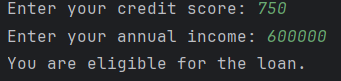
Java Assignment

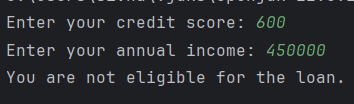
Sivaganesh Natarajavel

Task 1:

|  |
| --- |
| import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   System.*out*.print("Enter your credit score: ");  int creditScore = scanner.nextInt();   System.*out*.print("Enter your annual income: ");  double income = scanner.nextDouble();   if (creditScore > 700 && income >= 50000) {  System.*out*.println("You are eligible for the loan.");  } else {  System.*out*.println("You are not eligible for the loan.");  }  scanner.close();  } } |

Output:

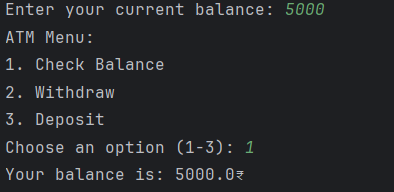


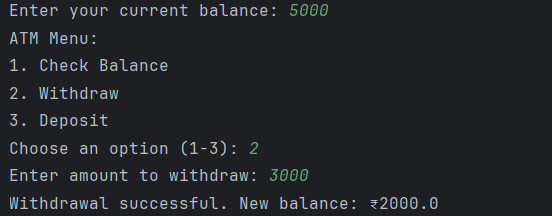


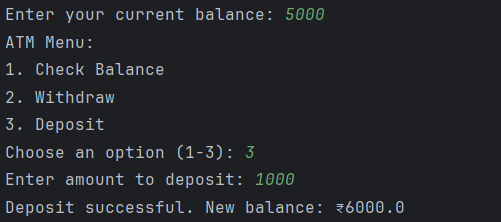
Task 2 :

|  |
| --- |
| package org.example; import java.util.Scanner; public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   double balance;  int option;   System.*out*.print("Enter your current balance: ");  balance = scanner.nextDouble();   System.*out*.println("ATM Menu:");  System.*out*.println("1. Check Balance");  System.*out*.println("2. Withdraw");  System.*out*.println("3. Deposit");  System.*out*.print("Choose an option (1-3): ");  option = scanner.nextInt();   if (option == 1) {  System.*out*.println("Your balance is: " + balance +"₹");  } else if (option == 2) {  System.*out*.print("Enter amount to withdraw: ");  double withdrawAmount = scanner.nextDouble();   if (withdrawAmount <= balance) {  if (withdrawAmount % 100 == 0 || withdrawAmount % 500 == 0) {  balance -= withdrawAmount;  System.*out*.println("Withdrawal successful. New balance: ₹" + balance);  } else {  System.*out*.println("Amount must be in multiples of 100 or 500.");  }  } else {  System.*out*.println("Insufficient balance.");  }  } else if (option == 3) {  System.*out*.print("Enter amount to deposit: ");  double depositAmount = scanner.nextDouble();  balance += depositAmount;  System.*out*.println("Deposit successful. New balance: ₹" + balance);  } else {  System.*out*.println("Invalid option.");  }   scanner.close();  } } |

Output:



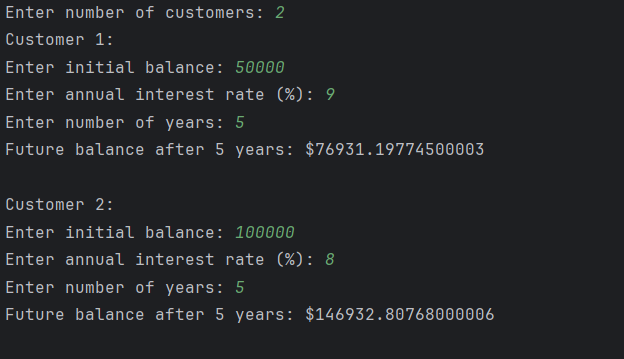




Task 3:

|  |
| --- |
| package org.example; import java.util.Scanner; public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   System.*out*.print("Enter number of customers: ");  int customers = scanner.nextInt();   for (int i = 1; i <= customers; i++) {  System.*out*.println("Customer " + i + ":");   System.*out*.print("Enter initial balance: ");  double initialBalance = scanner.nextDouble();   System.*out*.print("Enter annual interest rate (%): ");  double interestRate = scanner.nextDouble();   System.*out*.print("Enter number of years: ");  int years = scanner.nextInt();   double futureBalance = initialBalance \* Math.*pow*(1 + (interestRate / 100), years);   System.*out*.println("Future balance after " + years + " years: $" + futureBalance);  System.*out*.println();  }  scanner.close();  } } |

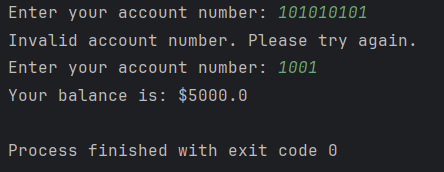
Output:



Task 4:

|  |
| --- |
| package org.example; import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   int[] accountNumbers = {1001, 1002, 1003, 1004, 1005};  double[] balances = {5000.0, 15000.0, 8000.5, 12000.75, 2000.0};   boolean valid = false;   while (!valid) {  System.*out*.print("Enter your account number: ");  int enteredAccNo = scanner.nextInt();   boolean found = false;   for (int i = 0; i < accountNumbers.length; i++) {  if (accountNumbers[i] == enteredAccNo) {  System.*out*.println("Your balance is: $" + balances[i]);  found = true;  valid = true;  break;  }  }   if (!found) {  System.*out*.println("Invalid account number. Please try again.");  }  }   scanner.close();  } } |

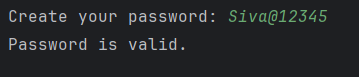
Output:



