Assignment 5

course_code: COMP4003

name: Emma Orhun

student_num: 101071651

date_completed: Dec. 13th 2019

Part One

```
drop table branch;
drop table account;
drop table customer;
drop table HelperTable;
drop table BranchNumTable;
drop table AccountNumTable;

create table branch
   (b_num char(3) primary key,
   address varchar(20) not null unique);

CREATE SEQUENCE branch_sequence START WITH 0 INCREMENT BY 1 minvalue 0;
CREATE OR REPLACE TRIGGER branch_on_insert BEFORE INSERT ON branch FOR EACH row BEGIN SELECT branch_sequence.nextval INTO :new.b_num

create table customer
   (c_num char(5) primary key,
   name varchar(10) not null unique);

CREATE SEQUENCE customer_sequence START WITH 0 INCREMENT by 1 minvalue 0;
CREATE OR REPLACE TRIGGER customer_on_insert BEFORE INSERT ON Customer FOR EACH row BEGIN SELECT customer_sequence.nextval INTO :new

create table account
   (a_num char(7) primary key,
   c_num char(5),
   balance integer,
   foreign key (c_num) references customer(c_num) on delete cascade)

CREATE SEQUENCE account_sequence START WITH 0 INCREMENT BY 1 minvalue 0;
CREATE OR REPLACE TRIGGER account_on_insert BEFORE INSERT ON Account FOREACH row BEGIN SELECT account_sequence.nextval INTO :new.a_n

create table HelperTable
   (branch_num integer primary key,
   customer_num integer,
   account_num integer);

create table BranchNumTable
  (num integer);
create table AccountNumTable
  (num integer);
```

Part Two

```
CREATE OR REPLACE PACKAGE Bank IS
    PROCEDURE branch (v_address Branch.address%TYPE);
    PROCEDURE open_branch (v_address Branch.address%TYPE);
    PROCEDURE close_branch (v_address Branch.address%TYPE);
    PROCEDURE show_branch(v_b_num Branch.b_num%TYPE);
    PROCEDURE show_all_branches();
    PROCEDURE create_customer(v_name Customer.name%TYPE);
    PROCEDURE remove_customer(v_name Customer.name%TYPE);
    PROCEDURE show_customer(v_name Customer.name%TYPE);
    PROCEDURE open_account(v_name Customer.name%TYPE, v_address Branch.address%TYPE, v_amount Account.balance%TYPE);
    PROCEDURE close_account(v_a_num Account.a_num%TYPE);
    PROCEDURE withdraw(v_a_num account.a_num%TYPE, amount Account.balance%TYPE);
    PROCEDURE deposit(v_a_num account.a_num%TYPE, v_amount Account.balance%TYPE);
    PROCEDURE transfer(v_accountnum1 Account.a_num%TYPE, v_accountnum2 Account.a_num%TYPE, v_amount Account.balance%TYPE);
end Bank;
/

CREATE OR RELEASE PACKAGE Bank AS
    PROCEDURE Branch (v_address Branch.address%TYPE) is
        DECLARE
        does_not_exist EXCEPTION;
        v_b_num branch.b_num%TYPE;
        BEGIN
            IF EXISTS (SELECT b_num FROM Branch INTO v_b_num WHERE address = v_address) THEN
                 dbms_output.put_line('b_num: ' || v_b_num);
            ELSE RAISE does_not_exist
            ENDIF;
        EXCEPTION
            WHEN does_not_exist THEN
            dbms_output.put_line('That branch does not exist');
        END;

    PROCEDURE open_branch(v_address in branch.address%TYPE) is
        DECLARE
            already_exists EXCEPTION;
        BEGIN
            if not exists (select * from branch where address = v_address) THEN
                insert into branch values (address) (v_address);
            else raise already_exists
            endif;
        EXCEPTION
            when already_exists THEN
            dbms_output.put_line('That branch already exists!');
        END;

    PROCEDURE close_branch(v_address IN branch.address%TYPE) IS
        DECLARE
            does_not_exist EXCEPTION;
        BEGIN
            IF EXISTS (SELECT * FROM branch WHERE address = v_address) THEN
                DELETE FROM branch WHERE address = v_address);
            ELSE RAISE does_not_exist
            ENDIF;
        EXCEPTION
            when does_not_exist THEN
            dbms_output.put_line('That branch doesn't exist');
        END;

    PROCEDURE create_customer(v_name IN customer.name%TYPE) IS
        DECLARE
        already_exists EXCEPTION;
        BEGIN
            IF NOT EXISTS (select * from customer where name = v_name) THEN
                INSERT INTO customer VALUES (name)(v_name);
            ELSE RAISE already_exists
            ENDIF;
        EXCEPTION
            when already_exists THEN
            dbms_output.put_line('That customer already exists!');
        END;

    PROCEDURE show_customer(v_c_num IN customer.name%TYPE) IS
        DECLARE
        does_not_exist EXCEPTION;
        v_name Customer.name%TYPE;
        v_a_num Account.a_num%TYPE;
        v_balance Account.balance%TYPE;

        CURSOR AC IS SELECT  a_num, balance INTO v_a_num, v_balance FROM Account where v_c_num LIKE c_num;
        BEGIN
            IF EXISTS (select c_num, name from Customer where c_num = v_c_num) THEN
                 SELECT c_num, name FROM Customer INTO v_c_num, v_name where c_num = v_c_num
                 dbms_output.put_line('CustomerNumber: ' || v_c_num || ', Name:' || v_name);
                 FOR at in ac
                    LOOP
                    DBMS_OUTPUT.PUT_LINE('Account number: ' || ac.column1 ||
                               ', balance: ' || ac.column2);
                    END LOOP;
            ELSE RAISE does_not_exist
            ENDIF;
        EXCEPTION
            WHEN does_not_exist THEN
            dbms_output.put_line('That customer does not exist.');
        END;

    PROCEDURE remove_customer(v_name IN customer.name%TYPE) IS
        DECLARE
            does_not_exist EXCEPTION;
        BEGIN
            IF EXISTS (SELECT * FROM customer WHERE name = v_name) THEN
                DELETE FROM customer WHERE name = v_name);
            ELSE RAISE does_not_exist
            ENDIF;
        EXCEPTION
            when does_not_exist THEN
            dbms_output.put_line('That customer does not exist');
        END;

    PROCEDURE open_account(v_name IN customer.name%TYPE, v_address IN branch.address%TYPE, v_amount IN account.balance%TYPE) IS
        DECLARE
            does_not_exist EXCEPTION;
            negative_amount EXCEPTION;
        BEGIN
            IF v_amount > 0 THEN
                IF EXISTS (select * from Customer where name = v_name) THEN
                    IF EXISTS (select * from branch where address = v_address) THEN
                        INSERT INTO account VALUES (balance, c_num) (balance, c_num));
                    ELSE RAISE does_not_exist;
                    ENDIF;
                ELSE RAISE does_not_exist;
                ENDIF;
            ENDIF;
        EXCEPTION
            when does_not_exist THEN
            dbms_output.put_line('That customer or branch does not exist);
            when negative_amount THEN
            dbms_output.put_line('Can't deposit a negative amount!');
        END;

        PROCEDURE close_account(v_a_num IN account.a_num%TYPE) IS
        DECLARE
            does_not_exist EXCEPTION;
        BEGIN
            IF EXISTS (SELECT * FROM account WHERE a_num = v_a_num) THEN
                DELETE FROM account WHERE a_num = v_a_num);
            ELSE RAISE does_not_exist
            ENDIF;
        EXCEPTION
            when does_not_exist THEN
            dbms_output.put_line('The account does not exist.');
        END;

        PROCEDURE deposit(v_a_num in account.a_num%TYPE, v_amount in account.balance%TYPE) is
        DECLARE
        does_not_exist EXCEPTION
        negative_amount EXCEPTION
        begin
            if v_amount >= 0 then
                if exists (select * FROM account where a_num = v_a_num) THEN
                    UPDATE account SET balance = balance + v_amount WHERE a_num = v_a_num;
                ELSE RAISE does_not_exist
                ENDIF;
            ELSE RAISE negative_amount
            ENDIF;
        EXCEPTION
            when does_not_exist THEN
            dbms_output.put_line('That account does not exist!');
            when negative_amount THEN
            dbms_output.put_line('Can't deposit a negative amount!');
        END;

        PROCEDURE withdraw(v_a_num IN account.a_num%TYPE, amount IN account.balance%TYPE) IS
            DECLARE
            does_not_exist EXCEPTION
            insufficient_funds EXCEPTION
            BEGIN
            IF EXISTS (select * from account where a_num = v_a_num) THEN
                IF v_amount >= balance THEN
                        UPDATE account SET balance = balance - v_amount WHERE a_num = v_a_num;
                    ELSE RAISE insufficient_funds
                    ENDIF;
                ELSE RAISE does_not_exist
                ENDIF;
            EXCEPTION
                when does_not_exist THEN
                dbms_output.put_line('The account does not exist.');
                when negative_amount THEN
                dbms_output.put_line('Insufficient funds!');
            END;

        PROCEDURE transfer(v_accountnum1 account.a_num%TYPE, v_accountnum2 account.a_num%TYPE, v_amount account.balance%TYPE); is
            DECLARE
                does_not_exist EXCEPTION
            BEGIN
            IF EXISTS (select * from account where a_num = v_accountnum1) THEN
                CALL withdraw(v_accountnum1, v_amount);
            ELSE RAISE does_not_exist
            ENDIF;
            IF EXISTS (SELECT * FROM account WHERE a_num = v_accountnum2) THEN
                CALL deposit(v_accountnum2, v_amount);
            ELSE RAISE does_not_exist
            ENDIF;
            EXCEPTION
                when does_not_exist THEN
                dbms_output.put_line('The account does not exist.')';
                when negative_amount THEN
                dbms_output.put_line('Insufficient funds!');
            END;

        PROCEDURE show_branch(v_address IN branch.address%TYPE) IS
            DECLARE
            does_not_exist EXCEPTION
            v_b_num branch.b_num%TYPE
            v_a_num account.a_num%TYPE
            v_c_num account.c_num%TYPE
            v_balance account.balance%TYPE
            cursor ac IS select a_num, c_num, balance into v_a_num, v_c_num, v_balance from account where  substr(a_num, 1, 3) = v_b
            BEGIN
            IF EXISTS (SELECT address FROM branch where  address = v_address)
                SELECT b_num, address FROM branch INTO v_b_num, v_address where address = v_address
                dbms_output.put_line('Branch number:' || v_b_num || ', address=' || v_address);
                FOR at in ac
                LOOP
                DBMS_OUTPUT.PUT_LINE('Account number: ' || ac.v_a_num ||
                                   ', Customer number: ' || ac.v_c_num ||
                                   ', Balance: ' || ac.v_balance );
                END LOOP;
            ELSE RAISE does_not_exist
            ENDIF;
            EXCEPTION
                when does_not_exist THEN
                dbms_output.put_line('That branch does not exist)';
            END;

        PROCEDURE show_all_branches IS
            DECLARE
            cursor ac is SELECT b_num FROM branch;
            BEGIN
                FOR at in ac
                    LOOP
                    CALL show_branch(ac.column1);
                    END LOOP;
            END;
END Bank;
/
```

