BANKING MANAGEMENT SYSTEM

A report submitted in partial fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

KOTTI VENKATA SAI MANIKANTA - 21B91A05H1

Under Supervision of Mr. A RAZA KHAN

Henotic IT Solutions Pvt Ltd

(Duration: 5th July 2023 to 5th September 2023)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

S.R.K.R. ENGINEERING COLLEGE

(Autonomous)

SRKR MARG, CHINNA AMIRAM, BHIMAVARAM-534204, A.P

(Recognized by A.I.C.T.E New Delhi) (Accredited by NBA & NAAC)

(Affiliated to JNTU, KAKINADA)

SAGI RAMA KRISHNAM RAJU ENGINEERING COLLEGE

(Autonomous)

DEPARTMENT OF ELECTRICAL AND ELECTONICS ENGINEERING

CERTIFICATE

This is to certify that the Summer Internship Report titled “Banking Management system” is the bonafide work done by Mr. **KOTTI VENKATA SAI MANIKANTA** bearing **21B91A05H1** at the end of second year second semester at Henotic IT Solutions Pvt Ltd from 5th July 2023 to 5th September 2023 in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering.

Department Internship coordinator **Dean -T & P Cell** **Head of the Department**

INTRODUCTION **:**

A simple Banking Management System that enables users to perform basic banking operations, such as adding a customer, depositing money, withdrawing money, checking balance, printing transactions, displaying customer details, and removing customers, transfer amount from one customer account to another account**.**

When we enter the customer's name and initial amount then program automatically generate a five-digit number and also store with customer name.

Customer have to functions like depositing money, withdrawing money, checking balance, printing transactions. while come to depositing the money, it have a customer account number, if account number is wrong it display the error else amount will be deposited .while come to withdrawing the money, it have a customer account number ,if account number is wrong it display the error and also if we withdrawing the amount more than then it display the error else amount will be withdraw.

Using two Hash map, one for storing of customer name, initial amount here customer name is act as key of hashmap and amount is a value. Another map is storing for Account number, name of customer here account number act as key of hashmap and name of customer act as value.

Using Stack for storing Transaction Details, like customer deposit the amount it stores the amount and deposit text. If customer withdraw the amount it stores the amount and withdraw text.

HashMap :

A hash map is a specific type of dictionary that uses a hashing function to efficiently store and retrieve key-value pairs.

It typically provides constant-time average complexity for basic operations like insertion, deletion, and retrieval, making it highly efficient for large data sets.

Here's a more detailed description of the key concepts and operations associated with maps:

Key Characteristics of Maps:

Keys **:** These are unique identifiers for each element in the map. Keys are used to access or retrieve the associated values. Keys must be distinct within a map.

Values **:** These are the data associated with the keys. Values can be of any data type, including numbers, strings, objects, or even other maps.

Key-Value Pairs **:** A map is made up of key-value pairs, where each key is associated with one specific value.

Common Operations on Maps:

Insertion **:** To add a new key-value pair to the map.

Deletion **:** To remove a key-value pair from the map.

Retrieval **:** To access the value associated with a specific key.
Update **:** To change the value associated with a specific key.
Iteration **:** To traverse through all key-value pairs in the map, typically in a specific order or through an iterator.

Stack :
A stack is a fundamental data structure in computer science and programming that follows the Last-In, First-Out (LIFO) principle. This means that the last element added to the stack is the first one to be removed. Stacks are used to manage data in a way that mimics real-world scenarios, such as a stack of books, plates, or any other physical items.

Key Characteristics of Stacks:
Last-In, First-Out (LIFO) **:** The last element added to the stack is the first one to be removed. Think of it like a stack of plates; the plate at the top is the first one you can remove.
Elements **:** Elements in a stack are often referred to as "items." Items are pushed onto the stack, and they are popped off the stack when they are removed.

Common Operations on Stacks:
Push **:** Adding an item to the top of the stack. This is the primary operation to insert elements into the stack.
Pop **:** Removing and returning the top item from the stack. This is the primary operation to remove elements from the stack.
Peek (or Top) **:** Viewing the top item without removing it. This operation allows you to check the item at the top of the stack without popping it.
Is Empty **:** Checking if the stack is empty. This is used to determine whether any elements are currently in the stack.

