**Assignment 2:**

**Due: January 12th 2018**

**(Group size : 3)**

This Assignment was designed to take you through the process of creating basic classes and integrating them with a simple javafx generated interface.

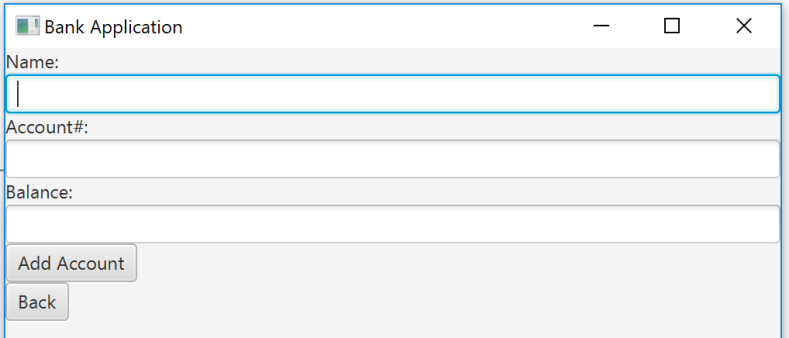
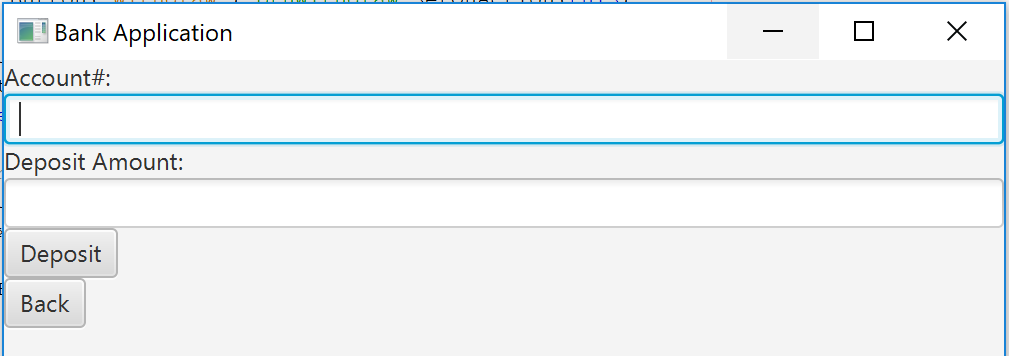
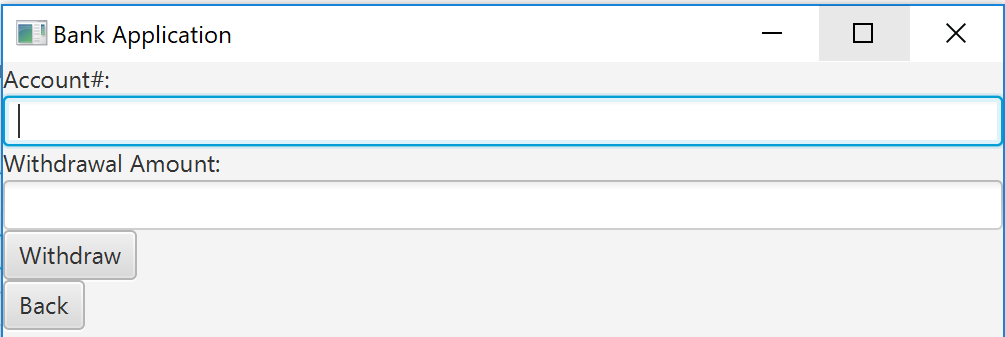