Task 5:

|  |
| --- |
| package org.example; import java.util.Scanner; public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   System.*out*.print("Create your password: ");  String password = scanner.nextLine();   boolean hasUpper = false;  boolean hasDigit = false;   if (password.length() >= 8) {  for (int i = 0; i < password.length(); i++) {  char ch = password.charAt(i);  if (Character.*isUpperCase*(ch)) {  hasUpper = true;  }  if (Character.*isDigit*(ch)) {  hasDigit = true;  }  }   if (hasUpper && hasDigit) {  System.*out*.println("Password is valid.");  } else {  System.*out*.println("Password must contain at least one uppercase letter and one digit.");  }  } else {  System.*out*.println("Password must be at least 8 characters long.");  }   scanner.close();  } } |

Output:

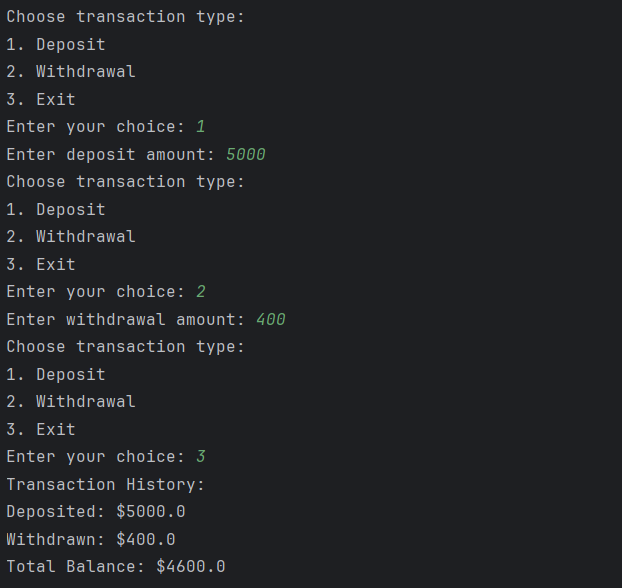




Task 6:

|  |
| --- |
| package org.example; import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   String[] transactions = new String[100];  int index = 0;  double balance = 0;  boolean running = true;   while (running) {  System.*out*.println("Choose transaction type:");  System.*out*.println("1. Deposit");  System.*out*.println("2. Withdrawal");  System.*out*.println("3. Exit");  System.*out*.print("Enter your choice: ");  int choice = scanner.nextInt();   if (choice == 1) {  System.*out*.print("Enter deposit amount: ");  double amount = scanner.nextDouble();  balance += amount;  transactions[index] = "Deposited: $" + amount;  index++;  } else if (choice == 2) {  System.*out*.print("Enter withdrawal amount: ");  double amount = scanner.nextDouble();  balance -= amount;  transactions[index] = "Withdrawn: $" + amount;  index++;  } else if (choice == 3) {  running = false;  } else {  System.*out*.println("Invalid option.");  }  }   System.*out*.println("Transaction History:");  for (int i = 0; i < index; i++) {  System.*out*.println(transactions[i]);  }   System.*out*.println("Total Balance: $" + balance);   scanner.close();  } } |

Output :



Task 7:

Main.java

|  |
| --- |
| package org.example;  public class Main {  public static void main(String[] args) {   Customer customer = new Customer("C123", "John", "Doe", "john.doe@example.com", "1234567890", "123 Elm Street");  customer.printCustomerInfo();    Account account = new Account("A12345", "Savings", 1000.0);  account.printAccountInfo();   account.deposit(500.0);  account.withdraw(200.0);  account.calculateInterest();   account.printAccountInfo();  } } |

Customer.java

|  |
| --- |
| package org.example; class Customer {   private String customerId;  private String firstName;  private String lastName;  private String emailAddress;  private String phoneNumber;  private String address;   // Default constructor  public Customer() {  }   // Overloaded constructor with attributes  public Customer(String customerId, String firstName, String lastName, String emailAddress, String phoneNumber, String address) {  this.customerId = customerId;  this.firstName = firstName;  this.lastName = lastName;  this.emailAddress = emailAddress;  this.phoneNumber = phoneNumber;  this.address = address;  }   // Getters and Setters  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) {  this.emailAddress = emailAddress;  }   public String getPhoneNumber() {  return phoneNumber;  }   public void setPhoneNumber(String phoneNumber) {  this.phoneNumber = phoneNumber;  }   public String getAddress() {  return address;  }   public void setAddress(String address) {  this.address = address;  }   // Method to print all information  public void printCustomerInfo() {  System.*out*.println("Customer ID: " + customerId);  System.*out*.println("First Name: " + firstName);  System.*out*.println("Last Name: " + lastName);  System.*out*.println("Email Address: " + emailAddress);  System.*out*.println("Phone Number: " + phoneNumber);  System.*out*.println("Address: " + address);  } } |

Account.java:

|  |
| --- |
| package org.example; class Account {  private String accountNumber;  private String accountType; // "Savings" or "Current"  private double accountBalance;   // Default constructor  public Account() {  }   // Overloaded constructor with account attributes  public Account(String accountNumber, String accountType, double accountBalance) {  this.accountNumber = accountNumber;  this.accountType = accountType;  this.accountBalance = accountBalance;  }   // Getters and Setters  public String getAccountNumber() {  return accountNumber;  }   public void setAccountNumber(String accountNumber) {  this.accountNumber = accountNumber;  }   public String getAccountType() {  return accountType;  }   public void setAccountType(String accountType) {  this.accountType = accountType;  }   public double getAccountBalance() {  return accountBalance;  }   public void setAccountBalance(double accountBalance) {  this.accountBalance = accountBalance;  }   // Method to print all information  public void printAccountInfo() {  System.*out*.println("Account Number: " + accountNumber);  System.*out*.println("Account Type: " + accountType);  System.*out*.println("Account Balance: " + accountBalance);  }   // Deposit method  public void deposit(double amount) {  if (amount > 0) {  accountBalance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   // Withdraw method  public void withdraw(double amount) {  if (amount > 0 && amount <= accountBalance) {  accountBalance -= amount;  System.*out*.println("Withdrew: " + amount);  } else if (amount > accountBalance) {  System.*out*.println("Insufficient balance.");  } else {  System.*out*.println("Invalid withdrawal amount.");  }  }   // Method to calculate interest  public void calculateInterest() {  if (accountType.equals("Savings")) {  double interest = accountBalance \* 0.045; // 4.5% interest rate  accountBalance += interest;  System.*out*.println("Interest added: " + interest);  } else {  System.*out*.println("Interest not applicable for Current account.");  }  } } |

