IMPROVED CIBILL SCORE SYSTEM

RESEARCH :

Summarize how the research extends the single-user CIBIL score model to a system that can store, compute, and analyze credit scores for multiple customers.
Mention how the project integrates data structures, credit evaluation and sorting algorithm to create a complete simulation of a credit rating database.

Explain:

- Why credit scoring is vital in financial systems.
- How a database of multiple customers helps simulate real-world banking scenarios.
- Your goal: To design a C program that accepts and stores customer data (name, age, account number, financial inputs) and automatically computes, sorts, and displays their CIBIL scores.

Objectives :

- To design a structured database using arrays and structs in C.
- To calculate the CIBIL score for each customer using input parameters (loan details, payment history, income, etc.).
- To implement sorting (ascending order) of customers based on their computed CIBIL scores.
- To display results in a tabular “report” format.
- To analyze relationships between different credit factors and the resulting CIBIL scores.

Sources :

Same as for Activity 3.

- https://www.researchgate.net/publication/356493603_Analysis_and_Prediction_of_CIBIL_Score_using_Machine_Learning
- https://www.scribd.com/document/508450015/CSS-Synopsis

ANALYSE :

The programn uses following steps :

1. Define CIBIL SCORE (eg: minimum = 300 , maximum = 900)
2. To calculate the CIBIL SCORE , take the following inputs :
    - Payment History
    - Credit Utilisation
    - Number of existing loans
    - Monthly Income
    - Name
    - Age
    - Account number

3. Compare calculated CIBIL SCORE to the ideal CIBIL SCORE range using if else conditions .
4. Collect the following data and give the calculated CIBIL SCORE to the customer and compare the CIBIL SCORE of 10 customers and the align them in ascending order .

ALGORITHM :

Step 1:

Start the program.

Step 2:

Declare the variables for Customer :

- Payment History
- Credit Utilisation
- Number of existing loans
- Monthly Income
- Name
- Age
- Account number

Step 3 :

Create a function for taking Input the following details from the user :

- Payment history (in %)
- Credit utilization (in %)
- Number of existing loans
- Monthly income
- Name
- Age
- Account Number

Step 4 :

Create a function for calculation Cibil score for a customer using following conditions :

1. Initialise score = 300

Step 5 :

Evaluate payment history :

- If payment history > 90, add 300 to score
- Else if payment history > 75, add 200 to score
- Else, add 100 to score

Step 6 :

Evaluate credit utilisation :

- If credit utilisation > 75, subtract 100 from score
- Else if credit utilisation > 50, subtract 50 from score

Step 7 :

Evaluate number of loans :

- If loans > 5, subtract 100 from score
- Else if loans >= 3, subtract 50 from score

Step 8 :

Evaluate income :

- If income > 50000, add 100 to score
- Else if income > 30000, add 50 to score

Step 9 :

