BANKING SYSTEM-OOPS,COLLECTIONS AND EXCEPTION HANDLING

TASK 10: HAS A RELATION / ASSOCIATION:

Customer.java:

```
import java.util.regex.Pattern;

public class Customer {
    private String customerID;
    private String firstName;
    private String lastName;
    private String emailAddress;
    private String phoneNumber;
    private String address;

    public Customer() {
    }

    public Customer(String customerID, String firstName, String lastName, String
emailAddress,
                    String phoneNumber, String address) {
        this.customerID = customerID;
        this.firstName = firstName;
        this.lastName = lastName;
        setEmailAddress(emailAddress);
        setPhoneNumber(phoneNumber);
        this.address = address;
    }


    public String getCustomerID() { return customerID; }
    public void setCustomerID(String customerID) { this.customerID = customerID; }

    public String getFirstName() { return firstName; }
    public void setFirstName(String firstName) { this.firstName = firstName; }

    public String getLastName() { return lastName; }
    public void setLastName(String lastName) { this.lastName = lastName; }

    public String getEmailAddress() { return emailAddress; }
    public void setEmailAddress(String emailAddress) {
        if (Pattern.matches("^[\\w.-]+@[\\w.-]+\\.[a-zA-Z]{2,6}$", emailAddress)) {
            this.emailAddress = emailAddress;
        } else {
            throw new IllegalArgumentException("Invalid email address.");
        }
    }

    public String getPhoneNumber() { return phoneNumber; }
```

```
    public void setPhoneNumber(String phoneNumber) {
        if (Pattern.matches("\\d{10}", phoneNumber)) {
            this.phoneNumber = phoneNumber;
        } else {
            throw new IllegalArgumentException("Phone number must be 10 digits.");
        }
    }

    public String getAddress() { return address; }
    public void setAddress(String address) { this.address = address; }

    public void printCustomerInfo() {
        System.out.println("Customer ID: " + customerID);
        System.out.println("Name: " + firstName + " " + lastName);
        System.out.println("Email: " + emailAddress);
        System.out.println("Phone: " + phoneNumber);
        System.out.println("Address: " + address);
    }
}
```

Account.java

```
public class Account {
    private long accountNumber;
    private String accountType;
    private float accountBalance;
    private Customer customer;

    public Account() {
    }

    public Account(long accountNumber, String accountType, float accountBalance, Customer
customer) {
        this.accountNumber = accountNumber;
        this.accountType = accountType;
        this.accountBalance = accountBalance;
        this.customer = customer;
    }

    public long getAccountNumber() { return accountNumber; }
    public void setAccountNumber(long accountNumber) { this.accountNumber =
accountNumber; }

    public String getAccountType() { return accountType; }
    public void setAccountType(String accountType) { this.accountType = accountType; }

    public float getAccountBalance() { return accountBalance; }
    public void setAccountBalance(float accountBalance) { this.accountBalance =
accountBalance; }
```

```
    public Customer getCustomer() { return customer; }
    public void setCustomer(Customer customer) { this.customer = customer; }

    public void printAccountInfo() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Type: " + accountType);
        System.out.println("Balance: " + accountBalance);
        customer.printCustomerInfo();
    }
}
```

Bank.java

```
import java.util.HashMap;
import java.util.Map;

public class Bank {
    private Map<Long, Account> accounts = new HashMap<>();
    private long nextAccountNumber = 1001;

    public long createAccount(Customer customer, String accType, float balance) {
        long accNo = nextAccountNumber++;
        Account account = new Account(accNo, accType, balance, customer);
        accounts.put(accNo, account);
        System.out.println("Account created successfully! Account Number: " + accNo);
        return accNo;
    }

    public float getAccountBalance(long accNo) {
        Account acc = accounts.get(accNo);
        if (acc != null) return acc.getAccountBalance();
        else throw new IllegalArgumentException("Account not found.");
    }

    public float deposit(long accNo, float amount) {
        Account acc = accounts.get(accNo);
        if (acc != null) {
            acc.setAccountBalance(acc.getAccountBalance() + amount);
            return acc.getAccountBalance();
        } else throw new IllegalArgumentException("Account not found.");
    }

    public float withdraw(long accNo, float amount) {
        Account acc = accounts.get(accNo);
        if (acc != null) {
            if (acc.getAccountBalance() >= amount) {
                acc.setAccountBalance(acc.getAccountBalance() - amount);
                return acc.getAccountBalance();
            } else throw new IllegalArgumentException("Insufficient balance.");
        } else throw new IllegalArgumentException("Account not found.");
```

```
    }

    public void transfer(long fromAcc, long toAcc, float amount) {
        withdraw(fromAcc, amount);
        deposit(toAcc, amount);
        System.out.println("Transfer successful from " + fromAcc + " to " + toAcc);
    }

    public void getAccountDetails(long accNo) {
        Account acc = accounts.get(accNo);
        if (acc != null) acc.printAccountInfo();
        else throw new IllegalArgumentException("Account not found.");
    }
}
```