Features and Functionality of program:

1. Add Customer (Option 1)
    - Generates a random account number for the customer.
    - Prompts the user for the customer's name and initial balance.
    - Adds the customer's information to the bank's records.
2. Deposit (Option 2)
    - Allows the user to deposit funds into an existing customer's account.
    - Prompts for the account number and the deposit amount.
    - Updates the customer's account balance and records the transaction.
3. Withdraw (Option 3)
    - Enables users to withdraw funds from an existing customer's account.
    - Requires the account number and the withdrawal amount.
    - Checks for sufficient balance and updates the account balance accordingly.
4. Check Balance (Option 4)

- Allows users to check the balance of an existing customer's account.
- Displays the current account balance.

5. Print Transactions (Option 5)
   - Displays a list of transactions (deposits and withdrawals) for a specific customer's account.
6. Display All Customer Details (Option 6)
   - Shows all customer details, including account numbers, names, and total balances.
7. Remove a Customer (Option 7)
   - Removes a specific customer from the bank's records based on their account number.
8. Remove All Customers (Option 8)
   - Clears all customer data from the system.
9. Exit (Option 9)
   - Exits the program.

User Interface:

The user interacts with the program through a Command-Line Interface (CLI) where a series of options are presented, and the user can select an option by entering the corresponding number.

Key Components and Methods:

- cus_Accounts :

A map that stores customer names and their corresponding account balances.

- cus_Accounts_num :
  A map that stores customer account numbers and their corresponding customer names.
- Transactions :
  A stack to keep track of customer transactions.
- addCustomer(int number, String cus_Name, double initial_Bal):
  Adds a new customer to the system.
- rem_cos(int number):
  Removes a customer based on their account number.
- deposit(int number):
  Allows a customer to deposit funds.
- withdraw(int number):
  Allows a customer to withdraw funds.
- checkBalance(int number):
  Checks and displays a customer's account balance.
- display(int number):
  Displays transaction details for a customer.

- dis_all():
  Displays details of all customers.
- rem_all():
  Removes all customer data.
- TransactionDetails Class :
  This simple class is used to represent transaction details, including the customer name, transaction type (deposit or withdrawal), and the transaction amount.

SOFTWARE REQUIREMENTS SPECIFICATIONS

System configurations :

The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements.

Software Requirements :

Operating system : Windows 11 Ultimate.

Coding Language : JAVA.

IDE Used : Eclipse(IDE) Professional.

TECHNOLOGIES:-

JAVA LANGUAGE:

Java is a versatile, high-level programming language widely used in a variety of applications, Java is an object-oriented, class-based programming language renowned for its platform independence, reliability, and extensive libraries. Developed by Sun Microsystems (now owned by Oracle), Java has earned its popularity due to its "Write Once, Run Anywhere".

ECLIPSE(IDE):

An Integrated Development Environment (IDE) in Java is a software application

1. Code Editing: Java IDEs offer advanced code editing features such as syntax highlighting, auto- completion, and code formatting to enhance productivity and code quality

2. Project Management**:** IDEs help developers organize their projects by providing

project templates, source code version control integration, and build tools for managing dependencies.

3. Debugging**:** Debugging tools in an IDE allow developers to identify and fix issues in their Java code. Developers can set breakpoints, step through code, inspect variables, and view stack traces.

4. Compiler and Build Tools**:** Java IDEs often come with integrated compilers and build tools like Ant, Maven, or Gradle, streamlining the process of building and running Java application

