Dept. of Computer Science & Software Eng., Concordia University
COMP 6481 --- Winter 2022

Programming and Problem Solving

Assignment 1 --- Due Sunday, February 20, 2022

Part I

Please read carefully: You must submit the answers to all the questions below. However, this part will not be marked. Nonetheless, failing to submit this part fully will result in you missing 50% of the total mark of the assignment.

Question 1

a) Given an array of integers of any size, $n \geq 1$, and a number m, develop an algorithm as a **pseudo code** (not a program!) that would move all numbers with value m to beginning of the array keeping order of remaining elements in the array same. You must perform this operation in place *i.e.,* without creating an additional array and keeping the number of operations as small as possible. For example, given an array [1, 3, 2, 7, 2, 4, 2], and a number 2, the algorithm will return [2, 2, 2, 1, 3, 7, 4]. Finally, your algorithm **must not** use any auxiliary/additional storage to perform what is needed.

b) What is the time complexity of your algorithm, in terms of Big-O?

c) What is the space complexity of your algorithm, in terms of Big-O?

Question 2

Given a string of random length and random contents of characters, that do not include special characters, write an algorithm, **using pseudo code** that will swap every (i+1)th character with the (i-1)th character, starting from the second character. For instance, given a string “assignment1” the algorithm should return the string: “signment1sa”.

a) What is the time complexity of your algorithm, in terms of Big-O?

b) What is the space complexity of your algorithm, in terms of Big-O?

Question 3

i) Develop a **well-documented pseudo code** that finds two farthest elements in the array with the difference of their digit sum equal to 1, and two consecutive elements with the largest difference in their digit sum. The code must display values and indices of these elements. For instance, given the following array [20, 52, 400, 3, 30, 70, 72, 47, 28, 38, 41, 53, 21] your code should find and display something like the following (notice that this is just an example. Your solution must not refer to this example.)

Two farthest elements with difference of their digit sum equal to 1 are 52 and 53 which have 9 elements between them. Two consecutive elements with the largest difference in their digit sum are 20 and 52. In case of multiple occurrences of the pairs with farthest distance or largest difference, your code must display the first found pair.

ii) Briefly justify the motive(s) behind your design.

iii) What is the time complexity of your solution? You must specify such complexity using the Big-O notation. Explain clearly how you obtained such complexity.

iv) What is the maximum size of stack growth of your algorithm? Explain clearly

Part II

Purpose: Practically, this part of the assignment can be considered as *Programming Assignment # **0***! The purpose of this assignment is to help you review some of the main topics covered in previous courses, including classes, loops, arrays, arrays of objects, static attributes, and static methods.

General Guidelines When Writing Programs:

- Include the following comments at the top of your source codes
  // ---
  // Assignment (include number)
  // © Your Name
  // Written by: (include your name and student id)
  // ---
- In a comment, give a general explanation of what your program does. As the programming questions get more complex, the explanations will get lengthier.
- Include comments in your program describing the main steps in your program.
- Display clear prompts for users when you are expecting the user to enter data from the keyboard.
- All output should be displayed with clear messages and in an easy-to-read format.
- End your program with a closing message so that the user knows that the program has terminated.

Part II A

For this part, you are required to design and implement the Vaccine class according to the following specifications:

✓ A Vaccine object has five attributes, a vaccine *brand* (Enum), vaccine *dose* (double), vaccine *expiry_date* (String), vaccine *id* (long), and a *price* tag (double).

- Upon the creation of a Vaccine object, the object must immediately be initialized with valid default values for all the attributes. (Hint: use constructors.).
- The design must allow enough flexibility so that value of any of these attributes can be modified later except the *id*. For example, one should be able to create a Vaccine object with a given price then change its price later. The design should also allow the user to obtain value of any of the attributes. (Hint: use accessors & mutators.)
- The design must also allow all information of an object to be displayed at once through the System.out.print() method. (Hint: use toString() method).
- It is required to know how many Vaccine objects have been created so far at any time. For that, you must have a method called findNumberOfCreatedVaccines(), in the Vaccine class. This method must return number of Vaccine objects prior to the time this method is called. The method would return 0 if no vaccines have been created by the time the method is called. (Hint: use Static – You can add other attributes to the class.).
- It is required to compare two Vaccine objects for equality. Two Vaccine objects are considered equal if they have the same *brand* and *dose*. (Hint: use equals() method).

Part II B

You are hired by a pharmacy to write a software application that helps their staff (users) in keeping track of the vaccines at the store.