Task 8:

Bank.java

|  |
| --- |
| package org.example;  import java.util.Scanner;  public class Bank {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);  System.*out*.println("Welcome to the SG Bank!");  System.*out*.println("Choose Account Type: ");  System.*out*.println("1. SavingsAccount");  System.*out*.println("2. CurrentAccount");  System.*out*.print("Enter choice: ");  int choice = scanner.nextInt();   Account account = null;   switch (choice) {  case 1:  System.*out*.print("Enter Account Number: ");  String savingsAccountNumber = scanner.next();  System.*out*.print("Enter Account Balance: ");  double savingsBalance = scanner.nextDouble();  System.*out*.print("Enter Interest Rate: ");  double interestRate = scanner.nextDouble();  account = new SavingsAccount(savingsAccountNumber, "Savings", savingsBalance, interestRate);  break;  case 2:  System.*out*.print("Enter Account Number: ");  String currentAccountNumber = scanner.next();  System.*out*.print("Enter Account Balance: ");  double currentBalance = scanner.nextDouble();  account = new CurrentAccount(currentAccountNumber, "Current", currentBalance);  break;  default:  System.*out*.println("Invalid choice.");  return;  }   System.*out*.println("\nAccount Created Successfully!");  account.printAccountInfo();   System.*out*.println("\nDeposit and Withdraw Operations:");  System.*out*.print("Enter amount to deposit: ");  double depositAmount = scanner.nextDouble();  account.deposit(depositAmount);   System.*out*.print("Enter amount to withdraw: ");  double withdrawAmount = scanner.nextDouble();  account.withdraw(withdrawAmount);   account.calculateInterest();  account.printAccountInfo();  } } |

Account.java

|  |
| --- |
| package org.example;  public class Account {  private String accountNumber;  private String accountType;  private double accountBalance;   public Account() {}   public Account(String accountNumber, String accountType, double accountBalance) {  this.accountNumber = accountNumber;  this.accountType = accountType;  this.accountBalance = accountBalance;  }   public String getAccountNumber() {  return accountNumber;  }   public void setAccountNumber(String accountNumber) {  this.accountNumber = accountNumber;  }   public String getAccountType() {  return accountType;  }   public void setAccountType(String accountType) {  this.accountType = accountType;  }   public double getAccountBalance() {  return accountBalance;  }   public void setAccountBalance(double accountBalance) {  this.accountBalance = accountBalance;  }   public void deposit(float amount) {  if (amount > 0) {  accountBalance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   public void withdraw(float amount) {  if (amount > 0 && amount <= accountBalance) {  accountBalance -= amount;  System.*out*.println("Withdrew: " + amount);  } else if (amount > accountBalance) {  System.*out*.println("Insufficient balance.");  } else {  System.*out*.println("Invalid withdrawal amount.");  }  }   public void deposit(int amount) {  if (amount > 0) {  accountBalance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   public void withdraw(int amount) {  if (amount > 0 && amount <= accountBalance) {  accountBalance -= amount;  System.*out*.println("Withdrew: " + amount);  } else if (amount > accountBalance) {  System.*out*.println("Insufficient balance.");  } else {  System.*out*.println("Invalid withdrawal amount.");  }  }   public void deposit(double amount) {  if (amount > 0) {  accountBalance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   public void withdraw(double amount) {  if (amount > 0 && amount <= accountBalance) {  accountBalance -= amount;  System.*out*.println("Withdrew: " + amount);  } else if (amount > accountBalance) {  System.*out*.println("Insufficient balance.");  } else {  System.*out*.println("Invalid withdrawal amount.");  }  }   public void calculateInterest() {  System.*out*.println("Interest calculation not applicable.");  }   public void printAccountInfo() {  System.*out*.println("Account Number: " + accountNumber);  System.*out*.println("Account Type: " + accountType);  System.*out*.println("Account Balance: " + accountBalance);  } } |

