CS1073 - Assignment #3 - Fall 2023

Submission Deadline: Friday, October 13th before 12:00 NOON (Atlantic) in the Assignment 3 submission folder 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
- illustrate the "has-a" relationship between classes
- review javadoc.

This assignment is to be done individually. What you hand in must be your own work. Incidents of plagiarism will be reported.

If you have questions about the assignment, you should first go to a scheduled help session. (Locations and times for all help sessions can be found on D2L). If you have attended a help session and the issue is unresolved, you may contact your course instructor. You are NOT to discuss this assignment with anyone else (including your classmates).

As always, begin by creating a new folder to hold your work for this assignment.

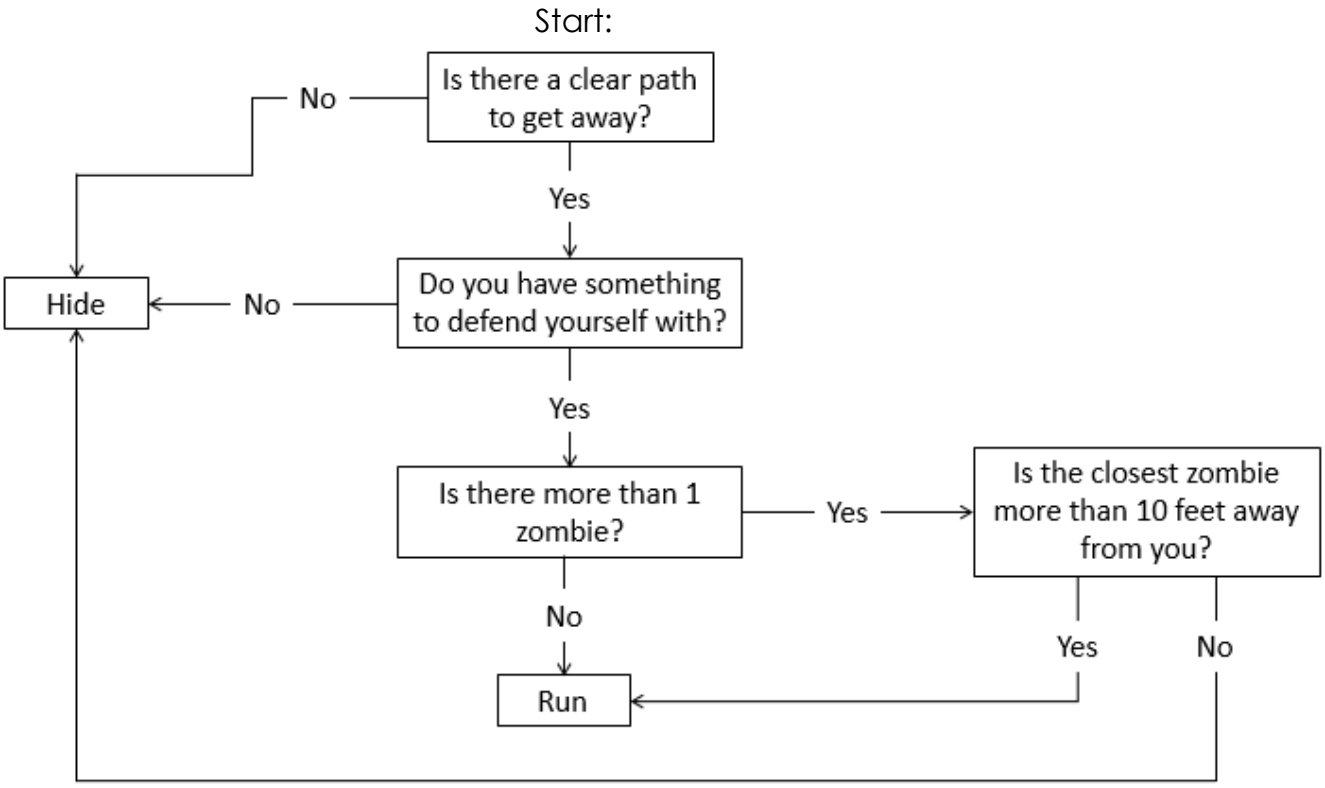
I. Programming Exercise: How does Ellie survive the zombies?

Write a program to help Ellie to determine whether she should hide from the zombies or run from them.

Your program should be interactive and should be based on the flow chart below. 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 you will be able to recommend that they should hide, or run.

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

Continued on the next page…



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). Other non-javadoc comments (using `/* … */` and/or `//`) could also be included within your main method to explain different sections of your code, but that is entirely optional.

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.

Note: Always adjust the size of your terminal window before taking a screenshot. Make the window taller if some of the sample output/input text is cut off. If there is a lot of extra whitespace beside your text, you can make the terminal window narrower. (That way, the text in the image will be larger and more legible when you copy it into your report.) It is important that the marker can easily read all of your sample output!

II. Magic The Gathering Card Appraisal Application

Magic The Gathering is a collectable card game. Each card has a particular set that it is a part of.

Part A:

Create a class named MagicSet that can be used to represent a set of Magic The Gathering (MTG) cards. Each set has its own name (e.g. Legends), the release year of the set (e.g. 2013), and if it is out of print or not (Note: a set is either out of print or it is not – there are only 2 options).

Provide a constructor method (which initializes all three instance variables), as well as a simple accessor method for each variable.

Provide a mutator method called setOutOfPrint which allows the variable to be updated when the set goes out of print, or when a set that had been out of print starts being printed again.