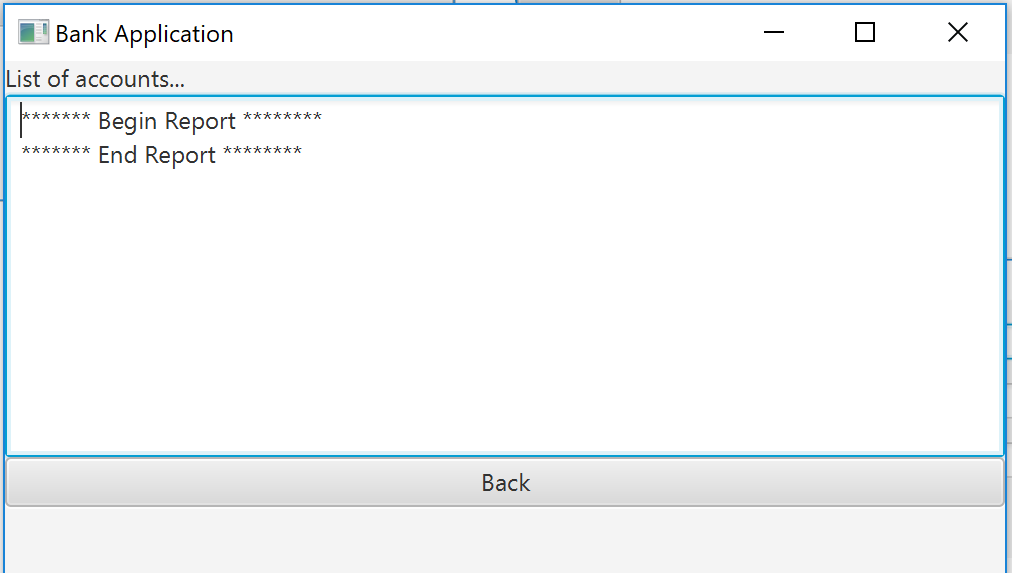
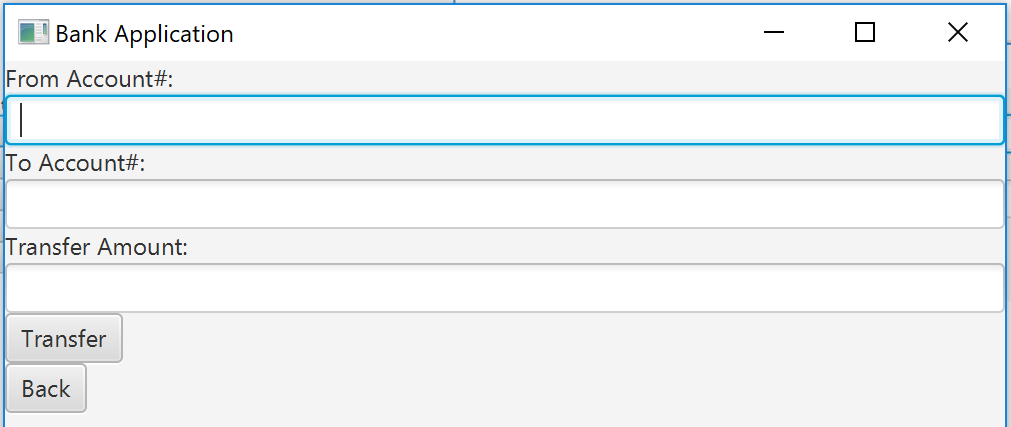
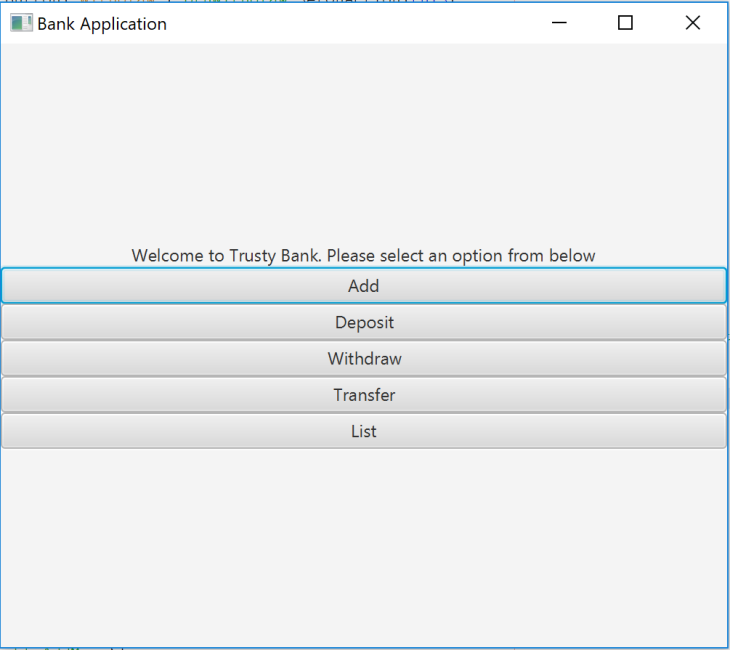
Your group must create a program that simulates a simple banking interface allowing the addition of new accounts, deposits , withdrawals and transfers between accounts in the bank. It also allows the viewing of all accounts in the system.