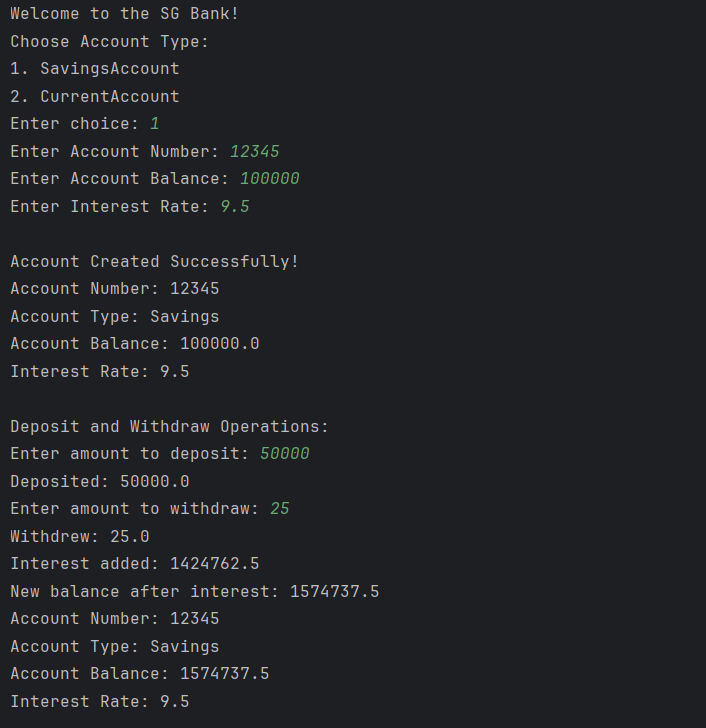
CurrentAccount.java

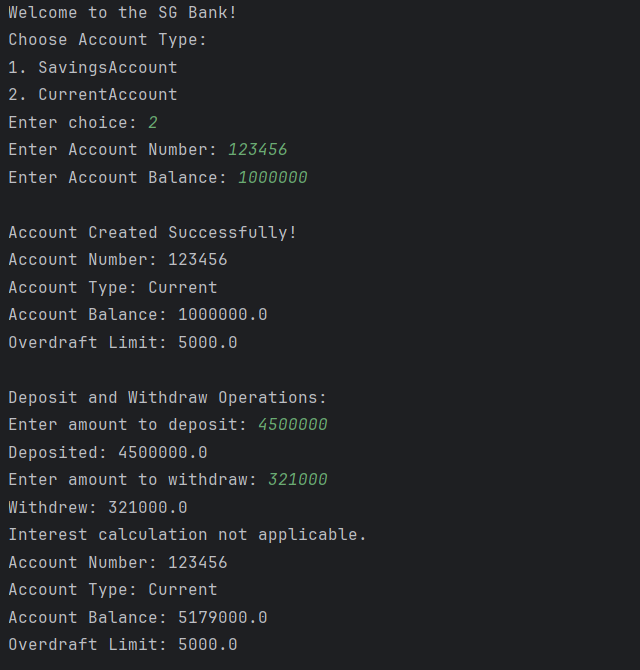
|  |
| --- |
| package org.example;  public class CurrentAccount extends Account {  private static final double *OVERDRAFT\_LIMIT* = 5000.0;   public CurrentAccount(String accountNumber, String accountType, double accountBalance) {  super(accountNumber, accountType, accountBalance);  }   @Override  public void withdraw(double amount) {  if (amount > 0 && (getAccountBalance() + *OVERDRAFT\_LIMIT*) >= amount) {  setAccountBalance(getAccountBalance() - amount);  System.*out*.println("Withdrew: " + amount);  } else if (amount > (getAccountBalance() + *OVERDRAFT\_LIMIT*)) {  System.*out*.println("Insufficient funds and overdraft limit exceeded.");  } else {  System.*out*.println("Invalid withdrawal amount.");  }  }   public void printAccountInfo() {  super.printAccountInfo();  System.*out*.println("Overdraft Limit: " + *OVERDRAFT\_LIMIT*);  } } |

SavingsAccount.java

|  |
| --- |
| package org.example;  public class SavingsAccount extends Account {  private double interestRate;   public SavingsAccount(String accountNumber, String accountType, double accountBalance, double interestRate) {  super(accountNumber, accountType, accountBalance);  this.interestRate = interestRate;  }   @Override  public void calculateInterest() {  double interest = getAccountBalance() \* interestRate;  double newBalance = getAccountBalance() + interest;  setAccountBalance(newBalance);  System.*out*.println("Interest added: " + interest);  System.*out*.println("New balance after interest: " + newBalance);  }   public void printAccountInfo() {  super.printAccountInfo();  System.*out*.println("Interest Rate: " + interestRate);  } } |

Output:





Task 9:

Bank.java

|  |
| --- |
| package org.example;  import java.util.Scanner;  public class Bank {  public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);  BankAccount account = null;   System.*out*.println("Choose Account Type:");  System.*out*.println("1. Savings Account");  System.*out*.println("2. Current Account");  System.*out*.print("Enter choice: ");  int choice = scanner.nextInt();   switch (choice) {  case 1:  System.*out*.print("Enter Account Number: ");  String saNumber = scanner.next();  System.*out*.print("Enter Customer Name: ");  scanner.nextLine(); // consume newline  String saName = scanner.nextLine();  System.*out*.print("Enter Balance: ");  float saBalance = scanner.nextFloat();  System.*out*.print("Enter Interest Rate: ");  float rate = scanner.nextFloat();  account = new SavingsAccount(saNumber, saName, saBalance, rate);  break;   case 2:  System.*out*.print("Enter Account Number: ");  String caNumber = scanner.next();  System.*out*.print("Enter Customer Name: ");  scanner.nextLine(); // consume newline  String caName = scanner.nextLine();  System.*out*.print("Enter Balance: ");  float caBalance = scanner.nextFloat();  account = new CurrentAccount(caNumber, caName, caBalance);  break;   default:  System.*out*.println("Invalid option.");  return;  }   System.*out*.println("\nAccount Created Successfully!");  account.printAccountInfo();   // Deposit  System.*out*.print("\nEnter amount to deposit: ");  float depositAmount = scanner.nextFloat();  account.deposit(depositAmount);   // Withdraw  System.*out*.print("Enter amount to withdraw: ");  float withdrawAmount = scanner.nextFloat();  account.withdraw(withdrawAmount);   // Calculate interest  account.calculateInterest();   // Final Info  System.*out*.println("\nFinal Account Info:");  account.printAccountInfo();  } } |

BankAccount.java

|  |
| --- |
| package org.example;  public abstract class BankAccount {  protected String accountNumber;  protected String customerName;  protected float balance;   public BankAccount() {}   public BankAccount(String accountNumber, String customerName, float balance) {  this.accountNumber = accountNumber;  this.customerName = customerName;  this.balance = balance;  }   public String getAccountNumber() {  return accountNumber;  }   public void setAccountNumber(String accountNumber) {  this.accountNumber = accountNumber;  }   public String getCustomerName() {  return customerName;  }   public void setCustomerName(String customerName) {  this.customerName = customerName;  }   public float getBalance() {  return balance;  }   public void setBalance(float balance) {  this.balance = balance;  }   public void printAccountInfo() {  System.*out*.println("Account Number: " + accountNumber);  System.*out*.println("Customer Name: " + customerName);  System.*out*.println("Balance: " + balance);  }   public abstract void deposit(float amount);  public abstract void withdraw(float amount);  public abstract void calculateInterest(); } |

