Assignment prefix: Lab01
Points: 100
Individual Assignment
Due Thursday, 01/20, at 11:59pm.

TASK:

You want to develop a Java program that will allow you to keep track of contacts in your address book. In reviewing your address book, you notice that your contacts fall into two categories: Friend contacts and Business contacts. The following table shows the information that you keep in your address book for each type of contact.

Field	Friend	Business
Name	Yes	Yes
Title	No	Yes
Business Name	No	Yes
Address	Yes	Yes
Phone	Yes	Yes
Birthday (as text MM/DD/YYYY)	Yes	No
Favorite Movie	Yes	No

Create a NetBeans project named Lab01

In this project create three classes named Contact, Friend and Business such that:

- The Contact class contains all of the fields common to both types of entries in your address book.
- The Friend class is a subclass of Contact and contains only those fields that are specific to the Friend entries in your address book.
- The Business class is a subclass of Contact and contains only those fields that are specific to the Business entries in your address book.
- Each of these classes contains all of the "normally expected" methods.
    - At least one constructor
        - Make this a overload constructor that includes all of the necessary information to create a instance.
    - A getter and setter (accessor and mutator) method for each instance/class variable
    - A toString method
    - An equals method
- Create your classes so that you can keep track of the
    - Number of Friend contacts static int FriendCount++.
    - Number of Business contacts
    - Total number of contacts

Create a fourth class name Client that will be used to test your other classes. In the Client class:

- This class must include the main method
- Create an array named addressBook of Contact
- Add a minimum of three Business contacts to the array
- Add a minimum of three Friend contacts to the array
- Data for each of the contacts must be entered from the keyboard.
- Test all of the methods in Contact, Friend and Business that are not explicitly tested in their respective class definitions.

Provide adequate documentation for your code where adequate documentation is defined as follows:

- Each instance or class variable should have a semantically rich name, i.e. the name should tell the reader what the variable represents.
- Each method should include a Java docs header.
- Each class should include a Java docs header comment block that includes the following:
    - Your name using the @author tag
    - The date using the @version tag
    - A brief description of the class
- The use of semantically rich identifiers can reduce the amount of documentation that needs to be written.
- Be sure to remove any unnecessary comments or code, e.g. the comment templates provided by NetBeans.

HOW TO TURN IN YOUR ASSIGNMENT:

- You will submit your solution as a single submission on Blackboard.
- Create a complete zip archive of your NetBeans project. This zip archive must be created using the NetBeans File->Export Project->To Zip command.
- **Note that a RAR file is not a ZIP file.**
- Create a Microsoft Word document (must be a .docx file) that contains the source code for each of your classes including the client class and the output of an example run of your client.
- Submit both your project zip archive and Word document as two separate files in a single Blackboard submission.