**16TIN2054 - Teknik Pemrogaman (Praktek)**

**Tugas Minggu 8**



**Dikerjakan oleh :**

Muhammad Fikri Hidayatulloh - 201524017

1A – D4 Jurusan Teknik Komputer dan Informatika

Tugas ini dikumpulkan untuk memenuhi sebagian persyaratan kelulusan mata kuliah Teknik Pemrograman (Praktek)

**Program Studi D4 Teknik Informatika**

**Jurusan Teknik Komputer dan Informatika**

**Politeknik Negeri Bandung**

**2020-2021**

**BankDatabase.java**

**package** Pertemuan8;

//BankDatabase.java

//Represents the bank account information database

**public** **class** BankDatabase {

**private** Account[] accounts; // array of Accounts

// no-argument BankDatabase constructor initializes accounts

**public** BankDatabase() {

accounts = **new** Account[2]; // just 2 accounts for testing

accounts[0] = **new** Account(12345, 54321, 1000.0, 1200.0);

accounts[1] = **new** Account(98765, 56789, 200.0, 200.0);

}

// retrieve Account object containing specified account number

**private** Account getAccount(**int** accountNumber) {

// loop through accounts searching for matching account number

**for** (Account currentAccount : accounts) {

// return current account if match found

**if** (currentAccount.getAccountNumber() == accountNumber) {

**return** currentAccount;

}

}

**return** **null**; // if no matching account was found, return null

}

// determine whether user-specified account number and PIN match

// those of an account in the database

**public** **boolean** authenticateUser(**int** userAccountNumber, **int** userPIN) {

// attempt to retrieve the account with the account number

Account userAccount = getAccount(userAccountNumber);

// if account exists, return result of Account method validatePIN

**if** (userAccount != **null**) {

**return** userAccount.validatePIN(userPIN);

}

**else** {

**return** **false**; // account number not found, so return false

}

}

// return available balance of Account with specified account number

**public** **double** getAvailableBalance(**int** userAccountNumber) {

**return** getAccount(userAccountNumber).getAvailableBalance();

}

// return total balance of Account with specified account number

**public** **double** getTotalBalance(**int** userAccountNumber) {

**return** getAccount(userAccountNumber).getTotalBalance();

}

// credit an amount to Account with specified account number

**public** **void** credit(**int** userAccountNumber, **double** amount) {

getAccount(userAccountNumber).credit(amount);

}

// debit an amount from Account with specified account number

**public** **void** debit(**int** userAccountNumber, **double** amount) {

getAccount(userAccountNumber).debit(amount);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* (C) Copyright 1992-2018 by Deitel & Associates, Inc. and \*

\* Pearson Education, Inc. All Rights Reserved. \*

\* \*

\* DISCLAIMER: The authors and publisher of this book have used their \*

\* best efforts in preparing the book. These efforts include the \*

\* development, research, and testing of the theories and programs \*

\* to determine their effectiveness. The authors and publisher make \*

\* no warranty of any kind, expressed or implied, with regard to these \*

\* programs or to the documentation contained in these books. The authors \*

\* and publisher shall not be liable in any event for incidental or \*

\* consequential damages in connection with, or arising out of, the \*

\* furnishing, performance, or use of these programs. \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**ATM.java**

