You will create a bank application which can open new accounts, modify existing accounts and close accounts. An account has a number associated with it and transactions, such as deposits and withdrawals require the account number.

**Specifications**

The Local Bank application allows new accounts to be opened, modified and closed. Each account has a unique account number, which is required for all transactions. Transactions include deposits and withdrawals. An account balance can also be checked.

The application should provide a menu of options. Depending on the option selected, additional input may be needed. When a transaction is performed, updated account information should be displayed.

The application can be modeled with a Bank object, account objects and customer objects. The bank object should store the accounts, add and remove accounts and perform transactions that also display updated account information.

The account objects should store the current balance, make deposits, make withdrawals and return the current balance and account ID. An account object must also have an equals() method for the Bank object to use when finding an account.

The customer objects should store the account holder’s name. The bank object will display account information, so the account and customer objects will override the toString() methods.

You must also be able to create a list of all customers (first and last names), account id’s, corresponding balances. There should be a minimum of 5 records in your report.

As always, you must ensure you integrate error checking logic.

**Sample Screen Layout**

Your screen must look like this or better

**Welcome to Local Bank Application!**

**Deposit (D), Withdrawal (W), Check Balance (B), Report for all Customer Accounts (R)**

**Create Account (A), Remove Account (X)**

**Enter your selection: A**

**First name: Adriana**

**Last name: Lee**

**Opening Balance: $1000**

**Account created. Account id is: ALee**

**Deposit (D), Withdrawal (W), Check Balance (B), Report for all Customer Accounts (R)**

**Create Account (A), Remove Account (X)**

**Enter your selection: W**

**Enter ID: Alee**

**Withdrawal amount: $122**

**Current balance is: $878**

**Employee (E), Pay (P), Quit (Q)**

**Enter your selection: P**

**Enter employee number (1, 2, or 3): 1**

**Select the Annual (A), Monthly (M) or Weekly (W) pay for Dan Martin: 2**

**Dan Martin, manager**

**$5,416.67**

**Employee (E), Pay (P), Quit (Q)**

**Enter your selection: Q**

***Thanks for using SalesCentre***

**Program Checklist**

|  |  |  |
| --- | --- | --- |
|  | **Description** | **Marks** |
| **Knowledge** | **Programming Concepts:**   * Appropriate declaration of variables with meaningful names and  suitable data types and creation and use of at least one 2-D array * Appropriate use of recursion * Appropriate use of object classes for input and output * Appropriate use of arithmetic operators to perform program calculations * Appropriate use of methods , objects and classes * Appropriate use of conditions & loops |  |
| **Thinking** | **Algorithms:**   * IPO Chart, pseudo code, or flowchart provides detailed step-by-step instructions to properly implement the program specifications. |  |
| **Communication** | **Method documentation:**   * Uses pre and post method documentation   **Program Header:**   * contains programmer’s name, course code, date program written, program name and a comprehensive description of the purpose of the program   **Internal Documentation:**   * comments are used appropriately within the program and provide a meaningful summary of major processes   **Formatting:**   * program source code is properly indented where required and contains appropriate white space for readability * User interface is courteous, esthetically pleasing, and free of spelling and grammar errors |  |
| **Application** | **Implementation:**   * Output is formatted exactly as displayed in the sample layout User input is formatted as specified * Program source code is efficient and executes as required with no syntax or logic errors * Error checking logic is used on user input * Error debugging is illustrated * Client file + at least 2 other files * **Bonus**: up to 1 additional bonus mark for improving the user interface beyond what was provided in the specification |  |