Part Three

```
Eimport java.io.*;
import java.sql.*;
import java.util.*;
import oracle.sql.*;
import oracle.jdbc.*;
```

```java
public class bank {
   static Connection conn = null;
   static Statement stmt = null;
   static CallableStatement cstmt = null;
   static Statement stmt2 = null;
   static ResultSet result = null;
   static ResultSet result2 = null;

   static int branch_num = 0;
   static int customer_num = 0;
   static int account_num = 0;

   public static Connection connectToJDBC() {
       try {
           DriverManager.registerDriver
            (new oracle.jdbc.driver.OracleDriver());

           System.out.println("Connecting to JDBC!");
           conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","fedora","oracle2019");

           System.out.println("JDBC connected!\n");
           return conn;
       } catch(Exception e) {
           System.out.println("Exception in connectToJDBC:");
           e.printStackTrace();
           return null;
       }

   }

   public static void closeJdbcConnection(Connection conn, CallableStatement stmt, ResultSet rs) {
       try {
           if (stmt != null) {
               stmt.close();
               System.out.println("stmt.close() successful!");
           }
           if (stmt2 != null) {
               stmt2.close();
               System.out.println("stmt2.close() successful!");
           }
           if (conn != null) {
               conn.close();
               System.out.println("conn.close() successful!");
           }
           if (rs != null) {
               rs.close();
               System.out.println("rs.close() successful!");
           }
           if(rs2 != null) {
               rs2.close();
               System.out.println("rs2.close() successful!");
           }
       } catch (Exception e)  {
           System.out.println("Exception in closeJdbcConnection: ");
           e.printStackTrace();
           System.exit(-1);
       }
   }

   public static void printBranches()  {
       try {
           rs=stmt.executeQuery
           ("select b_num, address from branch");
           System.out.print("\n\n~~~~~~~~ BRANCHES ~~~~~~~~\n");
           System.out.println("BranchNum          Address");
           System.out.println("-------------------------------");
           while(rs.next())
           {
               System.out.print(rs.getString("Num")+"                 ");
               System.out.print(rs.getString("Address")+"\n");
           }
       } catch(Exception e) {
           System.out.println("Exception in printBranches(): ");
           e.printStackTrace();
           System.exit(-1);
       }
       System.out.print("\n");
   }

   public static void printCustomers() {
       try {
```

