written report. This should begin with a title page; just as we described in Assignment #1, your title page should include: the course (CS 1073), your section (FR01A, FR02A, FR03A or FR04A), the assignment number, your full name, and your UNB student number. That should be followed by four sections, with each part clearly identified with a section heading. Include:

   a. the image(s) of the sample output you created by running the program 3 times as per Question 1.

   b. the source code for Question 1.

   c. an image of the sample output you created by running `TestLine` for Question 2.

   d. The source code for `LineSegment.java` and `TestLine.java` (do not include the source code for `CartesianPoint.java`).

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) The **SINGLE pdf file** containing your report will be submitted to the appropriate assignment drop box on Desire2Learn. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: Please name this report as follows: **YourName_As5_Report.pdf**

ii. an archive file (**.zip**) that contains your Java source code and output for this assignment. Make sure that your archive includes all .java files (in case the marker wishes to compile & run your code to test it). You should not include the report document or the .class files in your archive. This archive should be submitted as a **single file** to the appropriate drop box on Desire2Learn.

   Note: Please name this archive file as follows:
   YourName_As5_Archive.zip