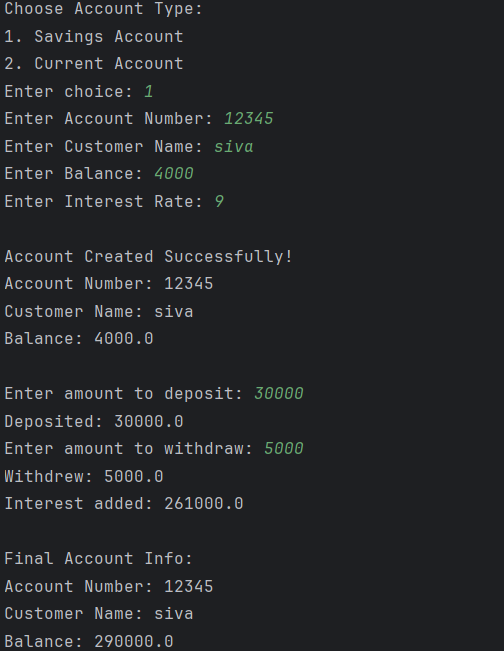
SavingsAccount.java

|  |
| --- |
| package org.example;  public class SavingsAccount extends BankAccount {  private float interestRate;   public SavingsAccount(String accountNumber, String customerName, float balance, float interestRate) {  super(accountNumber, customerName, balance);  this.interestRate = interestRate;  }   @Override  public void deposit(float amount) {  if (amount > 0) {  balance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   @Override  public void withdraw(float amount) {  if (amount > 0 && amount <= balance) {  balance -= amount;  System.*out*.println("Withdrew: " + amount);  } else {  System.*out*.println("Insufficient balance.");  }  }   @Override  public void calculateInterest() {  float interest = balance \* interestRate;  balance += interest;  System.*out*.println("Interest added: " + interest);  } } |

CurrentAccount.java

|  |
| --- |
| package org.example;  public class CurrentAccount extends BankAccount {  private static final float *OVERDRAFT\_LIMIT* = 5000;   public CurrentAccount(String accountNumber, String customerName, float balance) {  super(accountNumber, customerName, balance);  }   @Override  public void deposit(float amount) {  if (amount > 0) {  balance += amount;  System.*out*.println("Deposited: " + amount);  } else {  System.*out*.println("Invalid deposit amount.");  }  }   @Override  public void withdraw(float amount) {  if (amount > 0 && (balance + *OVERDRAFT\_LIMIT*) >= amount) {  balance -= amount;  System.*out*.println("Withdrew: " + amount);  } else {  System.*out*.println("Overdraft limit exceeded.");  }  }   @Override  public void calculateInterest() {  System.*out*.println("No interest for current account.");  } } |

Output :



Task 10:

BankApp.java

|  |
| --- |
| package org.example; import java.util.\*;  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 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 choice = sc.nextInt();  sc.nextLine();   switch (choice) {  case 1:  System.*out*.print("Enter Customer ID: ");  int custId = sc.nextInt();  sc.nextLine();  System.*out*.print("Enter First Name: ");  String fName = sc.nextLine();  System.*out*.print("Enter Last Name: ");  String lName = sc.nextLine();  System.*out*.print("Enter Email: ");  String email = sc.nextLine();  System.*out*.print("Enter Phone: ");  String phone = sc.nextLine();  System.*out*.print("Enter Address: ");  String addr = sc.nextLine();   System.*out*.print("Enter Account Type (Savings/Current): ");  String accType = sc.nextLine();  System.*out*.print("Enter Initial Balance: ");  float balance = sc.nextFloat();   try {  Customer cust = new Customer(custId, fName, lName, email, phone, addr);  bank.createAccount(cust, accType, balance);  } catch (IllegalArgumentException e) {  System.*out*.println("Error: " + e.getMessage());  }  break;   case 2:  System.*out*.print("Enter Account Number: ");  long depAcc = sc.nextLong();  System.*out*.print("Enter Deposit Amount: ");  float depAmt = sc.nextFloat();  System.*out*.println("Updated Balance: " + bank.deposit(depAcc, depAmt));  break;   case 3:  System.*out*.print("Enter Account Number: ");  long witAcc = sc.nextLong();  System.*out*.print("Enter Withdrawal Amount: ");  float witAmt = sc.nextFloat();  System.*out*.println("Updated Balance: " + bank.withdraw(witAcc, witAmt));  break;   case 4:  System.*out*.print("Enter Account Number: ");  long balAcc = sc.nextLong();  System.*out*.println("Current Balance: " + bank.getAccountBalance(balAcc));  break;   case 5:  System.*out*.print("Enter Sender Account Number: ");  long fromAcc = sc.nextLong();  System.*out*.print("Enter Receiver Account Number: ");  long toAcc = sc.nextLong();  System.*out*.print("Enter Amount to Transfer: ");  float amt = sc.nextFloat();  bank.transfer(fromAcc, toAcc, amt);  break;   case 6:  System.*out*.print("Enter Account Number: ");  long detailsAcc = sc.nextLong();  bank.getAccountDetails(detailsAcc);  break;   case 7:  System.*out*.println("Exiting. Thank you!");  sc.close();  System.*exit*(0);  }  }  } } |

Account.java

|  |
| --- |
| package org.example; public class Account {  private long accountNumber;  private String accountType;  private float balance;  private Customer customer;   public Account() {}   public Account(long accountNumber, String accountType, float balance, Customer customer) {  this.accountNumber = accountNumber;  this.accountType = accountType;  this.balance = balance;  this.customer = customer;  }   public long getAccountNumber() { return accountNumber; }  public String getAccountType() { return accountType; }  public float getBalance() { return balance; }  public Customer getCustomer() { return customer; }   public void setAccountNumber(long accountNumber) { this.accountNumber = accountNumber; }  public void setAccountType(String accountType) { this.accountType = accountType; }  public void setBalance(float balance) { this.balance = balance; }  public void setCustomer(Customer customer) { this.customer = customer; }   public void deposit(float amount) {  balance += amount;  System.*out*.println("Amount deposited successfully.");  }   public void withdraw(float amount) {  if (amount <= balance) {  balance -= amount;  System.*out*.println("Amount withdrawn successfully.");  } else {  System.*out*.println("Insufficient balance.");  }  }   public void printAccountInfo() {  System.*out*.println("Account Number: " + accountNumber);  System.*out*.println("Account Type: " + accountType);  System.*out*.println("Balance: " + balance);  System.*out*.println("--- Customer Details ---");  customer.printCustomerInfo();  } } |

