CECS 220
Summer 2019
Assignment 3
June 20, 2019

Due Date: Sunday, June 30, 2019, 23:59
Total Points: 20

1. **(20 points)** Solve the following problem
   (a) When a new user logs in for the first time on a website, the user has to submit personal information such as user_id, password, name, email address, telephone number, and so forth. Typically, there are two fields for passwords, requiring the user to enter the password twice, to ensure that the user did not make a typo in the first password field.

   Write a class encapsulating the concept of processing a form, called *ProcessForm*, with the following elements: User_id, name, password, re-enter password, email address, and telephone number. The *ProcessForm* class has one instance variable, called *UserInfo*. It is an array of 6 elements (in this case) of Strings, containing the user's personal information elements. The *ProcessForm* class contains the following methods:

   - A constructor with one parameter, that is a one dimensional array of 6 words representing the user's information, used to initialize the *ProcessForm* class instance variable. The constructor must check that the passed array length matches the array length of the instance variable. If the two arrays do not have the same length, the instance variable array elements are initialized with empty strings, and warning message is displayed on the screen.
   - Accessor method, *getUserInfo*, that returns an array of Strings for the user's information.
   - A method, called *SetInfo*, that accepts two parameters to set or update an element of the *UserInfo* array, a string value and an index related to the position of the element in the array.
   - *toString* method that returns a string of the user's information nicely formatted, containing all the personal information in a printable form. However, the user's password elements should be hidden.
   - A method, called *CheckPasswords*, which checks if the two Strings representing the passwords are identical. If they are, it returns *true*; if not, it returns *false*.
   - A method, called *CheckEmailAdd*, which used to check if the String representing the email address actually "looks like" an email address. For simplicity, we assume that an email address contains one and only one "@" character, and contains one or more periods after the @ character. The method returns *true* if the email address element looks like a valid one; otherwise, it returns *false*.
   - A method, called *CheckPhone*, which is used to check if the String representing the phone number satisfies the format of a phone number in North America. The phone number format is specified as follows:
     Local number (7-digits subscriber): Nxx-xxxx, where N is for digits 2-9.
     Domestic (10-digits): NPA-Nxx-xxxx, where NPA refers to 3-digit area code.

     The method returns *true* if the phone number element is valid; otherwise, it returns *false*.

(b) Write a client class, called *Website_User*, that allows a user to enter his/her personal information for a website. The program should prompt the user to enter each data element. The entered information is used to create an object of the *ProcessForm* class. Accordingly, the client program should check the validity of user's password, email address, and phone number, and allow the user to reenter each invalid information till a valid one is accepted. Finally, the program should display the user's information on the screen.

Report and Program Submission Guidelines

It is expected that your report to be well written and organized. The report reflects your understanding of the assignment and its solution. So, you must assume that your marks assigned to the report part is related to how is the report is written and presented. All reports must be submitted in PDF format. Each assignment should contain the following:

1. **Report**
   1.1. Title page with your name, assignment number and the day you are actually submitting this report (Not the assignment due date).
   1.2. A brief description of your solution of each problem of the assignment, you can also explain your solution using a pseudocode. *Number your descriptions according to the problem numbers.*
   1.3. A comprehensive set of snapshots showing the inputs submitted, outputs obtained in the case of a successful output or a failure, including required output formatting, prompts, and messages.

2. **Submission Procedure:**
   2.1. Java source files that contain your solutions. They must be a "*.java" files. Source programs should contain meaningful comments and variable names.
   2.2. Please zip both the PDF document with the source codes and submit one zipped file. Please name your zipped file as "HWx_firstname_lastname.zzz". Where, "firstname" and "lastname" refer to your first and last names, "x" refers to the homework number (e.g., 1, 2, etc), "zzz" refers to the file name extension for the software used for archiving.
   2.3. Submissions after the due date are accepted with a penalty of 25% per day (weekend days are counted as one-day delay).

Grading Table

Problem	Item		Points
Problem 1	Report (Description of solution)		2
	Output snapshots (input/output, formatting)		1
	Java Source Code Implementation	*ProcessForm* Class	12
		Website_User Class	5
	Total		20