HARDWARE REQUIREMENTS

1. Processor-Intel Core I5
2. RAM-8GB

***PROGRAM* :**

Main Class **:**

```
package com.project;
import java.util.*;
import java.util.concurrent.ThreadLocalRandom;

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

      System.out.println("\t\t\t--WELCOME--");
      while (true)
      {
          System.out.println();
          System.out.println("(1) Add Customer  ");
         System.out.print("(2) Deposit  ");
         System.out.print("\t\t\t\t (3) Withdraw   ");
```

```
System.out.print("\t\t(4) Check Balance   ");
System.out.println("\t(5) Print Transactions    ");
System.out.print("(6) Display All Customer Details  ");
System.out.print("\t (7) Remove a Customer ");
System.out.println("\t(8) Remove a11 Customer  ");
System.out.println("(9) Customer Acc To Another Customer Acc   ");
System.out.println("(10)Exit");
System.out.print("\nSelect an option: ");

int option = scanner.nextInt();
String cus_Name;
switch (option)
{
 case 1:
            int number;
            do {
           number = ThreadLocalRandom.current().nextInt(10000, 99999);
    } while (bank.cus_Accounts_num.containsKey(number));

            System.out.print("Enter customer Name: ");
            scanner.nextLine();
            cus_Name = scanner.nextLine();

            System.out.print("Initial Balance: ");
            double inital_Bal = scanner.nextDouble();

              System.out.println("Account number is :"+ "XXXXXXX"+number);
              bank.addCustomer(number,cus_Name, inital_Bal);
              System.out.println("----------------------");
               break;
   case 2:
            System.out.print("Enter customer Acc_num: ");
```

```
                number = scanner.nextInt();

               bank.deposit(number);
               System.out.println("----------------------");
               break;
        case 3:
               System.out.print("Enter customer Acc_num: ");
               number = scanner.nextInt();

               bank.withdraw(number);
               System.out.println("----------------------");
               break;
        case 4:
               System.out.print("Enter customer Acc_num: ");
               number = scanner.nextInt();

               bank.checkBalance(number);
               System.out.println("----------------------");
               break;
        case 5:
               System.out.print("Enter customer Acc_num: ");
               number = scanner.nextInt();

               bank.display(number);
               System.out.println("----------------------");
               break;
        case 6:
               bank.dis_all();
               System.out.println("----------------------");
               break;
        case 7:
               System.out.print("Enter customer Acc_num: ");
```

```
                number = scanner.nextInt();


                bank.rem_cos(number);
                System.out.println("----------------------");
                break;
            case 8:
                bank.rem_all();
                System.out.println("----------------------");
                break;
            case 9:
                System.out.println("We can transfer the amount, for these bank customer account only ");
                System.out.print("enter your Acc number  : ");
                int number1 = scanner.nextInt();

        System.out.println("\t customer name:"+ bank.cus_Accounts_num.get(number1));
                System.out.print("enter Another customer Acc number  : ");
                int number2 = scanner.nextInt();
         System.out.println("\t customer name :- "+bank.cus_Accounts_num.get(number2));


                bank.another_Acc(number1,number2);
                System.out.println("----------------------");
                break;


            case 10:
                scanner.close();
                System.err.print("U EXIT");
                System.exit(0);
            default:
                System.out.println("Invalid option.");
                break;
        }
```

```
        }
      }
    }
```