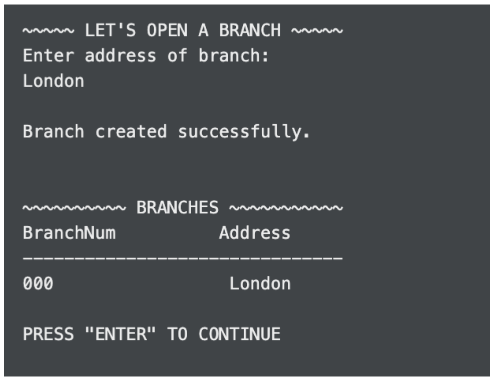
```
            rs=stmt.executeQuery
            ("select c_num, name from Customer");
            System.out.print("\n\n~~~~~~~ CUSTOMER ~~~~~~~\n");
            System.out.println(" CustomerNum         Name");
            System.out.println("------------------------------");
            while(rs.next())
            {
                System.out.print(rs.getString("Customer Number")+"           ");
                System.out.print(rs.getString("Name")+"   \n");
            }
        } catch(Exception e) {
            System.out.println("Exception in printCustomers(): ");
            e.printStackTrace();
            System.exit(-1);
        }
        System.out.print("\n");
    }

    public static void printAccounts() {
        try {
            rs=stmt.executeQuery
            ("select a_num, c_num, balance from account");
            System.out.print("\n~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~\n");
            System.out.println("AccountNum    CustomerNum   Balance");
            System.out.println("-----------------------------------");
            while(rs.next()) {
                System.out.print(" " + rs.getString("a_num")+"   ");
                System.out.print(rs.getString("c_num")+"     ");
                System.out.print(rs.getInt("balance")+"\n");
            }
            System.out.print("\n\n");
        } catch(Exception e) {
            System.out.println("Exception in printAccounts(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void menu() {
        Scanner sc = new Scanner(System.in);
        int exit = 0;
        while(exit == 0) {
            sc.reset();
            System.out.print("\033[H\033[2J");
            System.out.flush();
            System.out.println("----------------------------------- Bankify ---------------------------------------\n");
            System.out.println("~~~~~~~~ MAIN MENU ~~~~~~~~");
            System.out.println("1) Open a branch");
            System.out.println("2) Close a branch");
            System.out.println("3) Create a customer");
            System.out.println("4) Remove a customer");
            System.out.println("5) Open an account");
            System.out.println("6) Close an account");
            System.out.println("7) Withdraw some money");
            System.out.println("8) Deposit some money");
            System.out.println("9) Transfer some money");
            System.out.println("10) Show a branch");
            System.out.println("11) Show all branches");
            System.out.println("12) Show a customer");
            System.out.println("13) Quit");
            System.out.println("\n-------------------------------------------------------------------------------\n");
            System.out.print("Enter the number of desired action:");
            String user_input = sc.nextLine();
            int input = 0;
            try {
                input = Integer.parseInt(user_input);
            } catch(NumberFormatException e) {
                System.out.println("Please enter valid input!");
                continue;
            }

            if(input < 0 || input > 13) {
                System.out.println("Please enter valid input!");
                continue;
            } else {
                switch (input) {
                    case 1:
                        System.out.println("\n~~~~~ LET'S OPEN A BRANCH ~~~~~\n");
                        open_branch();
                        System.out.println("PRESS \"ENTER\" TO CONTINUE");
                        sc.nextLine();
```

```
                    continue;
                case 2:
                    System.out.println("\n~~~~~~~~~ CLOSE BRANCH ~~~~~~~~~~\n");
                    close_branch();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 3:
                    System.out.println("\n~~~~~~~~ NEW CUSTOMER ~~~~~~~~\n");
                    create_customer();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                       sc.nextLine();
                    continue;
                case 4:
                    System.out.println("\n~~~~~~~~ REMOVE CUSTOMER ~~~~~~~~\n");
                    remove_customer();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                       sc.nextLine();
                    continue;
                case 5:
                    System.out.println("\n~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~\n");
                    open_account();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                       sc.nextLine();
                    continue;
                case 6:
                    System.out.println("\n~~~~~~~~~~~  CLOSE ACCOUNT ~~~~~~~~~~~\n");
                    close_account();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 7:
                    System.out.println("\n~~~~~~~~~~~ WITHDRAW MONEY ~~~~~~~~~~~~\n");
                    withdraw();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 8:
                    System.out.println("\n~~~~~~~~~~~~~ DEPOSIT ~~~~~~~~~~~~~\n");
                    deposit();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 9:
                    System.out.println("\n~~~~~~~~~~~~~ TRANSER ~~~~~~~~~~~~~\n");
                    transfer();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 10:
                    System.out.println("\n~~~~~~~~~ SHOW BRANCH ~~~~~~~~~\n");
                    show_branch();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 11:
                    System.out.println("\n~~~~~~~~~~~ BRANCHES ~~~~~~~~~~~\n");
                    show_all_branches();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 12:
                    System.out.println("\n~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~\n");
                    show_customer();
                    System.out.println("PRESS \"ENTER\" TO CONTINUE");
                    sc.nextLine();
                    continue;
                case 13:
                    System.out.println("Quitting...");
                    exit = 1;
                    sc.close();
                    break;

            }
        }
    }
}

public static void open_branch() {
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter branch address:");
```

```
        String address = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call open_Branch(?)}");
            cstmt.setString(1, address);
            cstmt.execute();
            System.out.println("Branch created successfully.");
        } catch(Exception e) {
            System.out.println("Exception in open_branch(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void close_branch() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter address of branch: ");
        String address = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call close_Branch(?)}");
            cstmt.setString(1, address);
            cstmt.execute();
            System.out.println("Branch deleted successfully.");
        } catch(Exception e) {
            System.out.println("Exception in close_branch(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void create_customer() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter customer name:");
        String customer_name = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call create_customer(?)}");
            cstmt.setString(1, customer_name);
            cstmt.execute();
             System.out.println("Customer created successfully.");
        }
        catch(Exception e) {
            System.out.println("Exception in create_customer(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void remove_customer() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter customer name:");
        String customer_name = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call remove_customer(?)}");
            cstmt.setString(1, customer_name);
            cstmt.execute();
            System.out.println("Customer removed successfully.");
        }
        catch(Exception e) {
            System.out.println("Exception in remove_customer(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void close_account() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account number:");
        String account_num = sc.nextLine();
         try {
            cstmt = conn.prepareCall("{call close_account(?)}");
            cstmt.setString(1, account_num);
            cstmt.execute();
               System.out.println("Account successfully removed.");
        } catch(Exception e) {
            System.out.println("Exception in close_account(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static  void open_account() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter customer name:");
        String customer_name = sc.nextLine();
        System.out.print("Enter branch address:");
        String address = sc.nextLine();
        System.out.print("Enter amount to deposit:");
        Float amount = sc.nextLine();

        try {
            cstmt = conn.prepareCall("{call open_account(?, ?, ?)}");
            cstmt.setString(1, customer_name);
            cstmt.setString(2, address);
            cstmt.setFloat(3, amount);
            cstmt.execute();
             System.out.println("Account created successfully.");
        }
        catch(Exception e) {
            System.out.println("Exception in open_account()");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void  withdraw() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account number to withdraw from:");
        String account_num = sc.nextLine();
        System.out.print("Enter amount to withdraw:");
        Float amount = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call withdraw(?, ?)}");
            cstmt.setString(1, account_num);
            cstmt.setFloat(2, amount);
            cstmt.execute();
             System.out.println("Withdraw successfully completed!);
        } catch(Exception e) {
            System.out.println("Exception in withdraw(): ");
            e.printStackTrace();
            System.exit(-1);
        }

    }

    public static void transfer() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account number to withdraw from:");
        String account_num_1 = sc.nextLine();
        System.out.print("Enter account number to deposit to:");
        String account_num_2 = sc.nextLine();
        System.out.print("Enter amount to transfer:");
        Float amount = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call transfer(?, ?, ?)}");
            cstmt.setString(1, account_num_1);
            cstmt.setString(2, account_num_2);
            cstmt.setFloat(3, amount);
            cstmt.execute();
             System.out.println("The transfer was a success!");
        } catch(Exception e) {
            System.out.println("Exception in withdraw(): ");
            e.printStackTrace();
            System.exit(-1);
        }

    }

    public static void deposit() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter account number:");
        String account_num = sc.nextLine();
        System.out.print("Enter amount to deposit:");
        Float amount = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call deposit(?, ?)}");
            cstmt.setString(1, account_num);
            cstmt.setFloat(2, amount);
            cstmt.execute();
             System.out.println("The deposit was a success!");
        } catch(Exception e) {
            System.out.println("Exception in withdraw(): ");
            e.printStackTrace();
            System.exit(-1);
        }

    }

    public static void show_all_branches() {
        try {
            cstmt = conn.prepareCall("{call show_all_branches()}");
            cstmt.execute();
        } catch(Exception e) {
            System.out.println("Exception in show_all_branches(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void show_branch() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter branch address:");
        String address = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call show_branch(?)}");
            cstmt.setString(1, address);
            cstmt.execute();
        } catch(Exception e){
            System.out.println("Exception in show_branch(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }

    public static void show_customer() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter customer name:");
        String customer_name = sc.nextLine();
        try {
            cstmt = conn.prepareCall("{call show_customer(?)}");
            cstmt.setString(1, customer_name);
            cstmt.execute();
        }
        catch(Exception e) {
            System.out.println("Exception in show_customer(): ");
            e.printStackTrace();
            System.exit(-1);
        }
    }
    public static void main(String[] args){
        try {
            if(connectToJDBC() == null) {
                System.out.println("Connection failed!");
                System.exit(-1);
            }
            stmt = conn.createStatement();
            menu();
        } catch(Exception e){
            System.out.println("Exception in main(): ");
            e.printStackTrace();
            System.exit(-1);
        } finally {
            closeJdbcConnection(conn, cstmt, rs);
        }
    }

}
```

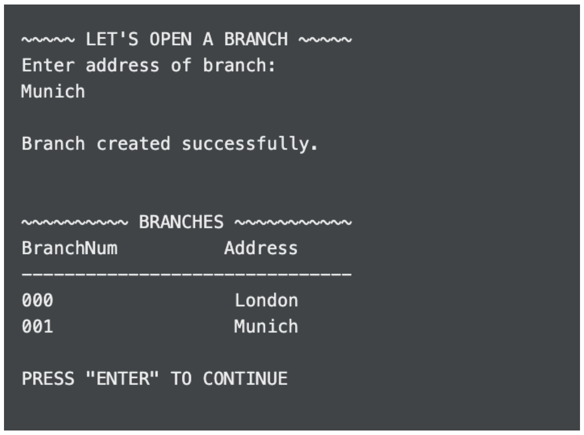
Part 4

2.



```
~~~~~ LET'S OPEN A BRANCH ~~~~~
Enter address of branch:
London

Branch created successfully.


~~~~~~~~~~ BRANCHES ~~~~~~~~~~~
BranchNum          Address
-------------------------------
000                 London

PRESS "ENTER" TO CONTINUE
```

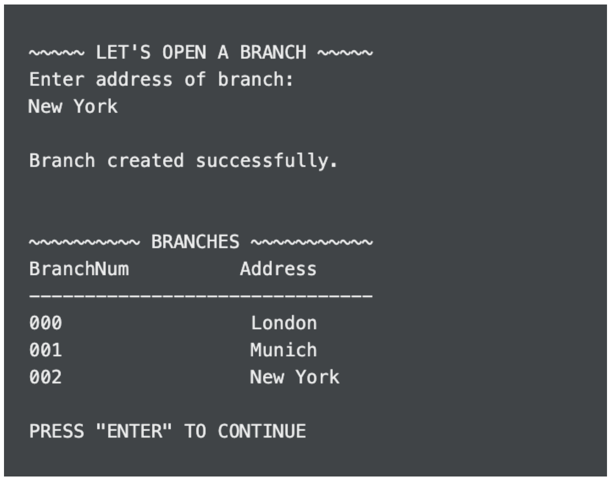
1.



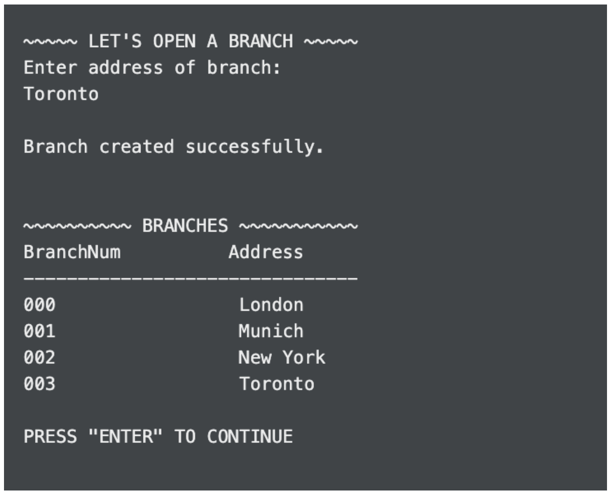
```
~~~~~ LET'S OPEN A BRANCH ~~~~~
Enter address of branch:
Munich

Branch created successfully.


~~~~~~~~~~ BRANCHES ~~~~~~~~~~~
BranchNum          Address
-------------------------------
000                 London
001                 Munich

PRESS "ENTER" TO CONTINUE
```

3.