Write a driver program that will contain the **main()** method and will perform following: (Note: You can have the main method in a separate driver file, or in the same file)

- Display a welcome message.
- Prompt the user for the maximum number of vaccines (maxVaccines) his/her store can contain. Create an empty array, called inventory, that will have the potential of keeping track of all the created Vaccine objects.
- Display a main menu (figure 1) with the following choices and keep prompting the user until they enter a number between 1 and 5 inclusive:

```
What do you want to do?
    1.  Enter new vaccines (password required)
    2.  Change information of a vaccine (password required)
    3.  Display all vaccines by a specific brand
    4.  Display all vaccines under a certain a price.
    5.  Quit
Please enter your choice >
```

Figure 1. Main menu

- When option 1 is entered:
  - Prompt the user for his/her password. (Make sure you have a constant variable containing the password "password" – do not use any other password as it will be easier for the marker to check your assignments). The user has a maximum of 3 attempts to enter the correct password. After the 3rd wrong attempt entry, the main menu in figure 1 is re-displayed. Additionally, after this is repeated 4 times (*i.e.*, after total 12 consecutive failed attempts), the program must display following message:

    "Program detected suspicious activities and will terminate immediately!", then the program must exit.
  - If correct password is entered, ask the user how many vaccines he/she wants to enter. Check to make sure that there is enough space in the store (array of Vaccines) to add that many vaccines. If so, add them; otherwise inform the user that he/she can only add the number of remaining places in the array. (How the vaccine information will be inputted/entered by the user, is up to you).
- When option 2 is entered:
  - Prompt the user for a password. (Make sure you have a constant containing the password "password" as a constant – do not use another password). Again, the user has 3 attempts to enter the correct password. However, after the 3rd wrong attempt, the main menu in figure 1 is simply re-displayed. (Notice the different behaviour in that case from the previous one above).
  - Once the correct password was entered, the user is asked which vaccine number he/she wishes to update. Vaccine number is the index in the array inventory. If there is no Vaccine object at specified index location, display a message asking the user if he/she wishes to re-enter another vaccine, or quit this operation and

go back to the main menu. If the entered index has a valid vaccine, display the current information of that vaccine in the following format:

```
Vaccine: # x (index of the vaccine in the inventory array)
ID: ID of the vaccine
Brand: Brand of the vaccine
Dose: Dosage amount of the vaccine
Expiry: Expiry date of the vaccine
Price: $ price
```

- Then ask the user which attribute they wish to change by displaying following menu.

```
What information would you like to change?
    1.  Brand
    2.  Dose
    3.  Expiry
    4.  Price
    5.  Quit
Enter your choice >
```

Figure 2. Update menu

- Once the user has entered a correct choice, make the changes to the attribute then display again all the attributes on the screen to show that the attribute has been changed. Keep prompting the user for additional changes until choice 5 is selected. Each time the user is prompted for a choice make sure that a number from 1 to 5 is entered, otherwise keep prompting until a valid number is entered. (Ensure that the user can change any of the choice 1 to 4 on figure 2).

➢ When option 3 (in the main menu shown in figure. 1) is entered, prompt the user to enter a brand name. You then need to display the information of all vaccines by that requested brand. (Hint: You may use a static method called findVaccinesBy, which accepts an Enum for a brand name name then performs the needed search).

➢ When option 4 (in the main menu shown in figure. 1) is entered, prompt the user to enter a value (representing a price). You then need to display all vaccines that have a value smaller than that entered value. (Hint: You may use a static method, for instance called findCheaperThan, which accepts a double value, for a price, then performs the needed search).

➢ When option 5 (in the main menu shown in figure. 1) is entered, display a closing message, and end the driver.

SUBMISSION INSTRUCTIONS

Submission format: All assignment-related submissions must be adequately archived in a ZIP file using your ID and last name as file name. The submission itself must also contain your name(s) and student ID. Use your "official" name only – no abbreviations or nick names; capitalize your "last" name. Inappropriate submissions will be heavily penalized.

IMPORTANT: For Part II of the assignment, a demo for about 5 to 10 minutes will take place with the marker. You **must** attend the demo and be able to explain their program to the marker. The schedule of the demos will be determined and announced by the markers, and students must reserve a time slot for the demo.

Now, please read very carefully:

- **If you fail to demo, a zero mark is assigned regardless of your submission.**
- **If you book a demo time, and do not show up, for whatever reason, you will be allowed to reschedule a second demo but a penalty of 50% will be applied.**
- **Failing to demo at the second appointment will result in zero marks and no more chances will be given under any conditions.**

EVALUATION CRITERIA

IMPORTANT: Part I must fully be submitted. Failure to submit that part will cost 50% of the total marks of the assignment!

Part II.A (Class Vaccine)	**4 pts**
Default & copy constructors	1 pt
Accessor/mutator method for static attribute	1 pt
equals, toString and static attributes/methods	2 pts
Part II.B (Driver & other static methods)	**6 pts**
Handling of password	1 pt
Handling of option 1	1 pt
Handling of option 2	1 pt
Handling of option 3	1 pt
Handling of option 4	1 pt
Handling of option 5	1 pt