Bank Class :

```
    package com.project;
    import java.util.*;
    class Bank
    {
      Map<String, Double> cus_Accounts = new HashMap<>();
      Map<Integer,String > cus_Accounts_num = new HashMap<>();
      Stack<TransactionDetails> transactions = new Stack<>();

      Scanner scanner = new Scanner(System.in);
      //adding a customer
      public void addCustomer(int number,String cus_Name, double inital_Bal)
      {
          cus_Accounts.put(cus_Name, inital_Bal);
          cus_Accounts_num.put(number, cus_Name);
      }

      //remove a coustermer
      public void rem_cos(int number)
      {
          String name1 = cus_Accounts_num.get(number);
              for(String name2 : cus_Accounts.keySet())
              {
                  if(name1 == name2)
                  {
                      cus_Accounts_num.remove(number);
```

```
                        cus_Accounts.remove(name1);
                        System.out.println(name1+" customer is remove ");
                        break;
                    }
                }
                if(name1 == null)
                {
                    System.out.println("customer not found ");
                }
        }
//amount deposit
public void deposit(int number)
{
        String name1 = cus_Accounts_num.get(number);
        for(String name2 : cus_Accounts.keySet())
                {
                    if(name1 == name2)
                    {
                        System.out.println("customer name :- "+name1);
                        System.out.print("\tEnter deposit amount: ");
                        double amount = scanner.nextDouble();
                        double balance = cus_Accounts.get(name1);
                        balance = balance+amount;
                System.out.println("total amount : "+balance);
                cus_Accounts.put(name1, balance);
                transactions.push(new TransactionDetails(name1,"Deposit ", amount));
                    }
                }
        if(name1 == null)
```

```
                {
                        System.out.println("Costomer not found ");
                }
  }

//amount withdraw
  public void withdraw(int number)
  {
        String name1 = cus_Accounts_num.get(number);
        for(String name2 : cus_Accounts.keySet())
                {
                        if(name1 == name2)
                        {
                                System.out.println("customer name :- "+name1);
                                System.out.print("\tEnter withdrawal amount: ");
                                 double amount = scanner.nextDouble();
                                double balance = cus_Accounts.get(name1);
                                if(amount <= balance)
                                {
                                        balance = balance-amount;
                            System.out.println("total amount : "+balance);
                                cus_Accounts.put(name1, balance);
                transactions.push(new TransactionDetails(name1,"Withdraw ", amount));
                                }
                                else
                                 {
                           System.out.println("Insufficient balance.");
                       }
                }
```

```
                }
        if(name1 == null)
                {
                        System.out.println("costomer not found ");
                }
}

//Balance check
public void checkBalance(int number)
{
        String name1 = cus_Accounts_num.get(number);
        for(String name2 : cus_Accounts.keySet())
                {
                        if(name1 == name2)
                        {
                                double balance = cus_Accounts.get(name1);
                     System.out.println("total amount : "+balance);
                        }
                }
        if(name1 == null)
                {
                        System.out.println("costomer not found ");
                }
}

//display all transataion
public void display(int number)
{
   System.out.println("Transaction\t\tAmount");
```

```
    String name1 = cus_Accounts_num.get(number);
        for(String name2 : cus_Accounts.keySet())
                {
                        if(name1 == name2)
                        {
                                for (TransactionDetails i : transactions)
                            {
                                                if(i.name.equals(name1))
                                                {
                                        System.out.println(i.type+"\t\t"+i.amount );
                                    }
                            }
                        }
                }
        if(name1 == null)
                {
                        System.out.println("costomer not found ");
                }
}
//display all cous
        public void dis_all()
        {
                for (int number : cus_Accounts_num.keySet())
                {
                        String name1 = cus_Accounts_num.get(number);
                        for(String name2 : cus_Accounts.keySet())
                        {
                                double amount = cus_Accounts.get(name2);
                                if(name1 == name2)
```

```
                                {
                                        System.out.println("Acc_Num : XXXXXXX" +number+"\n
\t customer name is : "+name1+"\n  Total amount is : "+amount+"\n");
                                }
                        }
                }
        }
        //remove all coustremers
        public void rem_all()
        {
                cus_Accounts.clear();
                cus_Accounts_num.clear();
                System.out.println("All cousters are remove ");
        }

        //transfer one acc to another account
        public int another_Acc(int number1, int number2) {

                String name11 = cus_Accounts_num.get(number1);
                String name12 = cus_Accounts_num.get(number2);
                if (name11 == null) {
                        System.out.println(" Sender costomer not found ");
                        return 0;
                }
                if (name12 == null) {
                        System.out.println(" Receiver costomer not found ");
                        return 0;
                }
```

```
                for(String name2 : cus_Accounts.keySet())
                {
                        if(name11 == name2)
                        {
                                System.out.print("Enter the amount: ");
                                double amount = scanner.nextDouble();
                                double balance = cus_Accounts.get(name11);
                                if(amount <= balance)
                                {
                                        balance = balance-amount;
                            System.out.println("your total amount : "+balance);
                            cus_Accounts.put(name11, balance);
                        transactions.push(new TransactionDetails(name11,"Transfer to another Acc ", amount));
                            System.out.println("Transaction Successfully completed ");

                            double balance1 = cus_Accounts.get(name12);
                                        balance1 = balance1+amount;
                            cus_Accounts.put(name12, balance1);
                            transactions.push(new TransactionDetails(name12,"Receive from another Acc ", amount));
                                }
                                else
                    {
                       System.out.println("Insufficient balance.");
                    }
                        }
                }
                return 0;
        }
```

```
}
```

TransactionDetails Class :

```
package com.project;

class TransactionDetails
{
	String name;
	String type;
	double amount;
	public TransactionDetails(String name,String type, double amount)
	{
		this.name = name;
		this.type = type;
		this.amount = amount;
	}
}
```

OUTPUT:

```
                                        --WELCOME--

(1) Add Customer
(2) Deposit          (3) Withdraw     (4) Check Balance    (5)Print Transactions
(6) Display All Customer Details      (7) Remove a Customer    (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 1
Enter customer Name: k mani
Initial Balance: 100
Account number is :XXXXXXX54204
----------------------

(1) Add Customer
(2) Deposit          (3) Withdraw     (4) Check Balance    (5)Print Transactions
(6) Display All Customer Details      (7) Remove a Customer    (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit
```