4.



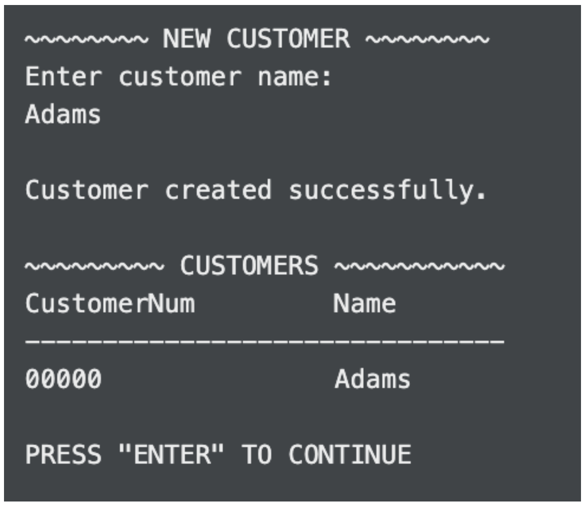
```
~~~~~ LET'S OPEN A BRANCH ~~~~~
Enter address of branch:
New York

Branch created successfully.


~~~~~~~~~~ BRANCHES ~~~~~~~~~~~
BranchNum          Address
-------------------------------
000                 London
001                 Munich
002                 New York

PRESS "ENTER" TO CONTINUE
```



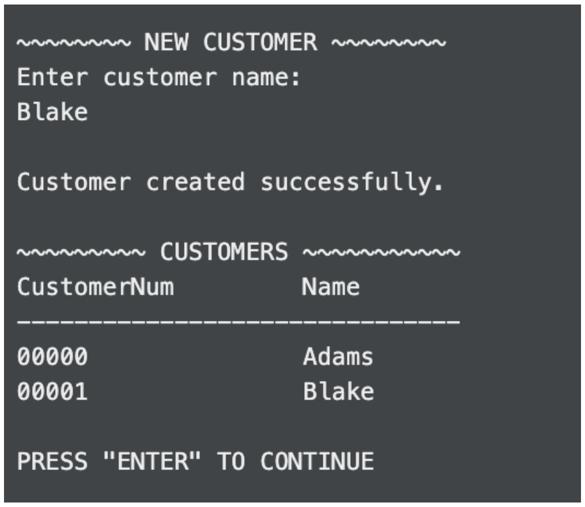
```
~~~~~ LET'S OPEN A BRANCH ~~~~~
Enter address of branch:
Toronto

Branch created successfully.


~~~~~~~~~~ BRANCHES ~~~~~~~~~~~
BranchNum          Address
-------------------------------
000                 London
001                 Munich
002                 New York
003                 Toronto

PRESS "ENTER" TO CONTINUE
```

5.

6.



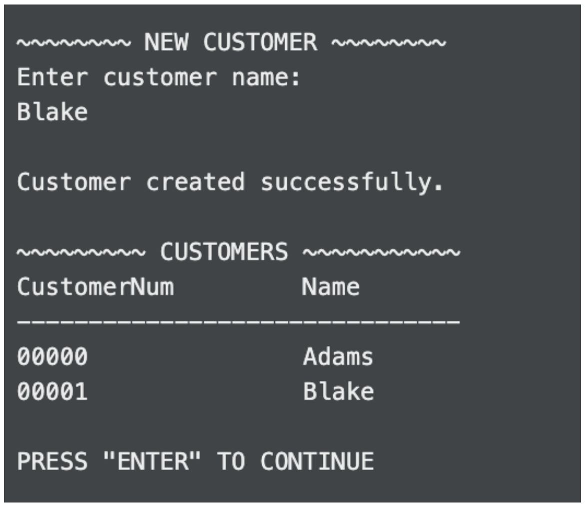
```
~~~~~~~~ NEW CUSTOMER ~~~~~~~~
Enter customer name:
Adams

Customer created successfully.

~~~~~~~~~ CUSTOMERS ~~~~~~~~~~~
CustomerNum         Name
-------------------------------
00000               Adams

PRESS "ENTER" TO CONTINUE
```



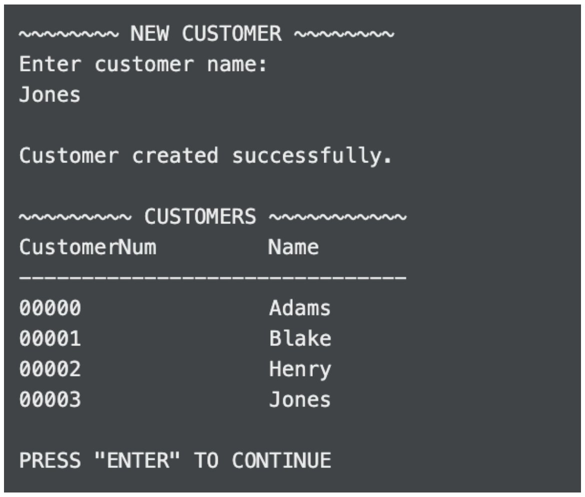
```
~~~~~~~~ NEW CUSTOMER ~~~~~~~~
Enter customer name:
Blake

Customer created successfully.

~~~~~~~~~ CUSTOMERS ~~~~~~~~~~~
CustomerNum         Name
-------------------------------
00000               Adams
00001               Blake

PRESS "ENTER" TO CONTINUE
```

7.

8.



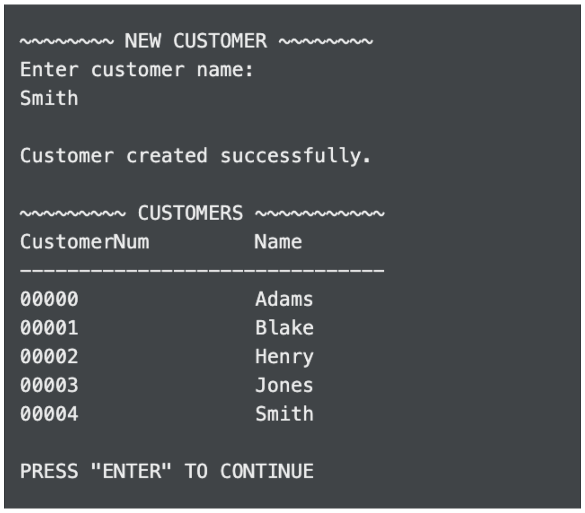
```
~~~~~~~~ NEW CUSTOMER ~~~~~~~~
Enter customer name:
Blake

Customer created successfully.

~~~~~~~~~ CUSTOMERS ~~~~~~~~~~
CustomerNum         Name
-------------------------------
00000               Adams
00001               Blake

PRESS "ENTER" TO CONTINUE
```



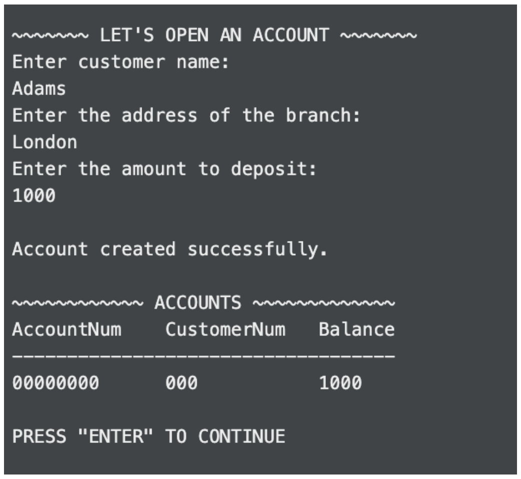
```
~~~~~~~~ NEW CUSTOMER ~~~~~~~~
Enter customer name:
Jones

Customer created successfully.

~~~~~~~~~ CUSTOMERS ~~~~~~~~~~
CustomerNum         Name
-------------------------------
00000               Adams
00001               Blake
00002               Henry
00003               Jones

PRESS "ENTER" TO CONTINUE
```

9.

10.