BankApp.java:

```
import java.util.Scanner;

public class BankApp {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Bank bank = new Bank();

        while (true) {
            System.out.println("\n------ BANK SYSTEM MENU ------");
            System.out.println("1. Create Account");
            System.out.println("2. Deposit");
            System.out.println("3. Withdraw");
            System.out.println("4. Get Balance");
            System.out.println("5. Transfer");
            System.out.println("6. Get Account Details");
            System.out.println("7. Exit");
            System.out.print("Enter your choice: ");
            int ch = sc.nextInt();
            sc.nextLine(); // Consume newline

            try {
                switch (ch) {
                    case 1:
                        System.out.print("Enter Customer ID: ");
                        String cid = sc.nextLine();
                        System.out.print("First Name: ");
                        String fname = sc.nextLine();
                        System.out.print("Last Name: ");
                        String lname = sc.nextLine();
                        System.out.print("Email: ");
                        String email = sc.nextLine();
                        System.out.print("Phone: ");
                        String phone = sc.nextLine();
```

```
                System.out.print("Address: ");
                String address = sc.nextLine();

                Customer customer = new Customer(cid, fname, lname, email, phone,
address);

                System.out.println("Choose Account Type:");
                System.out.println("1. Savings");
                System.out.println("2. Current");
                String accType = sc.nextInt() == 1 ? "Savings" : "Current";

                System.out.print("Initial Balance: ");
                float balance = sc.nextFloat();

                bank.createAccount(customer, accType, balance);
                break;

            case 2:
                System.out.print("Account Number: ");
                long accNo = sc.nextLong();
                System.out.print("Amount to deposit: ");
                float dep = sc.nextFloat();
                float newBalance = bank.deposit(accNo, dep);
                System.out.println("New Balance: " + newBalance);
                break;

            case 3:
                System.out.print("Account Number: ");
                long wacc = sc.nextLong();
                System.out.print("Amount to withdraw: ");
                float with = sc.nextFloat();
                float wbalance = bank.withdraw(wacc, with);
                System.out.println("New Balance: " + wbalance);
                break;

            case 4:
                System.out.print("Account Number: ");
                long gbacc = sc.nextLong();
                float gbalance = bank.getAccountBalance(gbacc);
                System.out.println("Balance: " + gbalance);
                break;

            case 5:
                System.out.print("From Account Number: ");
                long from = sc.nextLong();
                System.out.print("To Account Number: ");
                long to = sc.nextLong();
                System.out.print("Amount to transfer: ");
                float amt = sc.nextFloat();
                bank.transfer(from, to, amt);
```

```
                break;

            case 6:
                System.out.print("Account Number: ");
                long dacc = sc.nextLong();
                bank.getAccountDetails(dacc);
                break;

            case 7:
                System.out.println("Exiting... Thank you!");
                return;

            default:
                System.out.println("Invalid choice.");
          }
        } catch (Exception e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
  }
}
```

OUTPUT:

```
------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 1
Enter Customer ID: 1
First Name: Praveshini
Last Name: B N V
Email: bnvpraveshini@gmail.com
Phone: 9994650515
Address: Madurai
Choose Account Type:
1. Savings
2. Current
1
Initial Balance: 5000
Account created successfully! Account Number: 1001
```

```
------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 2
Account Number: 1001
Amount to deposit: 2000
New Balance: □7000.0

------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 3
Account Number: 1001
Amount to withdraw: 1000
New Balance: □6000.0

------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 4
Account Number: 1001
Balance: □6000.0

------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 6
Account Number: 1001
Account Number: 1001
Type: Savings
Balance: □6000.0
```

```
Customer ID: 1
Name: Praveshini B N V
Email: bnvpraveshini@gmail.com
Phone: 9994650515
Address: Madurai

------ BANK SYSTEM MENU ------
1. Create Account
2. Deposit
3. Withdraw
4. Get Balance
5. Transfer
6. Get Account Details
7. Exit
Enter your choice: 7
Exiting... Thank you!
BUILD SUCCESSFUL (total time: 3 minutes 1 second)
```