Select an option: 1
Enter customer Name: k aditya sai
Initial Balance: 200
Account number is :XXXXXXX63609
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 1
Enter customer Name: abhi ram
Initial Balance: 300
Account number is :XXXXXXX63386
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 1
Enter customer Name: ganesh
Initial Balance: 654123
Account number is :XXXXXXX91364
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 6
Acc_Num : XXXXXXX91364

customer name is : ganesh
Total amount is : 654123.0

Acc_Num : XXXXXXX63609
customer name is : k aditya sai
Total amount is : 200.0

Acc_Num : XXXXXXX63386
customer name is : abhi ram
Total amount is : 300.0

Acc_Num : XXXXXXX54204
customer name is : k mani
Total amount is : 100.0

---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 2
Enter customer Acc_num: 54204
customer name :- k mani
Enter deposit amount: 2000
total amount : 2100.0
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 2
Enter customer Acc_num: 54204
customer name :- k mani

```
        Enter deposit amount: 1
total amount : 2101.0
----------------------

(1) Add Customer
(2) Deposit          (3) Withdraw      (4) Check Balance     (5)Print Transactions
(6) Display All Customer Details       (7) Remove a Customer     (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit


Select an option: 3
Enter customer Acc_num: 54204
customer name :- k mani
        Enter withdrawal amount: 1
total amount : 2100.0
----------------------

(1) Add Customer
(2) Deposit          (3) Withdraw      (4) Check Balance     (5)Print Transactions
(6) Display All Customer Details       (7) Remove a Customer     (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 4
Enter customer Acc_num: 54204
total amount : 2100.0
----------------------

(1) Add Customer
(2) Deposit          (3) Withdraw      (4) Check Balance     (5)Print Transactions
(6) Display All Customer Details       (7) Remove a Customer     (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit


Select an option: 5
Enter customer Acc_num: 54204
Transaction          Amount
Deposit              2000.0
Deposit              1.0
```

```
Withdraw                1.0
----------------------

(1) Add Customer
(2) Deposit             (3) Withdraw        (4) Check Balance       (5)Print Transactions
(6) Display All Customer Details        (7) Remove a Customer       (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit


Select an option: 2
Enter customer Acc_num: 123
Costomer not found
----------------------

(1) Add Customer
(2) Deposit             (3) Withdraw        (4) Check Balance       (5)Print Transactions
(6) Display All Customer Details        (7) Remove a Customer       (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit


Select an option: 2
Enter customer Acc_num: 63609
customer name :- k aditya sai
        Enter deposit amount: 2
total amount : 202.0
----------------------

(1) Add Customer
(2) Deposit             (3) Withdraw        (4) Check Balance       (5)Print Transactions
(6) Display All Customer Details        (7) Remove a Customer       (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit


Select an option: 3
Enter customer Acc_num: 63609
customer name :- k aditya sai
        Enter withdrawal amount: 300
Insufficient balance.
```

---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 6
Acc_Num : XXXXXXX91364
customer name is : ganesh
Total amount is : 654123.0

Acc_Num : XXXXXXX63609
customer name is : k aditya sai
Total amount is : 202.0

Acc_Num : XXXXXXX63386
customer name is : abhi ram
Total amount is : 300.0

Acc_Num : XXXXXXX54204
customer name is : k mani
Total amount is : 2100.0

---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 9
We can transfer the amount, for these bank customer account only
enter your Acc number : 63609
customer name :- k aditya sai
enter Another customer Acc number : 54204
customer name :- k mani

Enter the amount: 100000
Insufficient balance.
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 9
We can transfer the amount, for these bank customer account only
enter your Acc number : 63609
customer name :- k aditya sai
enter Another customer Acc number : 54204
customer name :- k mani
Enter the amount: 200
your total amount : 2.0
Transaction Successfully completed
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 5
Enter customer Acc_num: 54204
Transaction Amount
Deposit 2000.0
Deposit 1.0
Withdraw 1.0
Receive from another Acc 200.0
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer

(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 5
Enter customer Acc_num: 63609
Transaction Amount
Deposit 2.0
Transfer to another Acc 200.0
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 7
Enter customer Acc_num: 91364
ganesh customer is remove
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 8
All cousters are remove
---

(1) Add Customer
(2) Deposit (3) Withdraw (4) Check Balance (5)Print Transactions
(6) Display All Customer Details (7) Remove a Customer (8) Remove a11 Customer
(9) Customer Acc To Another Customer Acc
(10)Exit

Select an option: 10
U EXIT