Compile your MagicSet class and check it over carefully. It is recommended that you write a short driver program to test this MagicSet class before proceeding. However, this test driver does not need to be submitted to Desire2Learn; it is just for your own testing/verification purposes.

Now, write javadoc comments for your MagicSet class. Include a comment for the class, for each instance variable and each method. Use @author, @param & @return tags where appropriate. Run the javadoc utility on your file and view the resulting MagicSet.html file in a browser to make sure that your Javadoc comments were inserted/formatted correctly. (Be sure to include author and private information when generating the documentation with javadoc.)

Once you have tested and documented your MagicSet class, you may move on to part B.

Part B:

In the same directory (folder), create a new class named MagicCard. This class will be used to represent a Magic The Gathering (MTG) playing card. For each MTG card, we will have the following instance variables:

- name – The name of the card (e.g. Upheaval)
- retailPrice – The price the card was originally sold for in stores (e.g. 2.50)
- type – The type of card (Note: there are 6 types of cards: planeswalker, creature, instant, sorcery, enchantment, artifact, land)
- set – The set of cards that this card is a part of. (Note: use the class that you created in part A.)

Provide a constructor method that will initialize all of the instance variables.

Provide the following accessor methods:

- getCardValue – This method will return the value of the card based on its retail price, type of card, and if its set is out of print. If the card is a land type then the value is half of the retail price. For all other types, if the card's set is not out of print, then its value is the same as its retail price. If the card's set is out of print, the value of the card is double the retail price.
- getCardDetails – This method will return a textual string that includes the card's name, type, the name of the set it is from, and the release date; the string should be in the following format:

```
Lightning Bolt (sorcery)
Legends 2013
```

Compile your MagicCard class and check it over carefully.

Then, write javadoc comments for your MagicCard class. Include a comment for the class, for each instance variable and each method. Use @author, @param & @return tags where appropriate. Run the javadoc utility on your file and view the resulting MagicCard.html file in a browser to make sure that your Javadoc comments were inserted/formatted correctly. (Be sure to include author and private information when generating the documentation with javadoc.)

Part C:

In a separate file in the same directory, create a test driver, (a class with a main method), named As3TestDriver. This test driver will use the code that you wrote for parts A and B (described above). In your test driver, please use the test data that is described below.

Begin by creating objects based on the following information:

- The MTG card, Ancestral Recall, is an instant from the Limited Edition Alpha set which was released in 1993. The set is out of print. The retail price of the card is $3.50.
- The MTG card, Island, is a land from the Fourth Edition set. The Fourth Edition set was released in 1995 and is not out of print. The retail price of the card is $1.75.
- The MTG card, Aeon Chronicler, is a creature from the Planar Chaos set. The Planar Chaos set was released in 2007 and is not out of print. The retail price of the card is $4.25.
- The MTG card, Necropotence, is an enchantment from the Wilds of Eldraine set. The Wilds of Eldraine set was released in 2023 and is not out of print. The retail price of the card is $5.95.
- The Planar Chaos set is now out of print.

Finally, add code to the bottom of your main method to retrieve and print out for each card:

- The card details.
- The value of the card.

Run your test driver and redirect the output to a file named As3Output.txt. Examine this file carefully to verify that your code works properly. If there are any problems, you may need to return to parts A and/or B to modify some of that code. If you do need to modify your code from parts A and/or B, check to see if any changes to the javadoc comments are also needed. Include a javadoc comment at the top of your As3TestDriver class. This comment should include a one-line description of the class and @author information.

Submission instructions are on the next 2 pages…

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

i. a written report. This should begin with a title page; your title page should include: the course (CS 1073), your section (FR01A, FR02A, FR03A, FR04A, FR05A or FR06A), the assignment number (Assignment #3 in this case), your full name, and your UNB student number. That should be followed by six sections, with each part clearly identified with a section heading. Include:

   a. the source code for question I

   b. an image of the sample output you created by running the program as per question 1

   c. the source code for MagicCard.java

   d. the source code for MagicSet.java

   e. the source code for A3TestDriver.java

   f. the sample output from running your application as per question II.

(Aside: Your source code should contain all of the javadoc comments mentioned above. However, you do not need to include the .html files in your report.)

This written report should be prepared using a word processor. (Options were outlined in Lab #1. If you choose to use the online version of MS Word, please see the note from Lab #1 about the issue when copying and pasting tabs into a document.)

Copy & paste the 6 items listed above into the report document. (These should appear in the document in the order that they are listed above). 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 and question II output to maintain proper indentation. (Examples of monospaced fonts were mentioned in Lab #1.)

Continued on the next page…

Once the report is complete and you've checked it all over, save the final document file for your own records. Then **save a second copy in PDF format for submission**. (Note: Be sure to open the second 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 lab submission folder in Desire2Learn. (It is important that you submit a PDF file and NOT the original Word or LibreOffice 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_As3_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 the .java files (in case the marker wishes to compile & run your code to test it), the image and output file (with the correct filename). You should not include the report document or the .class files in your archive. This archive should be submitted as a single .zip file to the appropriate submission folder on Desire2Learn.

Side note: While we encourage you to run the javadoc tool on your .java files (to check to make sure that you wrote the Javadoc comments correctly), you do not need to add to your As3 archive all of the files & folders that the javadoc utility creates.

This archive should be submitted as a **single .zip file** to the appropriate submission folder in Desire2Learn.

Note: Please name this archive file as follows: **YourName_As3_Archive.zip**

Reminder: Your submission in Desire2Learn should consist of TWO files (a .pdf and a .zip). Do NOT put your report inside your archive.

End of Assignment 3