**package** Pertemuan8;

**public** **class** ATM {

**private** **boolean** userAuthenticated; // whether user is authenticated

**private** **int** currentAccountNumber; // current user's account number

**private** Screen screen; // ATM's screen

**private** Keypad keypad; // ATM's keypad

**private** CashDispenser cashDispenser; // ATM's cash dispenser

**private** DepositSlot depositSlot; // ATM's deposit slot

**private** BankDatabase bankDatabase; // account information database

// constants corresponding to main menu options

**private** **static** **final** **int** ***BALANCE\_INQUIRY*** = 1;

**private** **static** **final** **int** ***WITHDRAWAL*** = 2;

**private** **static** **final** **int** ***DEPOSIT*** = 3;

**private** **static** **final** **int** ***EXIT*** = 4;

// no-argument ATM constructor initializes instance variables

**public** ATM() {

userAuthenticated = **false**; // user is not authenticated to start

currentAccountNumber = 0; // no current account number to start

screen = **new** Screen(); // create screen

keypad = **new** Keypad(); // create keypad

cashDispenser = **new** CashDispenser(); // create cash dispenser

depositSlot = **new** DepositSlot(); // create deposit slot

bankDatabase = **new** BankDatabase(); // create acct info database

}

// start ATM

**public** **void** run() {

// welcome and authenticate user; perform transactions

**while** (**true**) {

// loop while user is not yet authenticated

**while** (!userAuthenticated) {

screen.displayMessageLine("\nWelcome!");

authenticateUser(); // authenticate user

}

performTransactions(); // user is now authenticated

userAuthenticated = **false**; // reset before next ATM session

currentAccountNumber = 0; // reset before next ATM session

screen.displayMessageLine("\nThank you! Goodbye!");

}

}

// attempts to authenticate user against database

**private** **void** authenticateUser() {

screen.displayMessage("\nPlease enter your account number: ");

**int** accountNumber = keypad.getInput(); // input account number

screen.displayMessage("\nEnter your PIN: "); // prompt for PIN

**int** pin = keypad.getInput(); // input PIN

// set userAuthenticated to boolean value returned by database

userAuthenticated =

bankDatabase.authenticateUser(accountNumber, pin);

// check whether authentication succeeded

**if** (userAuthenticated) {

currentAccountNumber = accountNumber; // save user's account #

}

**else** {

screen.displayMessageLine(

"Invalid account number or PIN. Please try again.");

}

}

// display the main menu and perform transactions

**private** **void** performTransactions() {

// local variable to store transaction currently being processed

Transaction currentTransaction = **null**;

**boolean** userExited = **false**; // user has not chosen to exit

// loop while user has not chosen option to exit system

**while** (!userExited) {

// show main menu and get user selection

**int** mainMenuSelection = displayMainMenu();

// decide how to proceed based on user's menu selection

**switch** (mainMenuSelection) {

// user chose to perform one of three transaction types

**case** ***BALANCE\_INQUIRY***:

currentTransaction = createTransaction(***BALANCE\_INQUIRY***);

currentTransaction.execute();

**break**;

**case** ***WITHDRAWAL***:

currentTransaction = createTransaction(***WITHDRAWAL***);

currentTransaction.execute();

**break**;

**case** ***DEPOSIT***:

currentTransaction = createTransaction(***DEPOSIT***);

currentTransaction.execute();

**break**;

**case** ***EXIT***: // user chose to terminate session

screen.displayMessageLine("\nExiting the system...");

userExited = **true**; // this ATM session should end

**break**;

**default**: // user did not enter an integer from 1-4

screen.displayMessageLine(

"\nYou did not enter a valid selection. Try again.");

**break**;

}

}

}

// display the main menu and return an input selection

**private** **int** displayMainMenu() {

screen.displayMessageLine("\nMain menu:");

screen.displayMessageLine("1 - View my balance");

screen.displayMessageLine("2 - Withdraw cash");

screen.displayMessageLine("3 - Deposit funds");

screen.displayMessageLine("4 - Exit\n");

screen.displayMessage("Enter a choice: ");

**return** keypad.getInput(); // return user's selection

}

// return object of specified Transaction subclass

**private** Transaction createTransaction(**int** type) {

Transaction temp = **null**; // temporary Transaction variable

// determine which type of Transaction to create

**switch** (type) {

**case** ***BALANCE\_INQUIRY***: // create new BalanceInquiry transaction

temp = **new** BalanceInquiry(currentAccountNumber, screen, bankDatabase);

**break**;

**case** ***WITHDRAWAL***: // create new Withdrawal transaction

temp = **new** Withdrawal (currentAccountNumber, screen,bankDatabase, keypad, cashDispenser);