Overview:

A class Bank must be created that manages all of the Accounts in the system. A single instance of the Bank class must be created for this application. The Bank object may be declared in the main javafx application class and instantiated in the init( ) method of the javafx application.

Note: Starter code is provided for the Application. You must expand on this code GUI, create the Bank and Account classes and all three classes appropriately.

**The following page gives an overview of the different screens that are to be created.**



**The Account class:**

|  |
| --- |
| ACCOUNT |
| accountNum (long)  balance ( a double)  owner (a string) |
| Constructor (takes the accountNumber, balance and owner information)  withdraw(amount)  deposit(amount)  transfer(object,amount)  toString() |

**State information:**

accountNum - the account number. For example 192822227364

balance – the balance on the account. For example 34888.34

owner – a String representing the owner’s full name. For example “Andrew Rudder”

**Behaviour information:**

Constructor (takes the accountNumber, balance and owner information)

Boolean withdraw (amount) - reduces the balance by amount once there are sufficient funds

**If there are sufficient funds perform the transaction and return Boolean “true”**

void deposit(amount) increases the balance by amount.

Boolean transfer(account object, amount) transfers money from one account to another.

**You must check that there is sufficient money in the sending account**

**If there are sufficient funds perform the transactions and return Boolean “true”**

toString() returns a string containing the following account information:

**The account number,The balance on the account ,The name of the owner**

getAccountNumber() returns the account number.

getBalance() returns the balance on the account

getOwner() returns name of the owner

**The Bank Class:**

|  |
| --- |
| accountList -an array of type Account of size ten(1000). |
| numAccounts - an integer indicating the number of accounts in the array |
| maxAccount – an integer representing the maximum number of accounts allowed (set to 1000 in this case) |

|  |  |
| --- | --- |
| **Function Name** | **Signature** |
| addAccount | boolean addAccount(long accNum, double bal, String own) |
| printAccounts | String printAccounts() |
| findAccount | int findAccount(long accNum) |
| depositAccount | void depositAccount(long accNum, double amount) |
| withdrawAccount | boolean withdrawAccount(long accNum, double amount) |
| transfer | Boolean transfer(long accNumFrom,long accNumTo, double amount) |

addAccount – creates and adds an account to the array (accountList)

printAccounts –returns a string with all the account information for each account in the array.

findAccount – returns the location(index) of the account where the accountNumber equals accNum.

It returns -1 if the object does not exist .

depositAccount – finds the account with the matching account number in accountList and if it exists, adds the amount to the balance.

withdrawAccount – finds the account with the matching account number in accountList and if it exists, withdraws the amount from the balance if possible.

transfer – finds both of the accounts with the matching account numbers in accountList and if they exist, attempts to perform the transfer.

**Submission requirements:**

1. **The names and student numbers of all group members must be at the top of each class**
2. **You are required to compress (zip) and submit all the Java classes you created(Note: the files with the “.java” extension.**

**Marking Scheme:**

|  |  |
| --- | --- |
| **Attribute** | **Marks Allotment** |
| Account | 5 |
| JavaFx interface completion | 15 |
| Bank | 15 |
| Integration of all parts of the system | 15 |
| Documentation | 5 |
| Coding standards | 5 |
| TOTAL | 60 |

Final mark will be scaled as a percentage.