Ensure the final score is within range

- If score > 900, set score = 900
- If score < 300, set score = 300

Step 10 :
Display all the input taken from the user

Step 11 :
Display the calculated CIBIL SCORE for all the 10 customers

Step 12 :
Make the CIBIL SCORE in the ascending order

Step 13 :
Stop

BUILD :

```
#include <stdio.h>

#define MAX_CUSTOMERS 10
#define MAX_NAME_LEN 50


struct Customer {
    char name[MAX_NAME_LEN];
    int age;
    long acc_no;
    int loans;
    int credit_utilisation;
    int payment_history;
    int income;
    int cibil_score;
};
```

```
int calculateCibil(struct Customer c) {
    int score = 300;


    if (c.payment_history > 90)
        score += 300;
    else if (c.payment_history > 75)
        score += 200;
    else
        score += 100;


    if (c.credit_utilisation > 75)
        score -= 100;
    else if (c.credit_utilisation > 50)
        score -= 50;
    else if (c.credit_utilisation > 25)
        score -= 25;
    else
        score += 100;


    if (c.loans > 5)
        score -= 100;
    else if (c.loans >= 3)
        score -= 50;
    else if (c.loans >= 1)
        score -= 25;
```

```
    else
        score += 100;


    if (c.income > 50000)
        score += 100;
    else if (c.income > 30000)
        score += 50;
    else
        score += 25;


    if (score < 300) score = 300;
    if (score > 900) score = 900;

    return score;
}


void inputCustomers(struct Customer customers[], int n) {
    for (int i = 0; i < n; i++) {

printf("\n==================================================
==================");
        printf("\nEnter details for Customer %d", i + 1);

printf("\n==================================================
==================");

        printf("\nName: ");
```

```
        scanf(" %[^\n]", customers[i].name);
        printf("Age: ");
        scanf("%d", &customers[i].age);
        printf("Account Number: ");
        scanf("%ld", &customers[i].acc_no);
        printf("Number of Existing Loans: ");
        scanf("%d", &customers[i].loans);
        printf("Credit Utilisation (%%): ");
        scanf("%d", &customers[i].credit_utilisation);
        printf("Payment History (%% of on-time payments): ");
        scanf("%d", &customers[i].payment_history);
        printf("Monthly Income (in Rs): ");
        scanf("%d", &customers[i].income);


        customers[i].cibil_score = calculateCibil(customers[i]);
    }
}

void sortByCibil(struct Customer customers[], int n) {
    struct Customer temp;
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            if (customers[i].cibil_score > customers[j].cibil_score) {
                temp = customers[i];
                customers[i] = customers[j];
                customers[j] = temp;
            }
        }
    }
```

```
}


void displayCustomers(struct Customer customers[], int n) {
   printf("\n\n============================== CUSTOMER REPORT
==============================\n");
   printf("%-20s %-5s %-12s %-10s %-20s %-20s %-10s %-12s\n",
          "Name", "Age", "Acc No", "Loans", "Credit Util(%)",
          "Payment Hist(%)", "Income", "CIBIL Score");
   printf("--------------------------------------------------------------------------------\n");

   for (int i = 0; i < n; i++) {
      printf("%-20s %-5d %-12ld %-10d %-20d %-20d %-10d %-12d\n",
             customers[i].name, customers[i].age, customers[i].acc_no,
             customers[i].loans, customers[i].credit_utilisation,
             customers[i].payment_history, customers[i].income,
             customers[i].cibil_score);
   }
}


int main() {
   struct Customer customers[MAX_CUSTOMERS];


printf("\n================================================
==================");
   printf("\n            CIBIL SCORE CALCULATION SYSTEM (10
CUSTOMERS)");
```

```
printf("\n==========================================================
==================\n");

    inputCustomers(customers, MAX_CUSTOMERS);
    sortByCibil(customers, MAX_CUSTOMERS);
    displayCustomers(customers, MAX_CUSTOMERS);



    return 0;
}
```

TESTING :

```
============================================================
===========
            CIBIL SCORE CALCULATION SYSTEM (10 CUSTOMERS)
============================================================
===========

============================================================
===========
Enter details for Customer 1
============================================================
===========
Name: VED MULEY
Age: 18
Account Number: 1
Number of Existing Loans: 1
Credit Utilisation (%): 12
```

Payment History (% of on-time payments): 99
Monthly Income (in Rs): 123234678

==========

==========

Enter details for Customer 2

==========

==========

Name: SARVESH MANDE
Age: 18
Account Number: 2
Number of Existing Loans: 2
Credit Utilisation (%): 45
Payment History (% of on-time payments): 78
Monthly Income (in Rs): 1231456

==========

==========

Enter details for Customer 3

==========

==========

Name: JAYESH KULKARNI
Age: 18
Account Number: 3
Number of Existing Loans: 3
Credit Utilisation (%): 34
Payment History (% of on-time payments): 76
Monthly Income (in Rs): 1274435

==========

==========

Enter details for Customer 4

==========

==========

Name: SHIVAM PATHADE

Age: 19

Account Number: 4

Number of Existing Loans: 4

Credit Utilisation (%): 24

Payment History (% of on-time payments): 89

Monthly Income (in Rs): 1222268

==========

==========

Enter details for Customer 5

==========

==========

Name: MOKSHAD PATIL

Age: 19

Account Number: 5

Number of Existing Loans: 1

Credit Utilisation (%): 67

Payment History (% of on-time payments): 67

Monthly Income (in Rs): 12467823

========== CUSTOMER REPORT

==========

Name Age Acc No Loans Credit Util(%) Payment Hist(%) Income CIBIL Score

---

MOKSHAD PATIL 19 5 1 67 67 12467823 425

JAYESH KULKARNI 18 3 3 34 76 1274435 525

SARVESH MANDE 18 2 2 45 78 1231456 550

SHIVAM PATHADE 19 4 4 24 89 1222268 650

VED MULEY 18 1 1 12 99 123234678 775

=== Code Execution Successful ===

```
============================== CUSTOMER REPORT ==============================
Name                 Age   Acc No      Loans     Credit Util(%)      Payment Hist(%)     Income    CIBIL Score
-------------------------------------------------------------------------------
MOKSHAD PATIL        19    5           1         67                  67                  12467823  425
JAYESH KULKARNI      18    3           3         34                  76                  1274435   525
SARVESH MANDE        18    2           2         45                  78                  1231456   550
SHIVAM PATHADE       19    4           4         24                  89                  1222268   650
VED MULEY            18    1           1         12                  99                  123234678 775
```

As we can see code has calculated the cibil score for 5 customers and also made it in ascending order.

IMPLEMENTATION :

https://github.com/vedmuley536/Ved-Muley