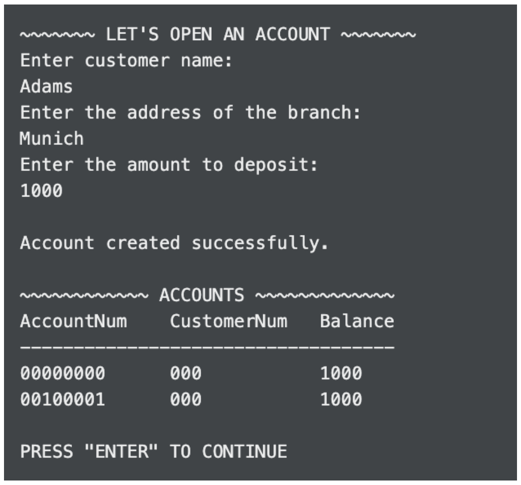
```
~~~~~~~~ NEW CUSTOMER ~~~~~~~~
Enter customer name:
Smith

Customer created successfully.

~~~~~~~~~ CUSTOMERS ~~~~~~~~~~
CustomerNum         Name
-------------------------------
00000               Adams
00001               Blake
00002               Henry
00003               Jones
00004               Smith

PRESS "ENTER" TO CONTINUE
```



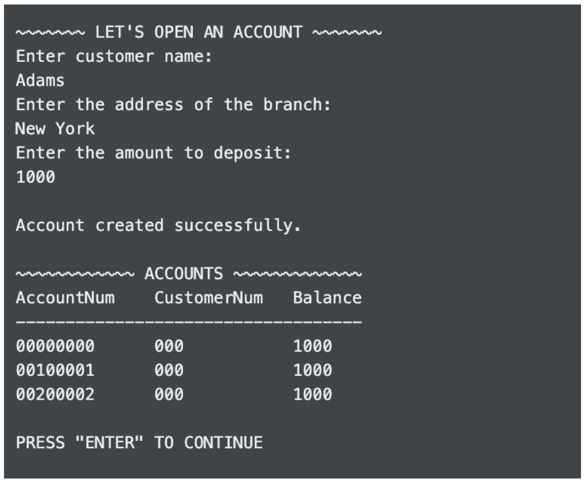
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Adams
Enter the address of the branch:
London
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
------------------------------------
00000000      000           1000

PRESS "ENTER" TO CONTINUE
```

11.

12.



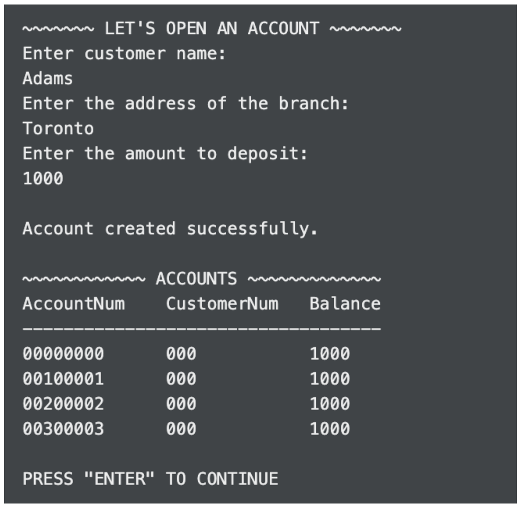
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Adams
Enter the address of the branch:
Munich
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000

PRESS "ENTER" TO CONTINUE
```



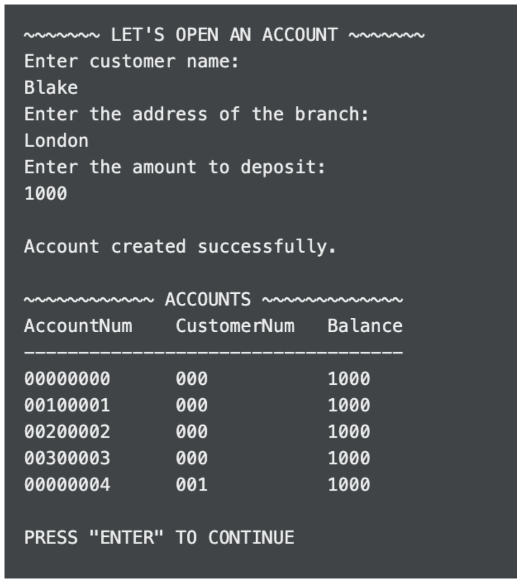
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Adams
Enter the address of the branch:
New York
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000

PRESS "ENTER" TO CONTINUE
```

13.

14.



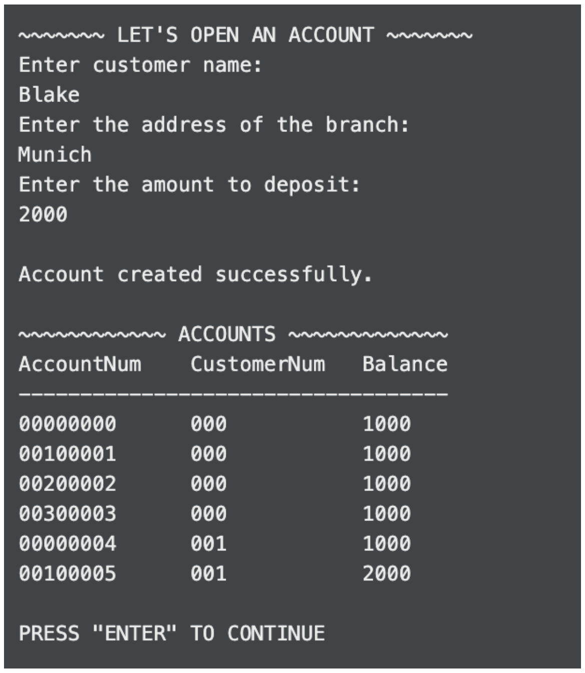
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Adams
Enter the address of the branch:
Toronto
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000

PRESS "ENTER" TO CONTINUE
```



```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Blake
Enter the address of the branch:
London
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000

PRESS "ENTER" TO CONTINUE
```

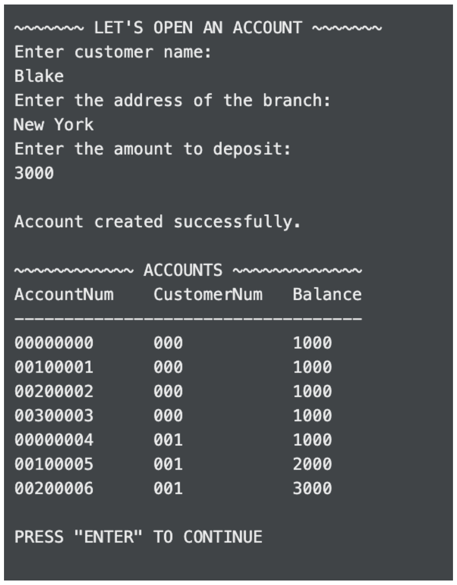
15.

16.



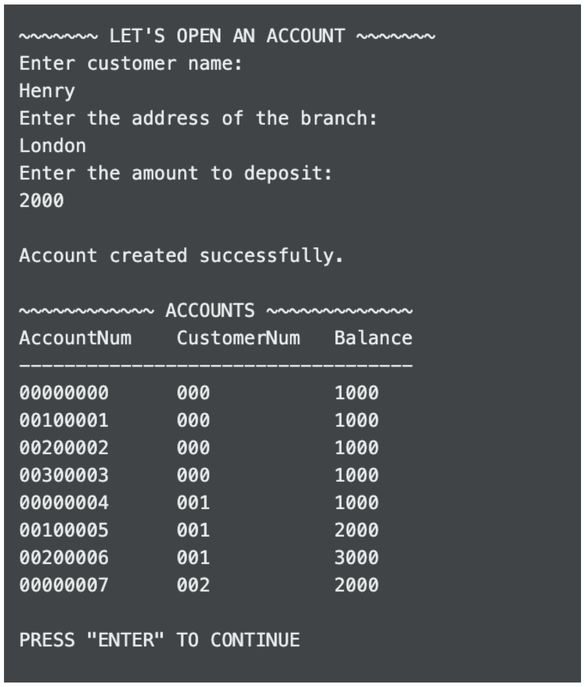
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Blake
Enter the address of the branch:
Munich
Enter the amount to deposit:
2000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000
00100005      001           2000

PRESS "ENTER" TO CONTINUE
```

17.



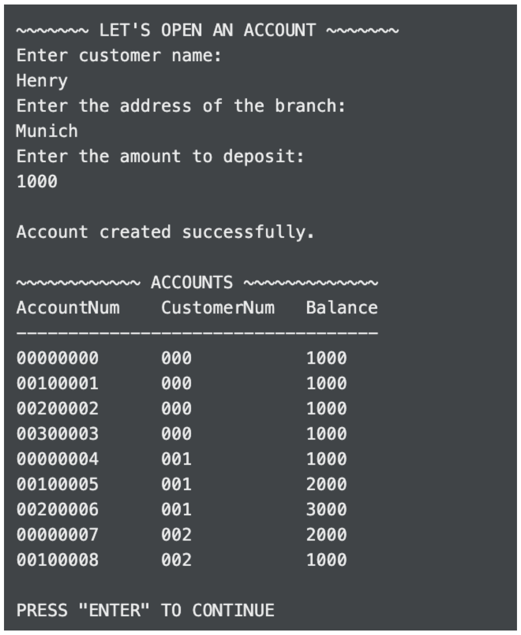
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Blake
Enter the address of the branch:
New York
Enter the amount to deposit:
3000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000
00100005      001           2000
00200006      001           3000

PRESS "ENTER" TO CONTINUE
```

18.



