VEHICLE INFORMATION SYSTEM

**Research:**

This program is designed in a way where user need to input it’s vehical number displayed on the number plate. Vehicle information is stored which consists of owner’s name, driving lisence number, registration number, chassis number, vehicle model, vehicle color & previous fines. After entering vehicle number all the above information about the vehicle and the owner will be displayed.

**Reference links:**

[**https://parivahan.gov.in/parivahan//en/content/driving-licence-0**](https://parivahan.gov.in/parivahan//en/content/driving-licence-0)

[**https://www.carinfo.app/**](https://www.carinfo.app/)

[**https://www.geeksforgeeks.org/structures-c/**](https://www.geeksforgeeks.org/structures-c/)

**Analyze:**

For this we used C programming language.

Understanding how to use structures (struct) to group related data is essential. In this case, we are grouping vehicle-related data like number plate, owner’s name, and other attributes into a single structure, which can then be used to store and manage the records efficiently.

This code defines a simple program in C that retrieves car information based on a given number plate.

This program provides a straightforward way to search and display car records based on the number plate. It uses a linear search and relies on predefined data in the records array, which makes it simple but limited to small datasets.

The struct CarInfo defines a template for storing information about a car.

Functions like strcmp() are used to compare strings, and scanf() is used to capture user input.

The car records are stored in an array of structures (struct CarInfo records[]).

The displayCarInfo function uses a linear search to check each car record to see if the number plate matches the input.

 **Objective**: To provide an efficient way to retrieve and display vehicle details using a number plate as the search key.

 **Primary Use Case**: Law enforcement, traffic management, and parking attendants who need quick access to a vehicle’s details for identification or verification.

 **Scope**: Currently, the code should display basic car and owner information. Future iterations might include updating, deleting, or adding new records.

**Storage Options in C**

* C offers limited data storage options since it lacks built-in support for complex databases:
  + **Structs**: Using struct to define a custom data structure (CarInfo) allows different data types (like int for fines and char[] for textual fields) to be grouped logically.
  + **Array of Structs**: An array of CarInfo structs is chosen for simplicity, storing a small set of records. However, for a larger dataset, a more dynamic approach (e.g., linked lists) or even database integration might be considered.

Strings used in program are:

1. **char numberPlate[15];**
   * Used to store the number plate of the car entered by the user. It is a string of characters that uniquely identifies a car.
2. **char ownerName[50];**
   * Stores the name of the car's owner. It is a string with a maximum length of 50 characters.
3. **char drivingLicense[20];**
   * Stores the driving license number associated with the car’s owner. It can store a string of up to 20 characters.
4. **char registrationNumber[20];**
   * This string stores the registration number of the vehicle. It has a maximum length of 20 characters.
5. **char chassisNumber[20];**
   * Stores the chassis number, which is a unique identifier for the car. It can be up to 20 characters long.
6. **char model[30];**
   * Stores the model of the vehicle. The string is limited to 30 characters.
7. **char color[20];**
   * Stores the color of the car, with a maximum length of 20 characters.

**Ideate:**

Improving Data Handling:

Instead of hard-coding the car records, the program could allow users to add, delete, or modify records. This would create a more interactive and dynamic system.

Search Functionality:

Enhancing the search functionality to support partial matches or case-insensitive searching could be a useful addition. Additionally, offering more advanced search options (by model, color, etc.) could improve the program's versatility.

Persistent Data Storage:

Implementing file I/O operations would allow the program to read and write car records to a file, making the data persistent across different sessions of the program.

User Interface:

A simple text-based user interface (UI) could be added to allow users to interact with the system more easily, such as offering options to search, add new records, or view all records.

**Build:**

#include <stdio.h>

#include <string.h>

struct CarInfo {

    char numberPlate[15];

    char ownerName[50];

    char drivingLicense[20];

    char registrationNumber[20];

    char chassisNumber[20];

    char model[30];

    char color[20];

    int previousFines;

};

void displayCarInfo(char numberPlate[], struct CarInfo records[], int size) {

    int found = 0;

    for (int i = 0; i < size; i++) {

        if (strcmp(records[i].numberPlate, numberPlate) == 0) {

            printf("Car Details for Number Plate %s:\n", numberPlate);

            printf("Owner's Name: %s\n", records[i].ownerName);

            printf("Driving License Number: %s\n", records[i].drivingLicense);

            printf("Registration Number: %s\n", records[i].registrationNumber);

            printf("Chassis Number: %s\n", records[i].chassisNumber);

            printf("Vehicle Model: %s\n", records[i].model);

            printf("Vehicle Color: %s\n", records[i].color);

            printf("Previous Fines: %d\n", records[i].previousFines);

            found = 1;

            break;