**break**;

**case** ***DEPOSIT***: // create new Deposit transaction

temp = **new** Deposit (currentAccountNumber, screen,bankDatabase, keypad, depositSlot);

**break**;

}

**return** temp; // return the newly created object

}

}

**else** {

screen.displayMessageLine(

"Invalid account number or PIN. Please try again.");

}

}

// display the main menu and perform transactions

**private** **void** performTransactions() {

// local variable to store transaction currently being processed

Transaction currentTransaction = **null**;

**boolean** userExited = **false**; // user has not chosen to exit

// loop while user has not chosen option to exit system

**while** (!userExited) {

// show main menu and get user selection

**int** mainMenuSelection = displayMainMenu();

// decide how to proceed based on user's menu selection

**switch** (mainMenuSelection) {

// user chose to perform one of three transaction types

**case** ***BALANCE\_INQUIRY***:

currentTransaction = createTransaction(***BALANCE\_INQUIRY***);

currentTransaction.execute();

**break**;

**case** ***WITHDRAWAL***:

currentTransaction = createTransaction(***WITHDRAWAL***);

currentTransaction.execute();

**break**;

**case** ***DEPOSIT***:

currentTransaction = createTransaction(***DEPOSIT***);

currentTransaction.execute();

**break**;

**case** ***EXIT***: // user chose to terminate session

screen.displayMessageLine("\nExiting the system...");

userExited = **true**; // this ATM session should end

**break**;

**default**: // user did not enter an integer from 1-4

screen.displayMessageLine(

"\nYou did not enter a valid selection. Try again.");

**break**;

}

}

}

// display the main menu and return an input selection

**private** **int** displayMainMenu() {

screen.displayMessageLine("\nMain menu:");

screen.displayMessageLine("1 - View my balance");

screen.displayMessageLine("2 - Withdraw cash");

screen.displayMessageLine("3 - Deposit funds");

screen.displayMessageLine("4 - Exit\n");

screen.displayMessage("Enter a choice: ");

**return** keypad.getInput(); // return user's selection

}

// return object of specified Transaction subclass

**private** Transaction createTransaction(**int** type) {

Transaction temp = **null**; // temporary Transaction variable

// determine which type of Transaction to create

**switch** (type) {

**case** ***BALANCE\_INQUIRY***: // create new BalanceInquiry transaction

temp = **new** BalanceInquiry(currentAccountNumber, screen, bankDatabase);

**break**;

**case** ***WITHDRAWAL***: // create new Withdrawal transaction

temp = **new** Withdrawal (currentAccountNumber, screen,bankDatabase, keypad, cashDispenser);

**break**;

**case** ***DEPOSIT***: // create new Deposit transaction

temp = **new** Deposit (currentAccountNumber, screen,bankDatabase, keypad, depositSlot);

**break**;

}

**return** temp; // return the newly created object

}

}

// determine which type of Transaction to create

**switch** (type) {

**case** ***BALANCE\_INQUIRY***: // create new BalanceInquiry transaction

temp = **new** BalanceInquiry(currentAccountNumber, screen, bankDatabase);

**break**;

**case** ***WITHDRAWAL***: // create new Withdrawal transaction

temp = **new** Withdrawal (currentAccountNumber, screen,bankDatabase, keypad, cashDispenser);

**break**;

**case** ***DEPOSIT***: // create new Deposit transaction

temp = **new** Deposit (currentAccountNumber, screen,bankDatabase, keypad, depositSlot);

