Seneca College

June 25, 2018

Applied Arts & Technology

SCHOOL OF COMPUTER STUDIES

JAC444   **Demo Due dates: July 06 and July 13, 2018**
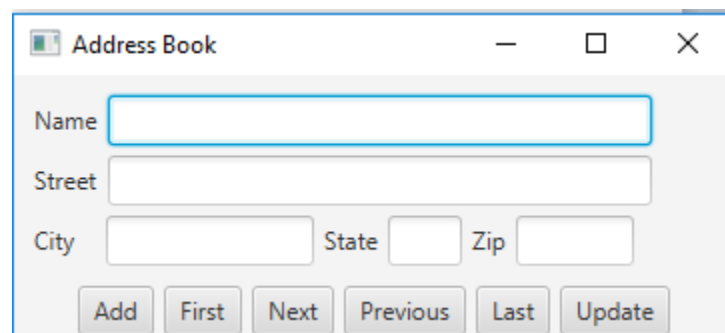**Final Code Submission Date: July 13, 2018**

Workshop 3

Notes:

i.   Each task should be presented during the lab, demo worth 50% of the workshop marks and code uploading worth the other 50%.

ii.   **At least one task** should be demoed in July 05th lab and the other task should be demoed on July 12th (Student can choose any task they want to give demo about first).

iii. Make sure you have all security and check measures in place, like exceptional handling and wrong data types etc.

iv.   Given output structure is just for student to have a glimpse what the output can look, student are free to make the output better in any way.

Other inputs can be given during demo, so make sure you test your program properly.

Task 1: (GUI practice)

(*Address book*) Write a GUI program that shows the same picture as below.

Now once you created the GUI of the address book, let's write a program which stores, retrieves, add, and updates addresses by using the GUI.

- Use a fixed-length string for storing each attribute in the address.
- Use random access file for reading and writing an address.
- Assume that the size of name, street, city, state, and zip is 32, 32, 20, 2, 5 bytes, respectively.

- Your program should generate a file name Address.dat, which stores all the information.
- Address.dat file will be used to retrieve the data for displaying the address.
- Upon updating an address your program should update the file Address.dat.
- Information on Random access file in java can be found on the link
  https://docs.oracle.com/javase/tutorial/essential/io/rafs.html
- Add at least 5 addresses.

Task 2 (a): (Lambdas Practice)

Suppose we have the following list of the top five male and female names of 2017:

List<String> topNames2017 = Arrays.asList(

"Amelia",

"Olivia",

"emily",

"Isla",

"Ava",

"oliver",

"Jack",

"Charlie",

"harry",

"Jacob"

);

With that list, you are suppose to write a code to print the items in the list in sorted order, and with the first letter in each name upper-cased. The name "harry" should be printed as "Harry" and should be printed after "Emily" and before "Isla". Use Lambda expressions where ever possible.

Task 2 (B):

Change the code for Task 1 (A) so that it uses method references. Remember that the method references looks like Class::MethodName.

Task 3(C):

Now do Task 1 (A) and (B) using a stream and chain of stream operations.