        }

    }

    if (!found) {

        printf("No record found for this number plate.\n");

    }

}

int main() {

    struct CarInfo records[] = {

        {"MH12AB3456", "Rajesh Patil", "DL1234567890123", "MH12B1234", "MAHABC12345678900", "Maruti suzuki Swift", "Red", 2},

        {"UP14CD5678","Anita Verma", "DL9876543210123", "UP14E5678", "UPD12345678901234", "Honda City", "Silver", 0},

        {"TN22EF3456", "Karthik Ramesh", "DL1234567890123", "TN22A1234", "TNF12345678901234", "Hundai Creta", "Black", 1},

        {"GJ01HK7890","Priya Patel","DL2345678901234","GJ01B1234","GJD12345678901234","Tata Nexon","Blue", 2},

        {"WB20FG6789","Rita Das","DL5678901234567","WB20A6789","WBD12345678901234","Kia Sonet","White", 0},

    };

    int recordCount = sizeof(records) / sizeof(records[0]);

    char numberPlate[15];

    printf("Enter the car number plate: ");

    scanf("%s", numberPlate);

    displayCarInfo(numberPlate, records, recordCount);

    return 0;

}

**Test:**

Test Case 1: Valid Number Plate (Existing Record)

* Input: "MH12AB3456"
* Expected Output:

Enter the car number plate: MH12AB3456

Car Details for Number Plate MH12AB3456:

Owner's Name: Rajesh Patil

Driving License Number: DL1234567890123

Registration Number: MH12B1234

Chassis Number: MAHABC12345678900

Vehicle Model: Maruti Suzuki Swift

Vehicle Colour: Red

Previous Fines: 2

Test Case 2: Valid Number Plate (No Record Found)

* Input: "KA03XY1234"
* Expected Output:

Enter the car number plate: KA03XY1234

No record found for this number plate.

Test Case 3: Case Sensitivity

* Input: "mh12ab3456" (lowercase version of an existing record)
* Expected Output:
  + Since strcmp() is case-sensitive, the code as written would treat this as not matching.
  + Expected Output:

Enter the car number plate: mh12ab3456

No record found for this number plate.

Test Case 4: Special Characters in Input

* Input: "MH12@B3456"
* Expected Output:
  + The current code would not explicitly handle special characters, so it would treat this input as a non-match:

Enter the car number plate: MH12@B3456

No record found for this number plate.

Test Case 5: Input Length Exceeds Limit

* Input: "MH12AB34567890123" (more than 15 characters)
* Expected Output:
  + scanf will only read up to the limit specified (15 characters), so the input will be truncated.
  + Assuming only "MH12AB345678901" is read, the output might look like this:

Enter the car number plate: MH12AB34567890123

No record found for this number plate**.**

**OUTPUT:**

Enter the car number plate: MH12AB3456

Car Details for Number Plate MH12AB3456:

Owner's Name: Rajesh Patil

Driving License Number: DL1234567890123

Registration Number: MH12B1234

Chassis Number: MAHABC12345678900

Vehicle Model: Maruti suzuki Swift

Vehicle Color: Red

Previous Fines: 2

**IMPLEMENTATION:**

This code can be implemented in various applications where vehicle information needs to be stored and retrieved based on a car's number plate. Here are some potential use cases:

**1. Traffic Management Systems**

* The code can be a part of traffic management or law enforcement systems to quickly access vehicle and owner details, helping authorities verify records, check previous fines, and validate ownership.

**2. Parking Management Solutions**

* In parking facilities, such as malls, offices, or gated communities, this code can be used to verify authorized vehicles or retrieve vehicle information for registered users.

**3. Vehicle Service Centres**

* Service centres can use this system to retrieve customer details, vehicle information, and service history by entering the number plate, making customer interaction smoother and efficient.

**4. Toll Collection and Verification**

* Toll booths can integrate this system to check the number plate against a database of registered vehicles, quickly verifying customer details and previous toll payments or fines.

**5. Government Vehicle Registration Databases**

* This code could be a part of a government database system for checking vehicle registration and license details in a simplified local system or prototype.

**6. Fleet Management**

* Companies with vehicle fleets (e.g., logistics, delivery services) could use this code to maintain records for all vehicles, allowing quick lookup of details for each vehicle in the fleet, including fines or other records.

In a production environment, this code would likely be part of a larger application with a more extensive database instead of a static array. It would also have secure access and be integrated with user interfaces or back-end systems for scalable deployment.

GITHUB LINK:

<https://github.com/swaradadeshpande/VEHICLE-DETAILS-LOOKUP/upload/main>