**break**;

}

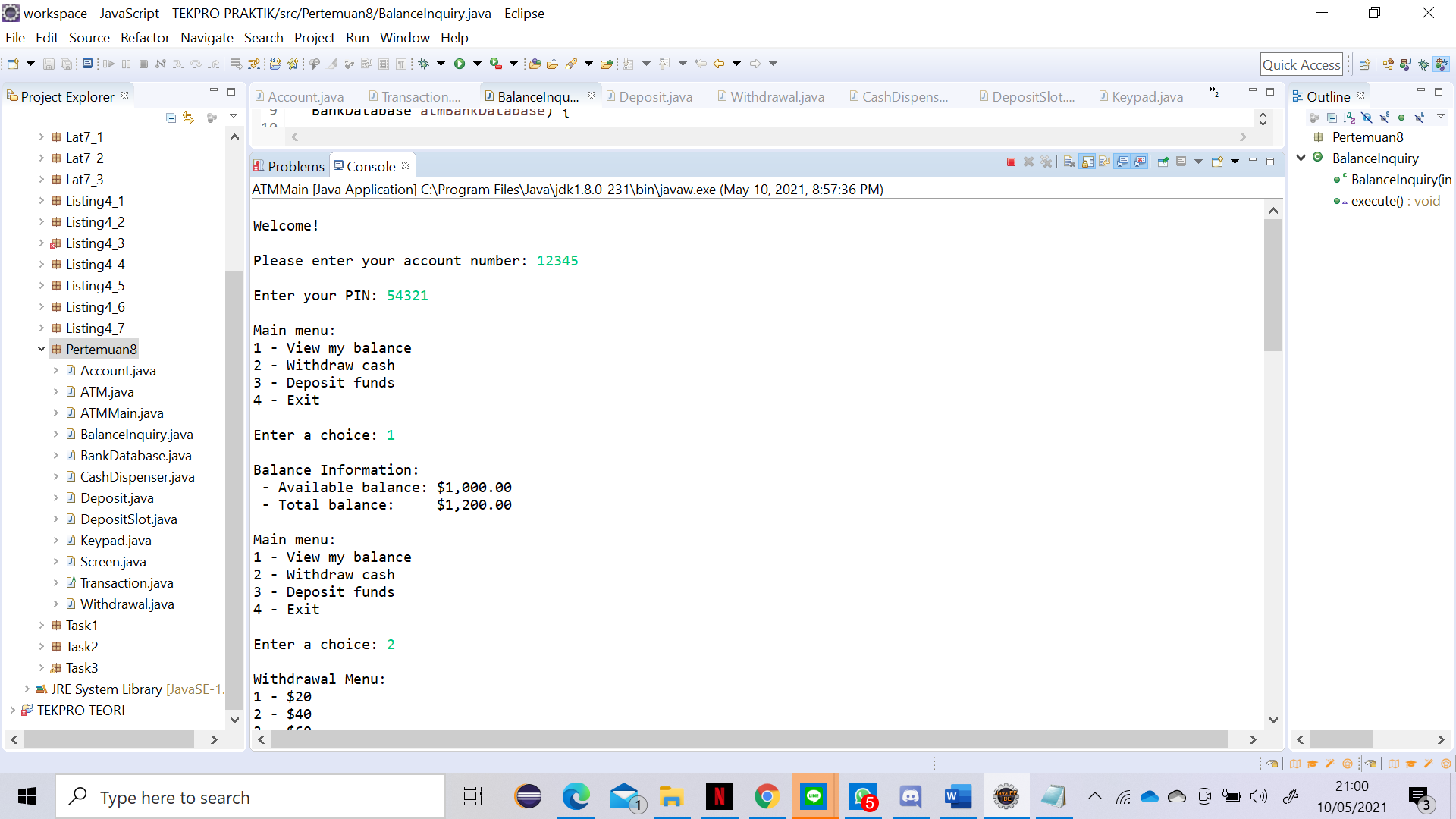
**return** temp; // return the newly created object