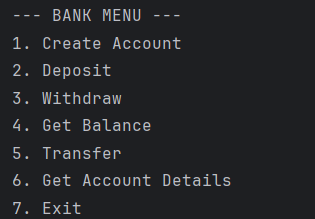
Bank.java

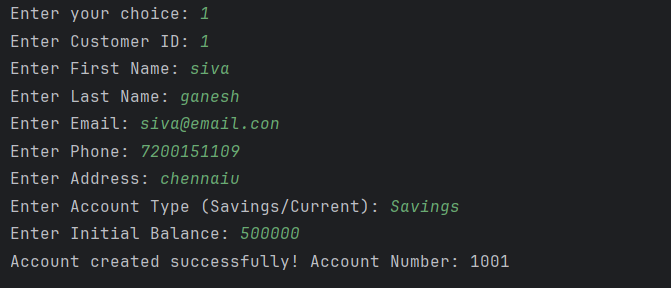
|  |
| --- |
| package org.example;  import java.util.\*;  public class Bank {  private Map<Long, Account> accounts = new HashMap<>();  private static long *nextAccountNumber* = 1001;   public long createAccount(Customer customer, String accType, float balance) {  long accNo = *nextAccountNumber*++;  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 account = accounts.get(accNo);  if (account != null) {  return account.getBalance();  }  System.*out*.println("Account not found.");  return -1;  }   public float deposit(long accNo, float amount) {  Account account = accounts.get(accNo);  if (account != null) {  account.deposit(amount);  return account.getBalance();  }  System.*out*.println("Account not found.");  return -1;  }   public float withdraw(long accNo, float amount) {  Account account = accounts.get(accNo);  if (account != null) {  account.withdraw(amount);  return account.getBalance();  }  System.*out*.println("Account not found.");  return -1;  }   public void transfer(long fromAccNo, long toAccNo, float amount) {  Account fromAccount = accounts.get(fromAccNo);  Account toAccount = accounts.get(toAccNo);  if (fromAccount != null && toAccount != null) {  if (fromAccount.getBalance() >= amount) {  fromAccount.withdraw(amount);  toAccount.deposit(amount);  System.*out*.println("Transfer successful.");  } else {  System.*out*.println("Insufficient balance to transfer.");  }  } else {  System.*out*.println("Invalid account number(s).");  }  }   public void getAccountDetails(long accNo) {  Account account = accounts.get(accNo);  if (account != null) {  account.printAccountInfo();  } else {  System.*out*.println("Account not found.");  }  } } |

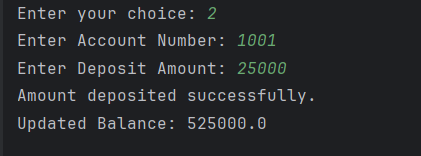
Customer.java

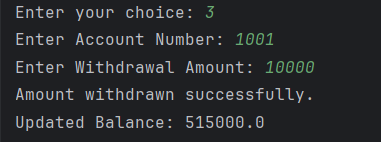
|  |
| --- |
| package org.example; public class Customer {  private int customerId;  private String firstName;  private String lastName;  private String email;  private String phone;  private String address;   public Customer() {}   public Customer(int customerId, String firstName, String lastName, String email, String phone, String address) {  if (!email.matches("^[\\w.-]+@[\\w.-]+\\.\\w+$")) {  throw new IllegalArgumentException("Invalid email address");  }  if (!phone.matches("\\d{10}")) {  throw new IllegalArgumentException("Phone number must be 10 digits");  }  this.customerId = customerId;  this.firstName = firstName;  this.lastName = lastName;  this.email = email;  this.phone = phone;  this.address = address;  }   public int getCustomerId() { return customerId; }  public String getFirstName() { return firstName; }  public String getLastName() { return lastName; }  public String getEmail() { return email; }  public String getPhone() { return phone; }  public String getAddress() { return address; }   public void setCustomerId(int customerId) { this.customerId = customerId; }  public void setFirstName(String firstName) { this.firstName = firstName; }  public void setLastName(String lastName) { this.lastName = lastName; }  public void setEmail(String email) {  if (!email.matches("^[\\w.-]+@[\\w.-]+\\.\\w+$")) {  throw new IllegalArgumentException("Invalid email address");  }  this.email = email;  }   public void setPhone(String phone) {  if (!phone.matches("\\d{10}")) {  throw new IllegalArgumentException("Phone number must be 10 digits");  }  this.phone = phone;  }   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: " + email);  System.*out*.println("Phone: " + phone);  System.*out*.println("Address: " + address);  } } |

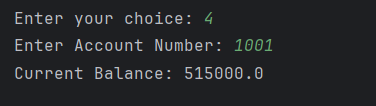
Output:

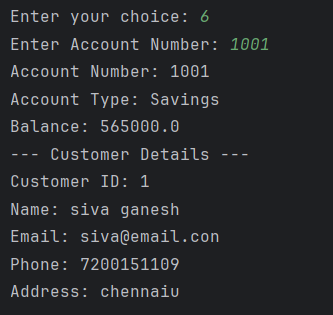
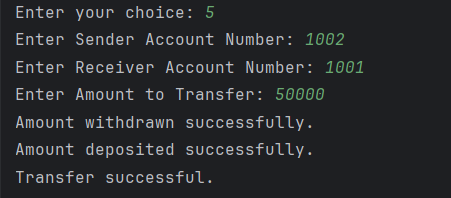


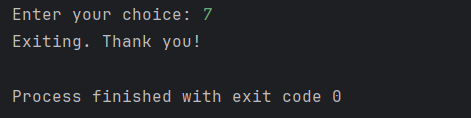












Task 11:

