**Please note, if your codes are not compiling or cannot be executed, they will be graded as 0. You can decide any unmentioned details.**

**You will implement a simple ATM login simulation application.**

Write a GUI based Java application ATMLogin (at least 25 marks deduction for the console based alternation) follow the instructions below. You can decide the details if not mentioned or defined.

1. (**10 marks**) First, please create database Bank and build Account table, you can use the following statements (reference only, you don’t have to use it):

Statement stmt = conn.createStatement();

// Create the table.

stmt.execute("CREATE TABLE Account(" +

"CardNum Char(10) NOT NULL, " +

"PassWord Char(25) NOT NULL, " +

"UserName Char(25) NOT NULL, " +

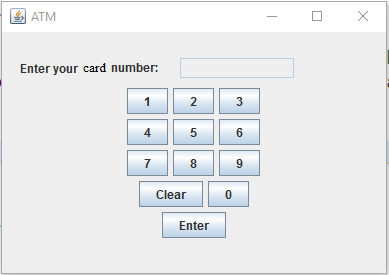
"ChequeBanlance Numeric(15,2) NOT NULL, " +

"SavingBanlance Numeric(15,2) " +

")");

Then insert data based on the SQL statement given in the **sql.txt file**.

2. (**30 marks**)Design and display key pad (using GUI) to allow user input the card number and password, a reference design for card number input is shown as



**Please not you need to figure out the GUI component type and layout.**

**Users are MUST be able to use the key pad displayed to input the card number.**

Once user click the clear button, all content input will be cleaned.

Once user finish card number input by clicking the “Enter” key, we need to check card number validity, which must be 10 digits and contained in the Account table.

**Hint: you can use the following query to verify if or not the card number is in the Account table:**

ResultSet resultSet = stmt.executeQuery(

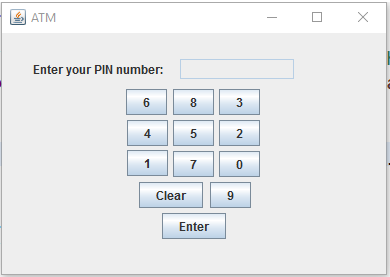
"SELECT UserName " +

"FROM Account " +

"WHERE CardNum = '" + cardNum + "'");

If the card number user input is not valid or not contained in the Account table please display a Message dialog (“Card number is invalid, please input again”) to ask user input again.

If the card number is valid, please randomly shuffle the number buttons of the key pad to enhance security:



**Hint: you don’t really need to re-design the GUI, you just need to randomly assign the numbers to those buttons.**

**Hint: Users are MUST be able to use the key pad displayed to input the card number.**

After user click the Enter button, use the card number and password input to login the account. The details will be discussed in the following part.

3.**(5 marks)** Define **interface AccountManagement** has the following method:

GetChequeBalance(): get the balance of the chequing account.

GetSavingBalance(): get the balance of the saving account.

Login(): log in the account using card number and password given.

4.**(10 marks)** Define **Bankcustomer** account which implements the **AccountManagement** interface. It has the following attributes

Name: customer’s name ;

CardNum: card number;

PassWord: pass word;

ChequeBalance: chequing account balance;

SavingBalance: saving account balance;

LoginStatus: whether or not user has been successfully logged in.

It will implements those methods:

GetChequeBalance(): get the balance of the chequing account using the SQL statement as:

ResultSet resultSet = stmt.executeQuery(

"SELECT ChequeBanlance" +

"FROM Account " +

"WHERE CardNum = '" + cardNum + "'");

GetSavingBalance(): get the balance of the saving account using similar SQL statement above.

Login(): log in the account using card number and password input, where you need to use the card number to query the password from Account table first and then compare the password with the one input. However, to improve the security level, the password in data base was encrypted, therefore, you need to decrypt the queried password first. We use a straight forward encryption approach:

The password was post-fixed by “999” at the end of the password string in database.

5. (**15 marks**) Define **Login** class which contains the main method. In the main method, display the GUI; create **Bankcustomer** ArrayList. Use the card number input to login and obtain the customer name create Bankcustomer object and add to the **Bankcustomer** ArrayList.

We will allow 3 times attempt for login, if the password input is failed 3 times, the corresponding card number will be locked and will not allow user to login and display Message dialog (“Card is locked!”).

**Hint: you can use ArrayList<String> to store the locked account number.**

Save the customer’s name, card number, chequing account balance and saving account balance to Log.txt file. Please refer to the test data in sql.txt to test your application.