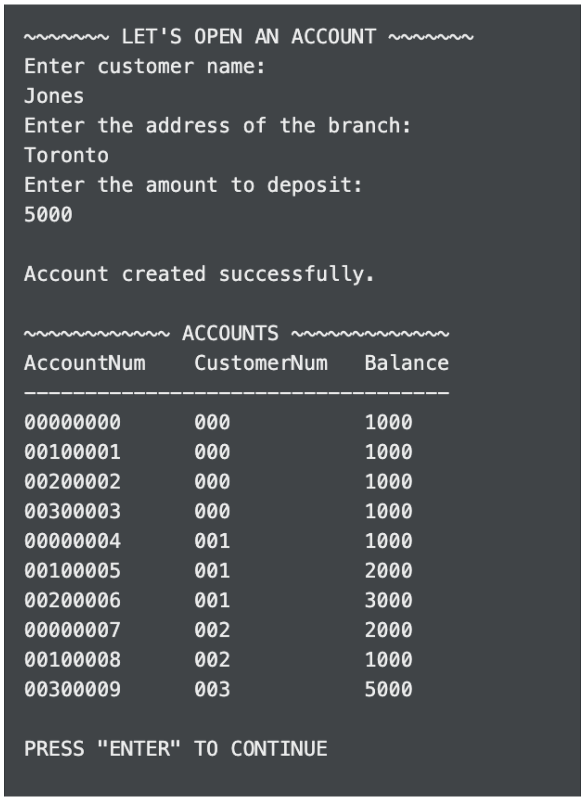
```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Henry
Enter the address of the branch:
London
Enter the amount to deposit:
2000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000
00100005      001           2000
00200006      001           3000
00000007      002           2000

PRESS "ENTER" TO CONTINUE
```

19.

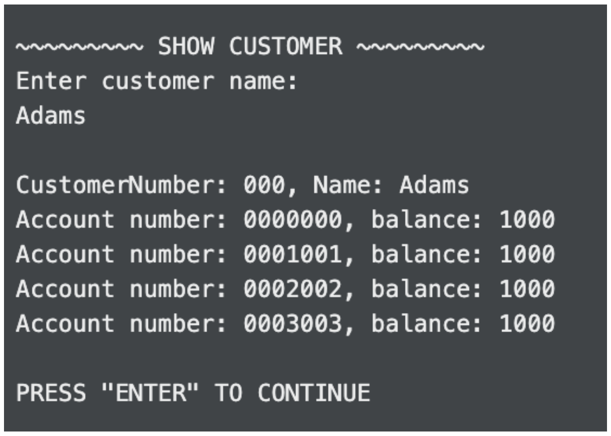


```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Henry
Enter the address of the branch:
Munich
Enter the amount to deposit:
1000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000
00100005      001           2000
00200006      001           3000
00000007      002           2000
00100008      002           1000

PRESS "ENTER" TO CONTINUE
```

20.

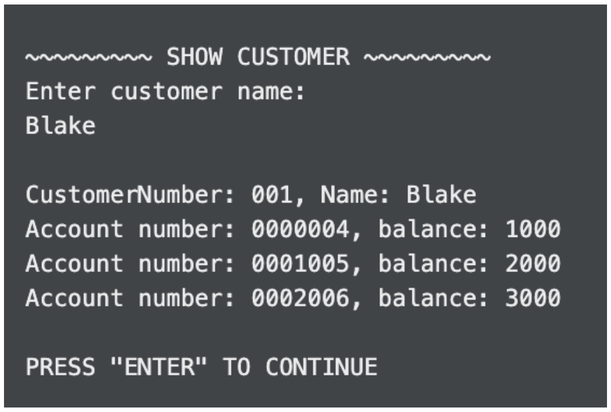


```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Jones
Enter the address of the branch:
Toronto
Enter the amount to deposit:
5000

Account created successfully.

~~~~~~~~~~~~ ACCOUNTS ~~~~~~~~~~~~~
AccountNum    CustomerNum   Balance
-----------------------------------
00000000      000           1000
00100001      000           1000
00200002      000           1000
00300003      000           1000
00000004      001           1000
00100005      001           2000
00200006      001           3000
00000007      002           2000
00100008      002           1000
00300009      003           5000

PRESS "ENTER" TO CONTINUE
```



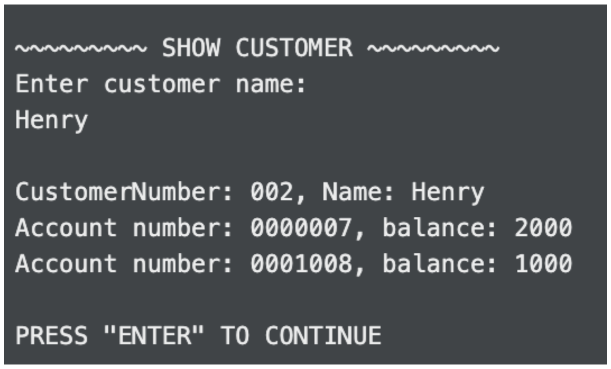
```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Adams

CustomerNumber: 000, Name: Adams
Account number: 0000000, balance: 1000
Account number: 0001001, balance: 1000
Account number: 0002002, balance: 1000
Account number: 0003003, balance: 1000

PRESS "ENTER" TO CONTINUE
```

21.



```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Blake

CustomerNumber: 001, Name: Blake
Account number: 0000004, balance: 1000
Account number: 0001005, balance: 2000
Account number: 0002006, balance: 3000

PRESS "ENTER" TO CONTINUE
```

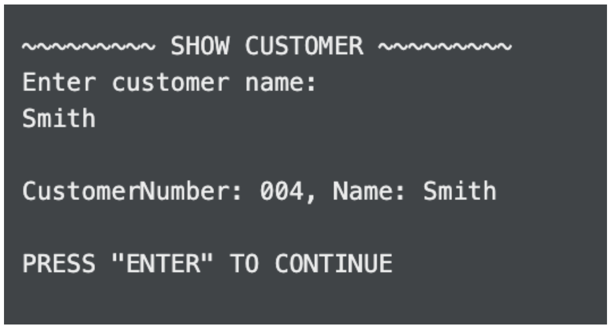
22.



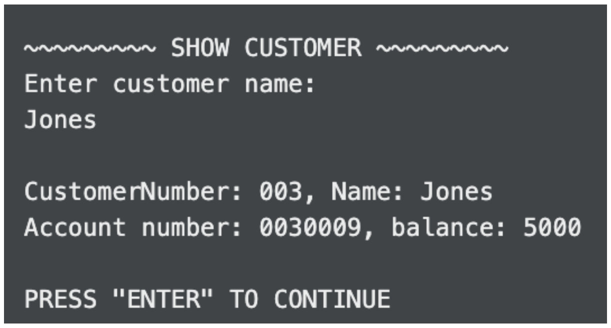
```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Henry

CustomerNumber: 002, Name: Henry
Account number: 0000007, balance: 2000
Account number: 0001008, balance: 1000

PRESS "ENTER" TO CONTINUE
```

23.

24.

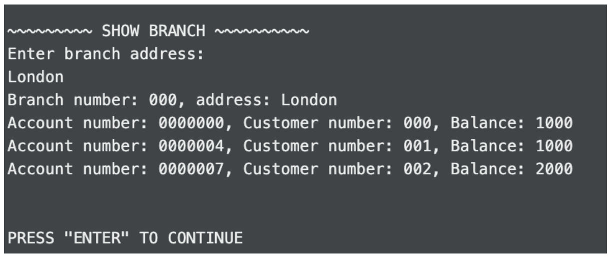


```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Smith

CustomerNumber: 004, Name: Smith

PRESS "ENTER" TO CONTINUE
```



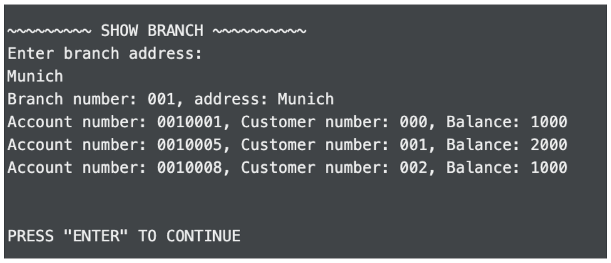
```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Jones

CustomerNumber: 003, Name: Jones
Account number: 0030009, balance: 5000

PRESS "ENTER" TO CONTINUE
```

25.



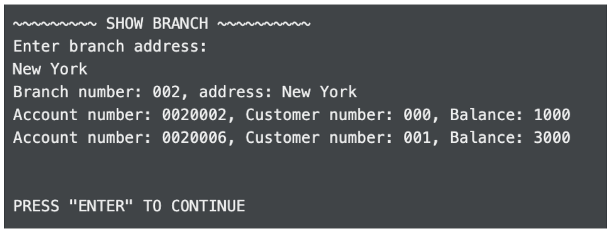
```
~~~~~~~~~ SHOW BRANCH ~~~~~~~~~~
Enter branch address:
London
Branch number: 000, address: London
Account number: 0000000, Customer number: 000, Balance: 1000
Account number: 0000004, Customer number: 001, Balance: 1000
Account number: 0000007, Customer number: 002, Balance: 2000


PRESS "ENTER" TO CONTINUE
```

26.



