CSCI 1300
Exam #2

Solutions should be submitted in git with the filename `Exam2/exam2.py`. A preliminary version of this file, complete with some test code, has been provided. Your final output should appear similar to that shown at the end of the exam.

Also put the names of your team members in the `Names` file, and **for each team member, indicate which was their primary task**.

1. **Task 1: Read the user data in from a file**

   You are given a file, `sys_users.csv`, that defines a set of users of your system. Each person is listed on one line of the file, the set of entries is organized as specified on the first line of the file (repeated here for your convenience):

   ```
   FirstName,MiddleInit,LastName,Password,SecurityQues,SecurityAns,Age
   ```

   Create a function with the following signature:

   ```
   def readFile (filename):
   ```

   That will open the file, read in the lines, and then create a new SysUser object for each person. The SysUser object (as described below), contains all the info for one person.

   The `readFile()` function should ensure that any exceptions generated by an invalid file name or failure to open a file are handled gracefully.

2. **Task 2: Create the SysUser class**

   Complete the `SysUser` class, which stores all the user info for one person. The constructor for this class receives a single list one person's info in it, but **internally the class should store each list element in a separate instance variable** (e.g. `_password`, `_secAns`, etc.).

   The class should provide an accessor methods for the full person's name (first name, middle initial, and last name all together).

   The class should also provide an accessor method for the person's username (which is all lowercase), defined as:
   - the first letter of their first name, followed by
   - the first letter of their middle name/initial, followed by
   - the first 6 letters of their last name

The class should also provide accessor methods for the person's age, and the integer number of their security question.

The class should include two boolean accessors (that return True or False) that check whether the entered value for the password or the security question match the saved password and security question answer.

Finally, the class should contain two mutator methods for changing the security question, and for changing the answer to the security question.

3. Task 3: Print out user info in tabulated form

The final step is to print out some info about the users in the system in tabular form. The three columns include:

- The person's full name (left justified)
- The person's username (left justified, lining up directly under the 'U' in 'Username')
- The person's age (right justified, with the one's digit lining up directly under the 'e' in 'Age')

For more complete details, see the example output, below:

Here's what your final output should look like:

```
username:  jtrobins
password:  mightyMouse
What was the make of your first car?  Honda
*** Welcome, Josh T. Robinson

username:  tpbradshaw
*** Username invalid
password:  endZone
What's the name of your favorite sports team?  Steelers
*** Login failed.  Please try again.

username:  jagarner
password:  uglyPeople
*** Password invalid
What's your school mascot?  Big Red
*** Login failed.  Please try again.

username:  dsbrown
password:  bubbleGum
In what city was your mother born?  Atalnta
*** Security answer invalid
*** Login failed.  Please try again.

   Name                  Username          Age
----------               --------          ---
Josh T. Robinson         jtrobins           42
Dave S. Brown            dsbrown             7
Sarah D. Clark           sdclark           119
Kate M. Larson           kmlarson           28
Jennifer A. Garner       jagarner           47
Bill J. Madison          bjmadiso           35
John Q. Adams            jqadams            61
Lily A. Lane             lalane             54
Terry P. Bradshaw        tpbradsh           70
```