**Assignment#2, CS-2365, OOP**

**Consider the following Assignment Question. It consists of 18 parts which are divided among groups of three students. For your portion of parts, coordinate with your group members to create one single solution. You have to retrieve your project’s single java file (developed in NetBeans 8.1 IDE) and put the names of all your group mates with r#s and group#s and your assignment question parts in the comment section of your java file. Then compress your java file bearing your group information with your project folder using winrar. Upload your compressed file on the black board. Hand-in: 04/04/2019, Hand-out: 03/07/2019. Note that in the lecture session instructor has discussed the creation of BankAccount class. However, instructor has not yet taught the topic of GUI components creation. Therefore, to help students, instructor would upload an assignment handout on the blackboard. Otherwise, instructor would teach you these topics in the class after spring break. If anyone faces any problems in understand the details of your tasks, kindly email your problem to the instructor.**

**Goal Of Assignemnt: To familiarize with inheritance and fundamental GUI components like PushButtons, TextBoxes and Radio Buttons**

**Total Marks: 10.0**

**Question. Create a BankAccount class (fields private: String strName (TTU1, TTU2,..), int id(100, 101,…), char accountType i.e ‘C’ for current &‘S’ for saving accounts, double balance (hard-coded values). Provide get & set methods and parameterize & non-parameterize constructors. Also provide debit(double amount) & credit(double amount) methods and displayBalance( ) and displayAll( ) method.**

**Create a SavingAccount class inheriting from Account class. SavingAccount class (private: double rate ; Methods: public addInterest(…), deductLoan(…), parameterize constructor) *Note for simplicity, assume that SavingAccount class is managing both Saving i.e. accountType =’s’ and Current i.e. accountType = ‘c’ accounHolders.***

**The first line of constructor should invoke the base class constructor using super and then provide code to initialize the local instance variables.. You can hard-code other values also.**

**Now create a class UseSavingBankAccount class which incorporates GUI components. Create 50 array of objects for SavingAccount class using parameterize constructors.. Use a ‘for’ loop to create array of objects. Now provide radio button to perform following operations .**

**G1 (Guatum, Utkrist Dylan) G9 (Precious, Bharat, Sabin), G16(Thomas, Faus, Rahil)**

**Provide three radio Buttons and a “submit” push button for the following two operations**

**(1)Find out the account id(s) & name(s) with minimum and maximum balance amount and display it using JOptionPane. Provide methods findMinBalance(SavingAccount[]), findMaxBalance(SavingAccount[]);**

**(2)Credit 1000 to lowest Balance account & debit 1000 to maximum Balance account if Balance is greater than 1000.**

**(3) Display Balance of all AccountHolders using JOptionPane**

**G2 (Roshan, Michael, Liam) G10(Katlyn, Nicholas, Eduardo) G17(Isaak, William, Nathanael)**

**Provide two radio Buttons and a “submit” push Button for the following two operations**

**(3) Display list of all BankAccounts with all fields using JOptionPane sorted by name. Provide a method sortByName(SavingAccount[]).**

**(4) Search by name and display all the information of account holder using JOptionPane if found. Note initially the “by field and its associated TextBox for inputting name” is disabled. But if we select this radio button then the “by field and its associated TextBox for inputting name” is enabled.). Provide a method searchByName(SavingAccount[], String)**

**G3 (Pranjal, Joey, Abbey) G11(Mushfique, Eric, Edmond),G18(Haroon, Ivan Feebi)**

**Provide three radio Buttons and a submit Push Button for the following three operations**

**(5) Search the balance of an account holder by Name and display the result by JOptionPane. Note initially the “by field and its associated text box to input Name” is disabled. But if we select this radio button then the “by field and its associated text box” is enabled. Provide a method searchBalanceByName(SavingAccount[], String)**

**(6) Add interest to the account holder’s balance by Name. addInterest method should call credit(…).Note initially the “by field and its associated text box for inputting Name” is disabled. But if we select this radio button then the “by field and its associated text box” is enabled. Provide a method:addInterestByName(SavingAccount[ ], String)**

**(7) Display all information about account holders by accountType using JOptionPane**

**(Note initially the “by field and its associated text box for inputting accountType” is disabled. But if we select this radio button then the “by field and its associated text box” is enabled.). Provide a method: displayAllByAccountType(SavingAccount[ ], char);**

**G4 (Collin, Grant, Andrew) G12(Marie, Leslye, Trisha), G19(Anil, Santosh, Blaine) Provide three radio Buttons and a “submit” Push Button for the following three operations**

**(8) Search the balance of an account holder by id and display by JOptionPane. Provide a method searchBalanceById(SavingAccount[], int)**

**(9) Add an interest to the account holder by id,. addInterestById(SavingAccount[], int, double…) method should call credit(…).**

**(10) Display all information about account holders by accountType using JOptionPane. Provide a method displayAllByAccountType (SavingAccount[ ], char)**

**(Note initially the “by fields” is disabled for all above cases. But if we select the radio button then the “by field and its associated TextBox” is enabled.)**

**G5 (Safwan, Alex, Tyler), G13(Anish, Demetrius, Domingo) G20(Deepen, Tecpal, Rick)**

**Provide three radio Buttons and a submit Push Button for the following three operations**

**(11) Display all AccountHolders’ information by accountType sorted by balance in ascending order using JOptionPane. Provide Method: displayAllByAccountType(SavingAccount[], char); sortByBalance(SavingAccount[])**

**(12) Deduct loan amount from all account holders’ balance,. based upon accountType.. Provide a Method deductLoanByAccountType(SavingAccount[], char) should call debit(…).**

**G6 (Ronak, Saransh, Mohak) G14(Colyn, Chen, Jan) G22((Ihua, Bijay, Steven) Provide three radio Buttons and a “submit” Push Button for the following three operations**

**(13) Delete an account by id. Provide a method deleteAnAccountById(SavingAccount[], int)**

**(14) Delete an account by Name. Provide a method deleteAnAccountByName(SavingAccount[ ], String)**

**(15) Display all account holder’s information Sorted by name; Provide method sortByName(SavingAccount[ ]) and then call displayAll()**

**G7 (Jeffrey, Chigozie, Samuel) G8 (Aaja, Sydney, Trevor),G15(Sohan, Benjamin, Chang)**

**Provide three radio Buttons and a “submit” Push Button for the following three operation**

**(16) Display all saving account holder’s information sorted by balance in descending order. Provide sortByBalanceDes(SavingAccount[ ], char accountType) and then call displayAll() method**

**(17) Alter the names of all Saving AccountHolders by appending the String for example “Lubbock”. Provide the method alterNames(SavingAccount[], char accountType, String);**

**(18) convert all current account into saving account and addInterest to all account holders. Provide method convertCurrentToSaving(SavingAccount[], char, double)**