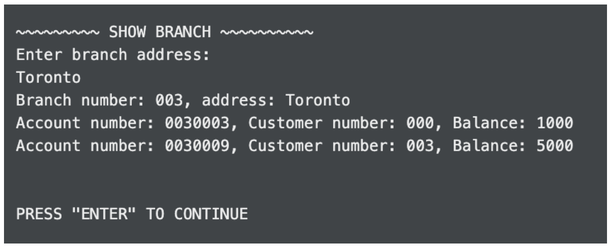
```
~~~~~~~~~ SHOW BRANCH ~~~~~~~~~~
Enter branch address:
Munich
Branch number: 001, address: Munich
Account number: 0010001, Customer number: 000, Balance: 1000
Account number: 0010005, Customer number: 001, Balance: 2000
Account number: 0010008, Customer number: 002, Balance: 1000


PRESS "ENTER" TO CONTINUE
```

27.



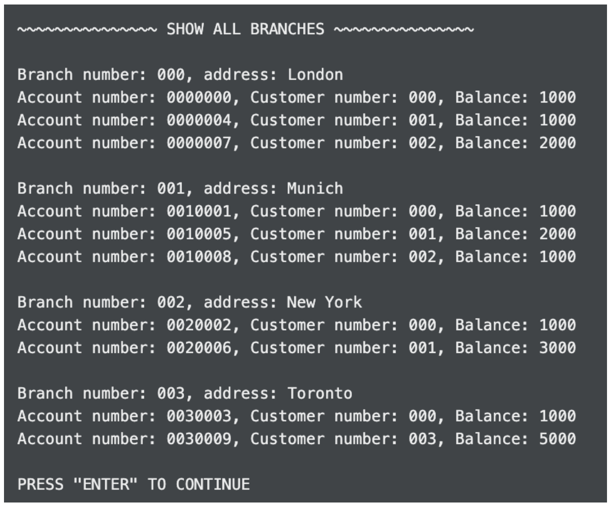
```
~~~~~~~~~ SHOW BRANCH ~~~~~~~~~~
Enter branch address:
New York
Branch number: 002, address: New York
Account number: 0020002, Customer number: 000, Balance: 1000
Account number: 0020006, Customer number: 001, Balance: 3000


PRESS "ENTER" TO CONTINUE
```

28.



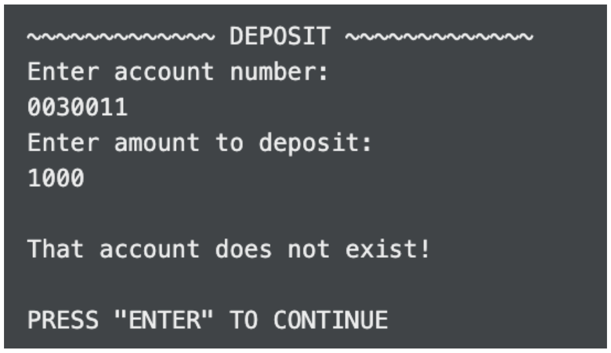
```
~~~~~~~~~ SHOW BRANCH ~~~~~~~~~~
Enter branch address:
Toronto
Branch number: 003, address: Toronto
Account number: 0030003, Customer number: 000, Balance: 1000
Account number: 0030009, Customer number: 003, Balance: 5000


PRESS "ENTER" TO CONTINUE
```

29.



```
~~~~~~~~~~~~~~~ SHOW ALL BRANCHES ~~~~~~~~~~~~~~~

Branch number: 000, address: London
Account number: 0000000, Customer number: 000, Balance: 1000
Account number: 0000004, Customer number: 001, Balance: 1000
Account number: 0000007, Customer number: 002, Balance: 2000

Branch number: 001, address: Munich
Account number: 0010001, Customer number: 000, Balance: 1000
Account number: 0010005, Customer number: 001, Balance: 2000
Account number: 0010008, Customer number: 002, Balance: 1000

Branch number: 002, address: New York
Account number: 0020002, Customer number: 000, Balance: 1000
Account number: 0020006, Customer number: 001, Balance: 3000

Branch number: 003, address: Toronto
Account number: 0030003, Customer number: 000, Balance: 1000
Account number: 0030009, Customer number: 003, Balance: 5000

PRESS "ENTER" TO CONTINUE
```

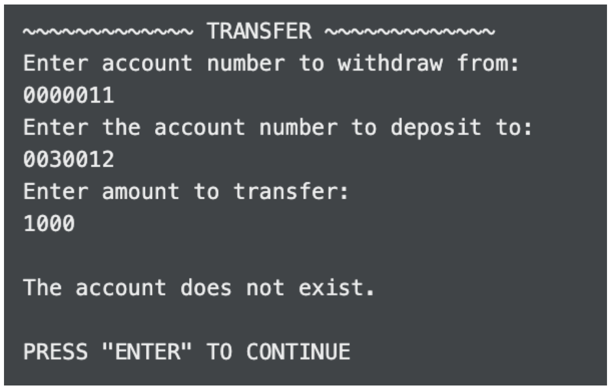
30.



```
~~~~~~~~~~~~~~ DEPOSIT ~~~~~~~~~~~~~~
Enter account number:
0030011
Enter amount to deposit:
1000

That account does not exist!

PRESS "ENTER" TO CONTINUE
```

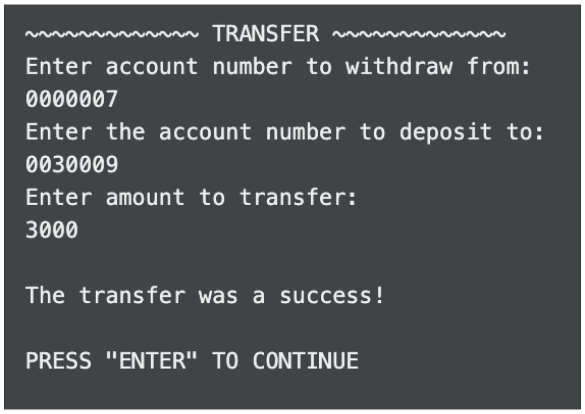
32.

31.



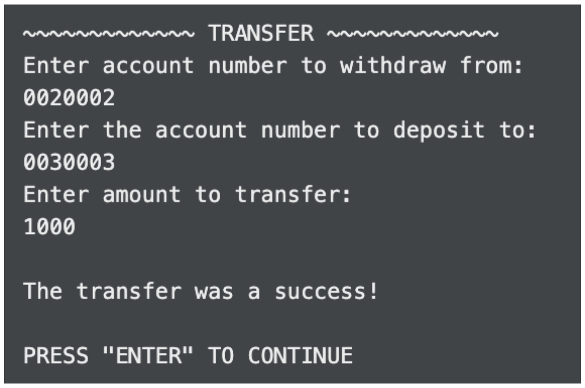
```
~~~~~~~~~~~~~ TRANSFER ~~~~~~~~~~~~~
Enter account number to withdraw from:
0000011
Enter the account number to deposit to:
0030012
Enter amount to transfer:
1000

The account does not exist.

PRESS "ENTER" TO CONTINUE
```

33.



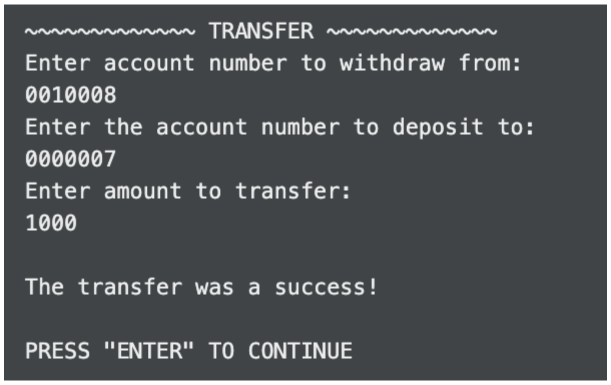
```
~~~~~~~~~~~~~ TRANSFER ~~~~~~~~~~~~~
Enter account number to withdraw from:
0000007
Enter the account number to deposit to:
0030009
Enter amount to transfer:
3000

The transfer was a success!

