# ITEC 120: Principles of Computer Science I

## Homework 10 - BookFace

**>>>>>>>> Due Date: Wed., April. 25th at 10:00PM via Desire2Learn <<<<<<<**

In doing this homework, remember to abide by the RU Honor Code.

---

**Problem 1**
50 points

With all of the privacy issues that social networks can cause, you need to create a social networking system solely for universities. Each university can have multiple students, and each student has a name, an array of messages (Strings) called a wall, and a list of friends (other students at the same university). Students can also poke each other on the system. Poking involves leaving a message on the target's wall telling them a certain person has poked them. All interaction between students can only happen with students that attend the same university.

**Input**
Input will consist of one of following commands typed into the command prompt / terminal. You do not need to worry about incorrect input, the input given to your program is guaranteed to be correct.

*addSchool* – Prompts the user for a university name, and creates a university.
*addStudent* – Prompts the user for the student's name and university, creates a student object and adds them to the appropriate university.
*writeWall* – Prompts the user for a university, student's name, and message that will be added to the appropriate student's wall.
*readWall* – Prompts the user for a university, student's name, and prints out the messages on the appropriate student's wall.
*friend* – Prompts the user for a university, student's name, and the friend's name they want to add.
*findNewFriends* – Prompts the user for a university, student's name, and finds prints out the friends that specified student's friends have that the specified student does not have.
*poke* – Prompts the user for a university, student's name, and the student to poke. A message stating that the user has been poked by the student's name is left on the target's wall.
*quit* – Stops the program.

***Sample input file***

```
addSchool
RU
addSchool
VT
addStudent
RU
John
writeWall
RU
John
Hello John at RU
writeWall
RU
John
Hello again John at RU
readWall
RU
John
addStudent
VT
John
writeWall
VT
John
Hello John at VT
readWall
VT
John
addStudent
VT
Jane
addStudent
VT
Brad
friend
VT
John
Jane
friend
VT
Jane
Brad
findNewFriends
VT
John
poke
```

```
VT
John
Brad
readWall
VT
Brad
quit
```

***WARNING***

Your program must work with file redirection. You can test this by typing java yourProgramName < inputFile.txt (if you only use one Scanner for your entire program, this will not be a problem). Due to the amount of input for this project, this is a requirement for testing your program. If your program will not run in this manner, then you will receive at most a 25 on this assignment. Make sure you copy the sample input in this assignment, paste it into a text editor, save it as a plain text file in the same directory where your program is stored, and test it running your program with it.

## Output

Your program must produce output that tells the user what is happening in the program and report the results of the different commands entered into the system.

### *Sample output file*

The output (other than prompts) from the previous file is:

```
Messages from John's wall
Message  0:  Hello John at RU
Message  1:  Hello again, John at RU
Messages from John's wall
Message  0:  Hello John at VT
Your friend Jane has a friend named Brad that you do not have.
Messages from Brad's wall
Message  0:  You got poked by John
```

### Implementation suggestions

For a program of this size (about 200 lines of java code for my solution, not counting any documentation), it is essential that you practice incremental implementation. That is, don't attempt to write the entire program at once, even though you may have a complete design. Here is a suggested order of implementation:

- Create the capability to add and print messages from a Wall.
- Create the capability to store a wall, other students, and a name in a Student class.
- Create the capability to store multiple Students in a University class.
- Create the capability to poke Students in a university.
- Create the capability to friend each other, and to find new friends.
- Create the capability to read commands and execute them.

Do not wait until two or three days before the assignment is due to start it. You will probably need to allocate between five and fifteen hours in order to complete this homework assignment. You will probably not be able to write, test, and document it at one time.

**Class requirements for this program**
You must use at least four classes (including the driver class) in your solution.
You must make good use of classes in your implementation. There are many opportunities to do so in this project. For instance, you might use a class to encapsulate the information for a student, a class to model a university, and a class to manage the walls.

**Constraints**
Your main method should read in commands until the user types quit.

The reference solution for this project is 206 lines of code without comments. Feel free to use less or more code in your program.

**Submission requirements**
You must submit the .java file containing your program to Desire2Learn under the Homework #11 assignment. If your submitted file does not compile, it will receive a 0. You can demo the homework the next school day after it is due, or it will be graded automatically. If you choose not to demo your homework you cannot contest the grade you receive.

**Grading Rubric**
20 Points - Are classes designed / implemented properly?
15 Points - Do the commands work as expected?
10 Points - Is the program commented using inline and javadoc comments?
 5 Points - Was a reasonable test case included in your submission?