#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define MAX\_CARS 100

#define MAX\_LENGTH 50

// Structure to hold car registration details

typedef struct

{

char carNumber[MAX\_LENGTH];

char owner[MAX\_LENGTH];

char phoneNumber[MAX\_LENGTH];

char location[MAX\_LENGTH];

} CarRegistration;

// Function to search for car details (case insensitive)

void searchCar(CarRegistration cars[], int count, const char \*carNumber)

{

for (int i = 0; i < count; i++)

{

// Case-insensitive comparison of car numbers

if (strcasecmp(cars[i].carNumber, carNumber) == 0)

{

printf("Car Number: %s\n", cars[i].carNumber);

printf("Owner: %s\n", cars[i].owner);

printf("Phone Number: %s\n", cars[i].phoneNumber);

printf("Location: %s\n", cars[i].location);

return;

}

}

printf("Car number %s is not registered in the database.\n", carNumber);

}

// Function to register a new car

void registerCar(CarRegistration cars[], int \*count)

{

if (\*count >= MAX\_CARS)

{

printf("Error: Cannot register more cars, database is full.\n");

return;

}

// Get car details from the user

printf("Enter car registration number: ");

fgets(cars[\*count].carNumber, MAX\_LENGTH, stdin);

cars[\*count].carNumber[strcspn(cars[\*count].carNumber, "\n")] = '\0'; // Remove newline

printf("Enter owner's name: ");

fgets(cars[\*count].owner, MAX\_LENGTH, stdin);

cars[\*count].owner[strcspn(cars[\*count].owner, "\n")] = '\0'; // Remove newline

printf("Enter owner's phone number: ");

fgets(cars[\*count].phoneNumber, MAX\_LENGTH, stdin);

cars[\*count].phoneNumber[strcspn(cars[\*count].phoneNumber, "\n")] = '\0'; // Remove newline

printf("Enter car location: ");

fgets(cars[\*count].location, MAX\_LENGTH, stdin);

cars[\*count].location[strcspn(cars[\*count].location, "\n")] = '\0'; // Remove newline

(\*count)++; // Increment the count of registered cars

printf("Car registered successfully!\n");

}

// Function to clear all car registration data

void clearCarData(CarRegistration cars[], int \*count)

{

\*count = 0; // Set count to 0, effectively clearing all registered cars

// Optionally, clear each entry in the cars array

for (int i = 0; i < MAX\_CARS; i++)

{

memset(cars[i].carNumber, 0, MAX\_LENGTH);

memset(cars[i].owner, 0, MAX\_LENGTH);

memset(cars[i].phoneNumber, 0, MAX\_LENGTH);

memset(cars[i].location, 0, MAX\_LENGTH);

}

printf("All car registration data has been cleared.\n");

}

int main()

{

// Initialize an empty cars array with no data

CarRegistration cars[MAX\_CARS] = {};

int count = 0; // Start with no cars registered

int choice;

char searchNumber[MAX\_LENGTH];

while (1)

{

printf("\n--- Car Registration System ---\n");

printf("1. Search for a car\n");

printf("2. Register a new car\n");

printf("3. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

getchar(); // To consume the newline left by scanf

switch (choice)

{

case 1:

// Search for a car

printf("Enter car number to search: ");

fgets(searchNumber, sizeof(searchNumber), stdin);

searchNumber[strcspn(searchNumber, "\n")] = '\0'; // Remove newline

// Validate input

if (strlen(searchNumber) == 0)

{

printf("Error: Invalid input. Please enter a valid car number.\n");

}

else

{

searchCar(cars, count, searchNumber);

}

break;

case 2:

// Register a new car

registerCar(cars, &count);

break;

case 3:

// Exit

printf("Exiting the system. Goodbye!\n");

return 0;

default:

printf("Invalid choice, please try again.\n");

break;

}

}

return 0;

}