PRESS "ENTER" TO CONTINUE
```

35.



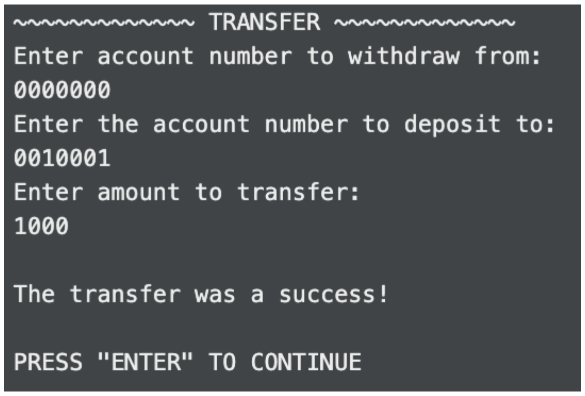
```
~~~~~~~~~~~~~ TRANSFER ~~~~~~~~~~~~~
Enter account number to withdraw from:
0020002
Enter the account number to deposit to:
0030003
Enter amount to transfer:
1000

The transfer was a success!

PRESS "ENTER" TO CONTINUE
```

36.



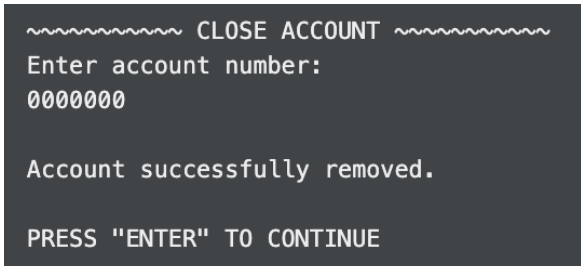
```
~~~~~~~~~~~~~ TRANSFER ~~~~~~~~~~~~~
Enter account number to withdraw from:
0010008
Enter the account number to deposit to:
0000007
Enter amount to transfer:
1000

The transfer was a success!

PRESS "ENTER" TO CONTINUE
```

34.

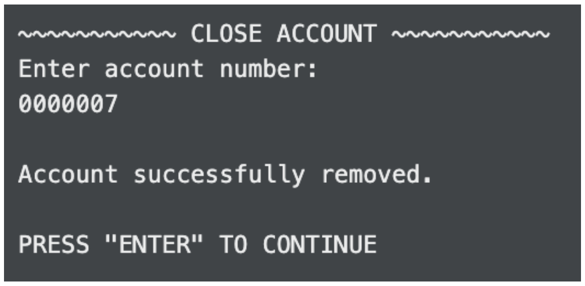


```
~~~~~~~~~~~~~ TRANSFER ~~~~~~~~~~~~~
Enter account number to withdraw from:
0000000
Enter the account number to deposit to:
0010001
Enter amount to transfer:
1000

The transfer was a success!

PRESS "ENTER" TO CONTINUE
```

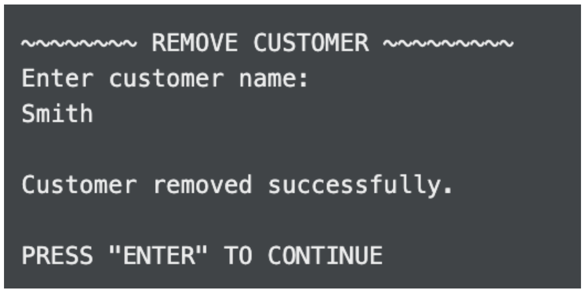
37.



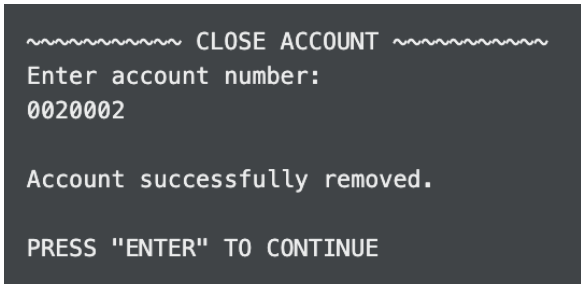
```
~~~~~~~~~~~ CLOSE ACCOUNT ~~~~~~~~~~~
Enter account number:
0000000

Account successfully removed.

PRESS "ENTER" TO CONTINUE
```



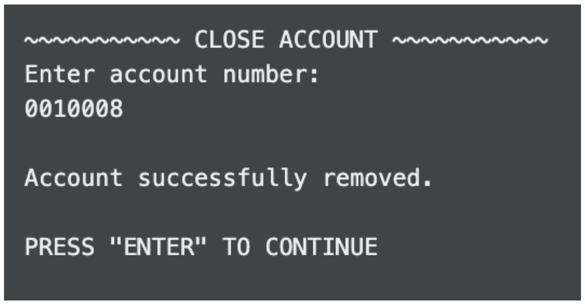
```
~~~~~~~~~~~ CLOSE ACCOUNT ~~~~~~~~~~~
Enter account number:
0000007

Account successfully removed.

PRESS "ENTER" TO CONTINUE
```



```
~~~~~~~~~ REMOVE CUSTOMER ~~~~~~~~~
Enter customer name:
Smith

Customer removed successfully.

PRESS "ENTER" TO CONTINUE
```

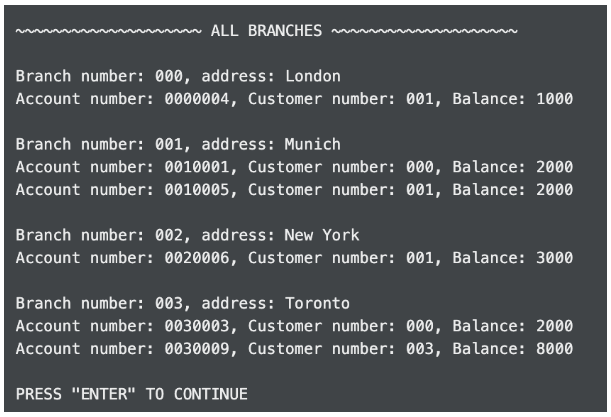


```
~~~~~~~~~~~ CLOSE ACCOUNT ~~~~~~~~~~~
Enter account number:
0020002

Account successfully removed.

PRESS "ENTER" TO CONTINUE
```



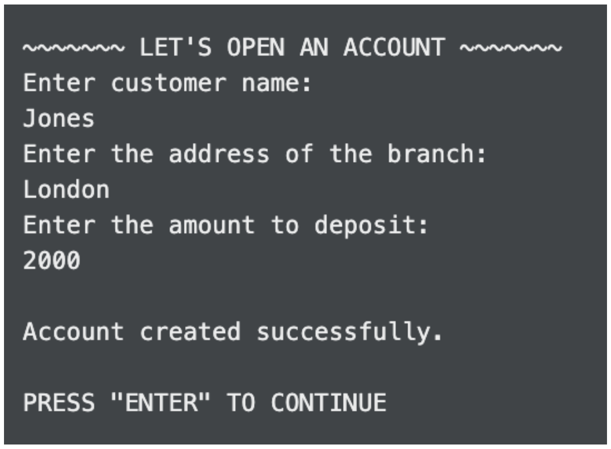
```
~~~~~~~~~~~ CLOSE ACCOUNT ~~~~~~~~~~~
Enter account number:
0010008

Account successfully removed.

PRESS "ENTER" TO CONTINUE
```

38.



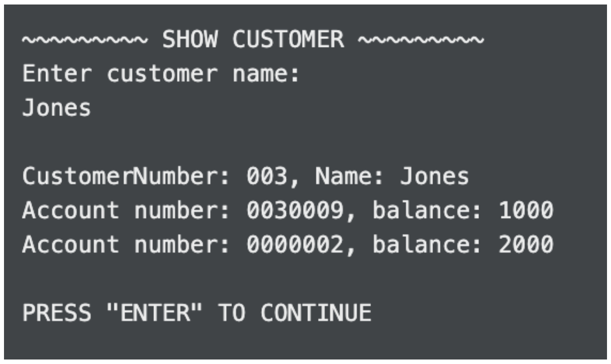
```
~~~~~~~~~~~~~~~~~~~~ ALL BRANCHES ~~~~~~~~~~~~~~~~~~~~

Branch number: 000, address: London
Account number: 0000004, Customer number: 001, Balance: 1000

Branch number: 001, address: Munich
Account number: 0010001, Customer number: 000, Balance: 2000
Account number: 0010005, Customer number: 001, Balance: 2000

Branch number: 002, address: New York
Account number: 0020006, Customer number: 001, Balance: 3000

Branch number: 003, address: Toronto
Account number: 0030003, Customer number: 000, Balance: 2000
Account number: 0030009, Customer number: 003, Balance: 8000

PRESS "ENTER" TO CONTINUE
```

39.



```
~~~~~~~ LET'S OPEN AN ACCOUNT ~~~~~~~
Enter customer name:
Jones
Enter the address of the branch:
London
Enter the amount to deposit:
2000

Account created successfully.

PRESS "ENTER" TO CONTINUE
```

40.



```
~~~~~~~~~ SHOW CUSTOMER ~~~~~~~~~
Enter customer name:
Jones

CustomerNumber: 003, Name: Jones
Account number: 0030009, balance: 1000
Account number: 0000002, balance: 2000

PRESS "ENTER" TO CONTINUE
```