Main/BankApp.java:

|  |
| --- |
| package org.example.main;  import org.example.entity.Account; import org.example.entity.Customer; import org.example.service.BankServiceProviderImpl;  import java.util.Scanner;  public class BankApp {  public static void main(String[] args) {  Scanner sc = new Scanner(System.*in*);  BankServiceProviderImpl bank = new BankServiceProviderImpl();  System.*out*.println("Welcome to Sg Bank System");   while (true) {  System.*out*.println("\n================= 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. List All Accounts");  System.*out*.println("8. Calculate Interest");  System.*out*.println("9. Exit");  System.*out*.print("Enter your choice: ");   int choice;  try {  choice = Integer.*parseInt*(sc.nextLine());  } catch (Exception e) {  System.*out*.println(" Invalid input. Please enter a number from 1 to 9.");  continue;  }   switch (choice) {  case 1:  try {  System.*out*.println("Enter Customer ID:");  long id = Long.*parseLong*(sc.nextLine());  System.*out*.println("Enter First Name:");  String fn = sc.nextLine();  System.*out*.println("Enter Last Name:");  String ln = sc.nextLine();  System.*out*.println("Enter Email:");  String email = sc.nextLine();  System.*out*.println("Enter Phone Number:");  String ph = sc.nextLine();  System.*out*.println("Enter Address:");  String addr = sc.nextLine();   Customer cust = new Customer(id, fn, ln, email, ph, addr);   System.*out*.println("Choose Account Type:\n1. Savings\n2. Current\n3. ZeroBalance");  int typeChoice = Integer.*parseInt*(sc.nextLine());  String type = switch (typeChoice) {  case 1 -> "Savings";  case 2 -> "Current";  case 3 -> "ZeroBalance";  default -> {  System.*out*.println(" Invalid account type. Account not created.");  yield null;  }  };  if (type == null) break;   System.*out*.println("Enter Initial Balance:");  float bal = Float.*parseFloat*(sc.nextLine());   long accNo = bank.create\_account(cust, type, bal);  if (accNo != -1)  System.*out*.println(" Account created successfully! Your Account Number is: " + accNo);  else  System.*out*.println(" Account creation failed.");   } catch (Exception e) {  System.*out*.println(" Invalid input. Please try again.");  }  break;   case 2:  try {  System.*out*.println("Enter Account Number:");  long accNo = Long.*parseLong*(sc.nextLine());  System.*out*.println("Enter Amount to Deposit:");  float amount = Float.*parseFloat*(sc.nextLine());  float updatedBalance = bank.deposit(accNo, amount);  if (updatedBalance != -1)  System.*out*.println(" Deposit successful. Updated Balance: " + updatedBalance);  else  System.*out*.println(" Account not found.");  } catch (Exception e) {  System.*out*.println(" Invalid input. Please try again.");  }  break;   case 3:  try {  System.*out*.println("Enter Account Number:");  long accNo = Long.*parseLong*(sc.nextLine());  System.*out*.println("Enter Amount to Withdraw:");  float amount = Float.*parseFloat*(sc.nextLine());  float updatedBalance = bank.withdraw(accNo, amount);  if (updatedBalance != -1)  System.*out*.println(" Withdraw successful. Updated Balance: " + updatedBalance);  else  System.*out*.println(" Withdraw failed. Check balance or account number.");  } catch (Exception e) {  System.*out*.println(" Invalid input. Please try again.");  }  break;   case 4:  try {  System.*out*.println("Enter Account Number:");  long accNo = Long.*parseLong*(sc.nextLine());  float balance = bank.getAccountBalance(accNo);  if (balance != -1)  System.*out*.println("✅ Account Balance: ₹" + balance);  else  System.*out*.println("❌ Account not found.");  } catch (Exception e) {  System.*out*.println("❌ Invalid input. Please try again.");  }  break;   case 5:  try {  System.*out*.println("Enter Sender Account Number:");  long from = Long.*parseLong*(sc.nextLine());  System.*out*.println("Enter Receiver Account Number:");  long to = Long.*parseLong*(sc.nextLine());  System.*out*.println("Enter Amount to Transfer:");  float amount = Float.*parseFloat*(sc.nextLine());  bank.transfer(from, to, amount);  } catch (Exception e) {  System.*out*.println(" Invalid input. Please try again.");  }  break;   case 6:  try {  System.*out*.println("Enter Account Number:");  long accNo = Long.*parseLong*(sc.nextLine());  bank.getAccountDetails(accNo);  } catch (Exception e) {  System.*out*.println(" Invalid input. Please try again.");  }  break;   case 7:  Account[] accounts = bank.listAccounts();  if (accounts.length == 0) {  System.*out*.println("No accounts found.");  } else {  System.*out*.println("List of Accounts:");  for (Account a : accounts) {  System.*out*.println("Account No: " + a.getAccountNumber() +  ", Type: " + a.getAccountType() +  ", Balance: ₹" + a.getAccountBalance() +  ", Owner: " + a.getCustomer().getFirstName() + " " + a.getCustomer().getLastName());  }  }  break;   case 8:  bank.calculateInterest();  break;   case 9:  System.*out*.println("Thank you for using Sg Bank System. Goodbye!");  sc.close();  System.*exit*(0);   default:  System.*out*.println(" Invalid choice. Please select between 1 and 9.");  }  }  } } |

Entity/account.java

|  |
| --- |
| package org.example.entity;  public abstract class Account {  private static long *lastAccNo* = 1000;  private long accountNumber;  private String accountType;  private float accountBalance;  private Customer customer;   public Account() {  this.accountNumber = ++*lastAccNo*;  }   public Account(String accountType, float accountBalance, Customer customer) {  this.accountNumber = ++*lastAccNo*;  this.accountType = accountType;  this.accountBalance = accountBalance;  this.customer = customer;  }   public long getAccountNumber() {  return accountNumber;  }   public String getAccountType() {  return accountType;  }   public float getAccountBalance() {  return accountBalance;  }   public void setAccountBalance(float accountBalance) {  this.accountBalance = accountBalance;  }   public Customer getCustomer() {  return customer;  }   public void calculateInterest() {  } } |

Entity/currentAccount.java

|  |
| --- |
| package org.example.entity;  public class CurrentAccount extends Account {  private final float overdraftLimit;   public CurrentAccount(float balance, Customer customer, float overdraftLimit) {  super("Current", balance, customer);  this.overdraftLimit = overdraftLimit;  }   @Override  public void calculateInterest() {}   public float getOverdraftLimit() {  return overdraftLimit;  } } |