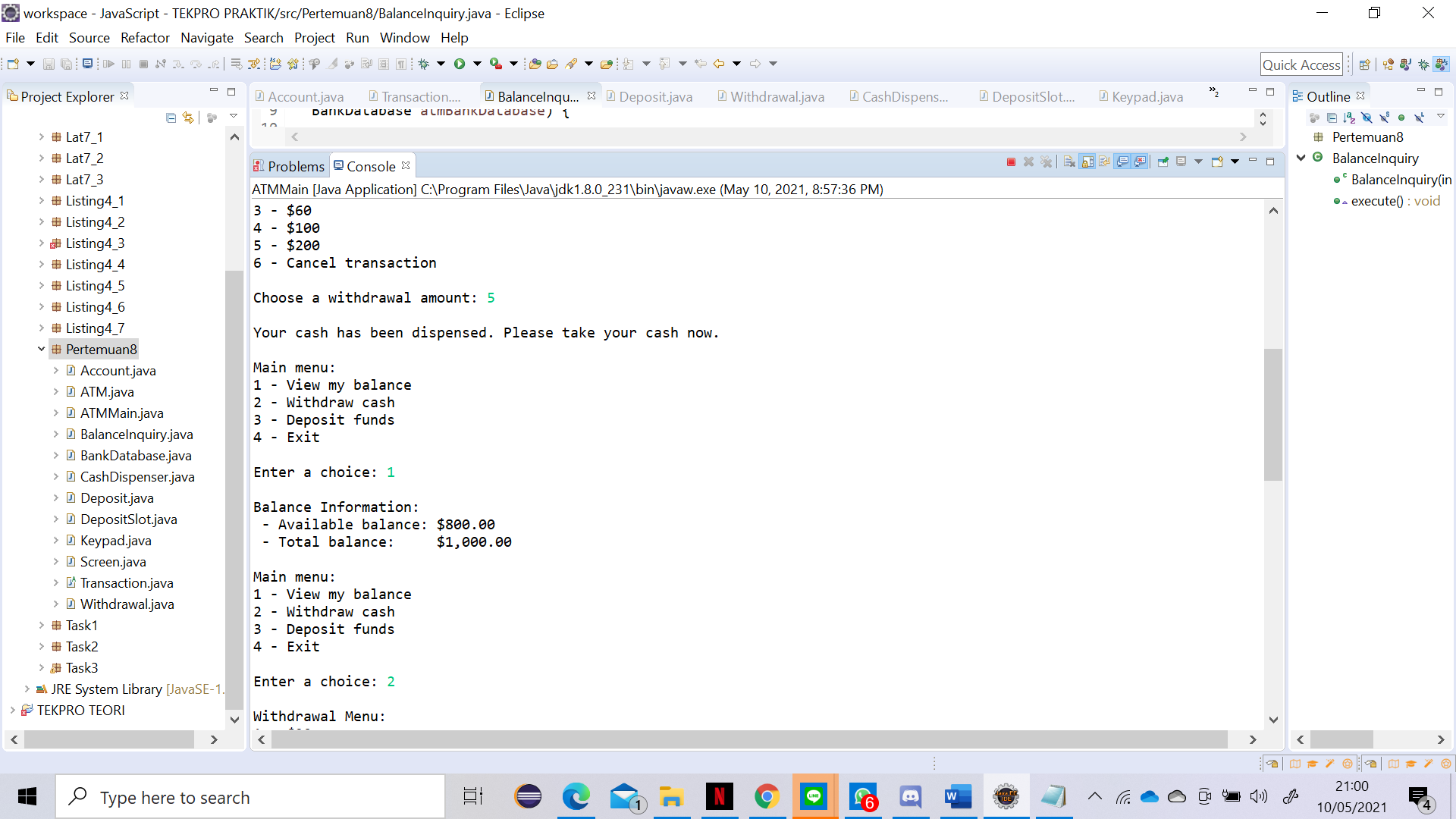
}

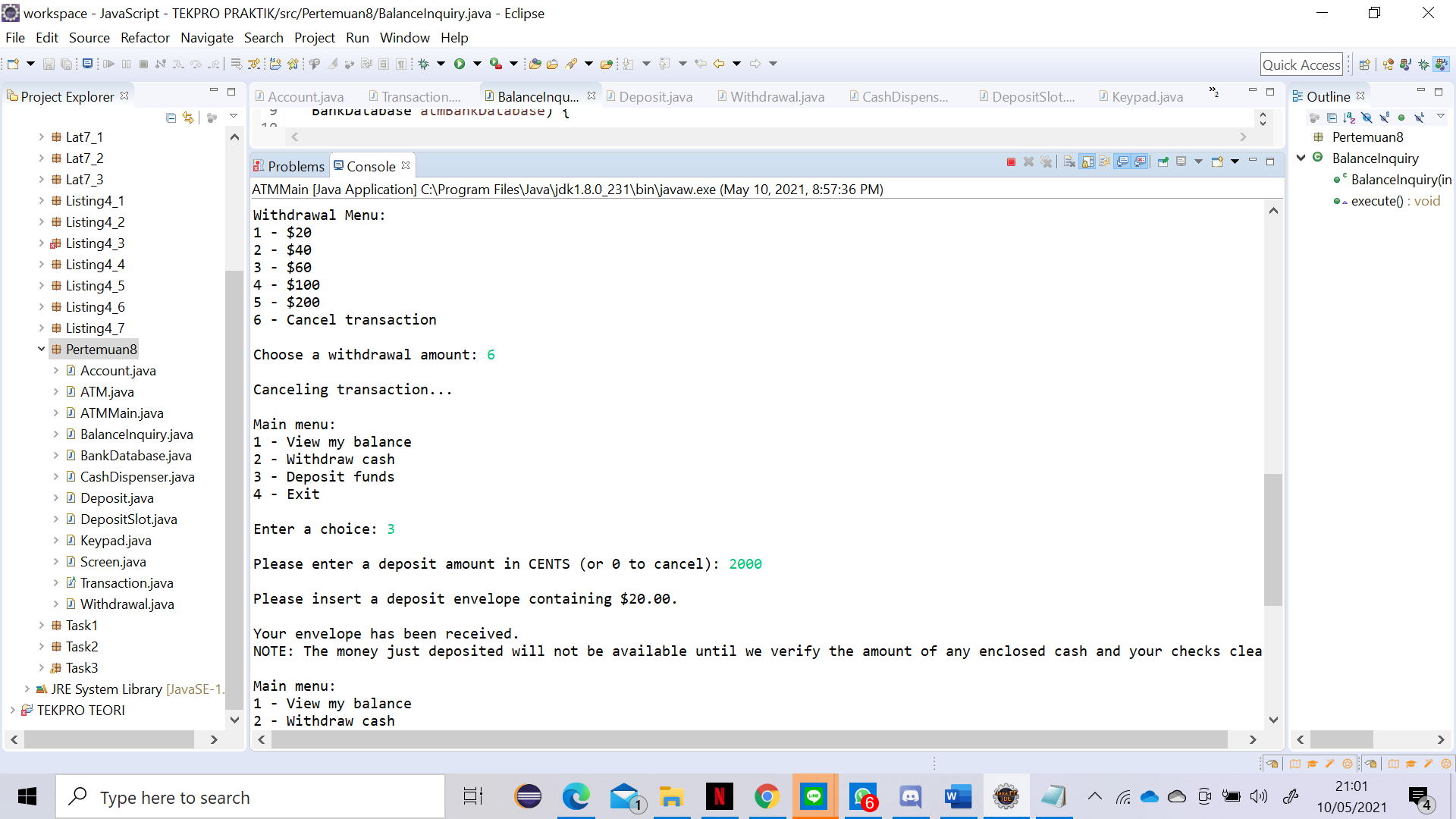
}

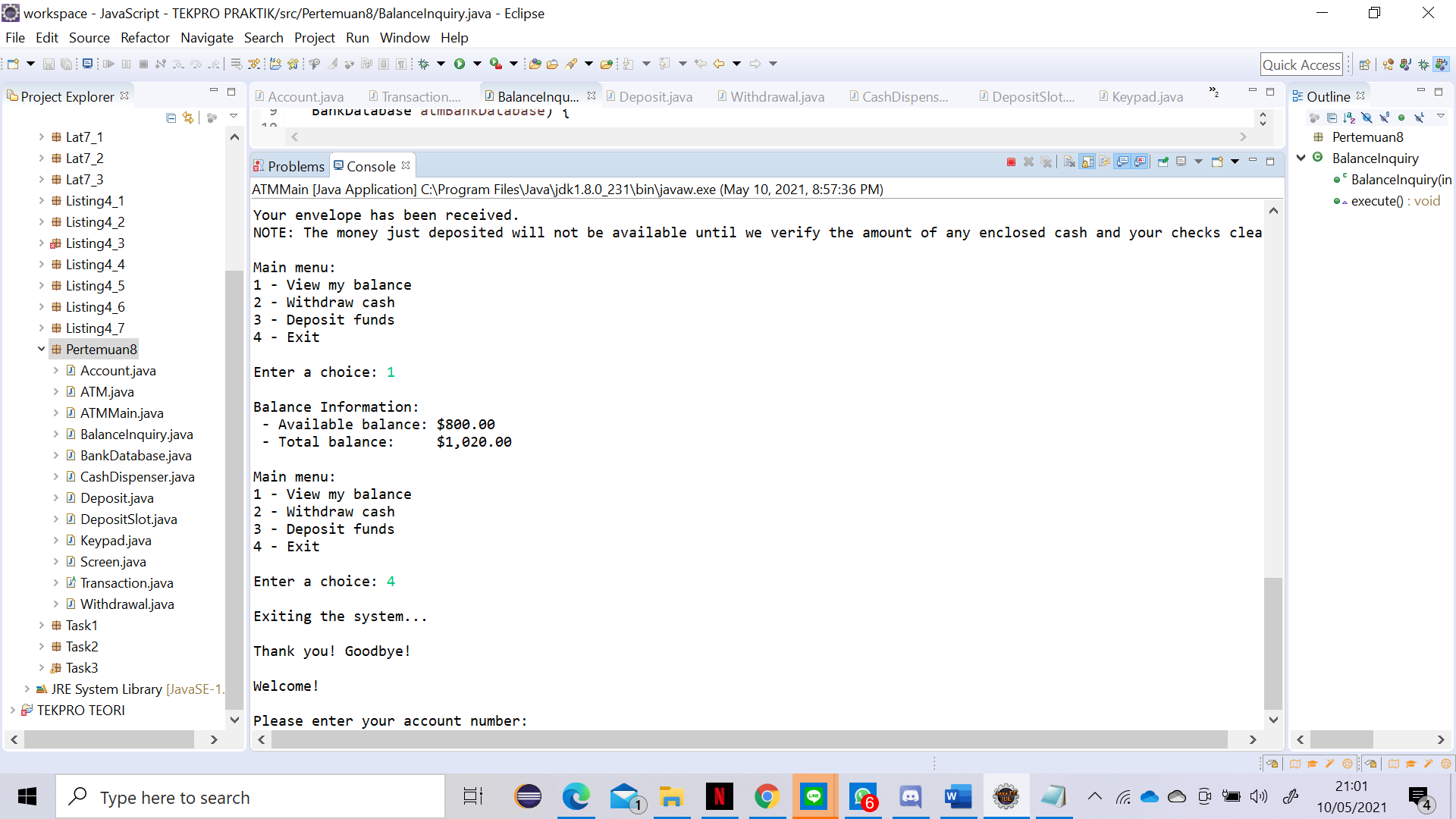
**Hasil Akhir Program**

1. Screenshoot hasil akhir program.









1. Permasalahan yang dihadapi.

Saat program pertama kali di-*compile* dan di-*run*, terdapat kesalahan pada *password*. serta, saat memilih *menu,* tidak tampil pilihan yang diinginkan

1. Solusi dari permasalahan yang dihadapi

Mengubah *password* di kelas BankDatabase.java sesuai dengan yang diminta. Untuk menampilkan menu-menu, saya memanggil modul execute sesuai dengan kelas yang sesuai.

1. Nama teman yang membantu memecahkan permasalahan di persoalan ini

* Syahda
* M Faza H