Entity/savingsAccount.java:

|  |
| --- |
| package org.example.entity;  public class SavingsAccount extends Account {  private float interestRate;   public SavingsAccount(float balance, Customer customer, float interestRate) {  super("Savings", balance >= 500 ? balance : 500, customer);  this.interestRate = interestRate;  }   @Override  public void calculateInterest() {  float interestRate = 0.05f; // 5% interest  float interest = getAccountBalance() \* interestRate;  setAccountBalance(getAccountBalance() + interest);  } } |

Entity/Customer.java:

|  |
| --- |
| package org.example.entity;  public class SavingsAccount extends Account {  private float interestRate;   public SavingsAccount(float balance, Customer customer, float interestRate) {  super("Savings", balance >= 500 ? balance : 500, customer);  this.interestRate = interestRate;  }   @Override  public void calculateInterest() {  float interestRate = 0.05f; // 5% interest  float interest = getAccountBalance() \* interestRate;  setAccountBalance(getAccountBalance() + interest);  } } |

Entity/zeroBalanceAccount.java:

|  |
| --- |
| package org.example.entity;  public class ZeroBalanceAccount extends Account {  public ZeroBalanceAccount(Customer customer) {  super("ZeroBalance", 0, customer);  }   @Override  public void calculateInterest() {} } |

Service/BankServiceProviderImpl:

|  |
| --- |
| package org.example.service;  import org.example.entity.\*;  public class BankServiceProviderImpl extends CustomerServiceProviderImpl implements IBankServiceProvider {  private String branchName = "Hexaware Bank";  private String branchAddress = "Chennai";   @Override  public long create\_account(Customer customer, String accType, float balance) {  Account newAccount = null;  switch (accType.toLowerCase()) {  case "savings":  if (balance < 500) {  System.*out*.println(“ Minimum balance for Savings Account is ₹500.");  return -1;  }  newAccount = new SavingsAccount(balance, customer, 4.0f);  break;  case "current":  newAccount = new CurrentAccount(balance, customer, 10000.0f);  break;  case "zerobalance":  newAccount = new ZeroBalanceAccount(customer);  break;  default:  System.*out*.println(" Invalid account type.");  return -1;  }   accountList[accountCount++] = newAccount;  return newAccount.getAccountNumber(); // Return the new account number  }   @Override  public Account[] listAccounts() {  Account[] list = new Account[accountCount];  System.*arraycopy*(accountList, 0, list, 0, accountCount);  return list;  }   @Override  public void calculateInterest() {  for (int i = 0; i < accountCount; i++) {  Account acc = accountList[i];  float oldBalance = acc.getAccountBalance();   acc.calculateInterest(); // subclass handles logic   float newBalance = acc.getAccountBalance();  if (newBalance > oldBalance) {  System.*out*.println(" Interest added to Account No: " + acc.getAccountNumber());  System.*out*.printf("Old Balance: ₹%.2f, New Balance: ₹%.2f%n", oldBalance, newBalance);  }  }  System.*out*.println("Interest calculation completed for all applicable accounts.");  } } |

Service/CustomerServiceProviderImpl:

|  |
| --- |
| package org.example.service;  import org.example.entity.\*;  public class CustomerServiceProviderImpl implements ICustomerServiceProvider {  protected Account[] accountList = new Account[100];  protected int accountCount = 0;   @Override  public float getAccountBalance(long accNo) {  Account acc = findAccount(accNo);  return acc != null ? acc.getAccountBalance() : -1;  }   @Override  public float deposit(long accNo, float amount) {  Account acc = findAccount(accNo);  if (acc != null) {  acc.setAccountBalance(acc.getAccountBalance() + amount);  return acc.getAccountBalance();  }  return -1;  }   @Override  public float withdraw(long accNo, float amount) {  Account acc = findAccount(accNo);  if (acc == null) return -1;   if (acc instanceof CurrentAccount) {  float newBalance = acc.getAccountBalance() - amount;  if (newBalance >= -((CurrentAccount) acc).getOverdraftLimit()) {  acc.setAccountBalance(newBalance);  return newBalance;  }  } else if (acc instanceof SavingsAccount || acc instanceof ZeroBalanceAccount) {  if (acc.getAccountBalance() - amount >= 500 || acc instanceof ZeroBalanceAccount) {  acc.setAccountBalance(acc.getAccountBalance() - amount);  return acc.getAccountBalance();  }  }   return -1;  }   @Override  public void transfer(long fromAcc, long toAcc, float amount) {  float withdrawn = withdraw(fromAcc, amount);  if (withdrawn != -1) {  deposit(toAcc, amount);  }  }   @Override  public void getAccountDetails(long accNo) {  Account acc = findAccount(accNo);  if (acc != null) {  System.*out*.println("Account No: " + acc.getAccountNumber());  System.*out*.println("Type: " + acc.getAccountType());  System.*out*.println("Balance: " + acc.getAccountBalance());  System.*out*.println("Customer Info: " + acc.getCustomer());  } else {  System.*out*.println("Account not found.");  }  }   protected Account findAccount(long accNo) {  for (int i = 0; i < accountCount; i++) {  if (accountList[i].getAccountNumber() == accNo) {  return accountList[i];  }  }  return null;  } } |

Service/IBankServiceProvider:

|  |
| --- |
| package org.example.service;  import org.example.entity.Customer; import org.example.entity.Account;  public interface IBankServiceProvider {  long create\_account(Customer customer, String accType, float balance);  Account[] listAccounts();  void calculateInterest(); } |

Service/IcustomerServiceProvider:

|  |
| --- |
| package org.example.service;  public interface ICustomerServiceProvider {  float getAccountBalance(long accNo);  float deposit(long accNo, float amount);  float withdraw(long accNo, float amount);  void transfer(long fromAcc, long toAcc, float amount);  void getAccountDetails(long accNo); } |

Output:

Same Output as of last task only the way of modularizing the code changed in this task……….

Task 12: