CAR SALES PROGRAM

#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
#include <string.h>  
  
#define TOTAL\_CARS 5  
#define FILENAME "sales\_data.txt"  
#define ENCRYPTION\_KEY 7  
  
int totalSales = 0;  
  
// Structure to hold sale data  
typedef struct {  
 char \*customerName;  
 int customerAge;  
 float totalPrice;  
 int discountGiven;  
 float discountValue;  
 int numberOfCars;  
 char \*dateOfPurchase;  
 int feedbackRating;  
 char \*feedbackComment;  
} Sale;  
  
// Function prototypes  
void viewCarsStock();  
void buyCars();  
void viewSalesData();  
void customerFeedback();  
void saveSalesDataToFile();  
void loadSalesDataFromFile();  
char \*encryptDecrypt(const char \*input);  
void displayCustomerPurchaseSummary(); // New function declaration  
  
// Arrays to store car information  
char \*carModels[TOTAL\_CARS] = {"Toyota Camry", "Honda Civic", "Ford Mustang", "Chevrolet Silverado", "Tesla Model 3"};  
int carPrices[TOTAL\_CARS] = {25000, 22000, 35000, 38000, 45000};  
int carYears[TOTAL\_CARS] = {2022, 2023, 2022, 2024, 2023};  
int carAmounts[TOTAL\_CARS] = {10, 15, 8, 5, 3}; // Assuming initial amounts of cars  
  
// Array to store sale data  
Sale \*salesData = NULL;  
  
// Customer purchase summary variables  
char customerNameSummary[500] = "";  
char carModelSummary[500] = "";  
char dateSummary[500] = "";  
float totalPriceSummary = 0.0;  
int numberOfCarsSummary = 0;  
  
int main() {  
 int choice;  
 char firstName[50], lastName[50];  
  
 // Seed random number generator  
 srand(time(NULL));  
  
 // Load previous sales data from file  
 loadSalesDataFromFile();  
  
 // Get user's first and last names  
 printf("Welcome to Velocity Auto Sales office!\n");  
 printf("Please enter your first name: ");  
 scanf("%s", firstName);  
  
 printf("Please enter your last name: ");  
 scanf("%s", lastName);  
  
 // Program  
 printf("\nHello, %s %s!\n", firstName, lastName);  
  
 do {  
 // Display menu  
 printf("\nMenu:\n");  
 printf("1. View Cars Stock\n");  
 printf("2. Buy Cars\n");  
 printf("3. View Sales Data\n");  
 printf("4. Customer Feedback\n");  
 printf("5. Customer Purchase Summary\n"); // New menu option  
 printf("6. Exit\n");  
 printf("Enter your choice: ");  
  
 // Get user choice  
 if (scanf("%d", &choice) != 1) {  
 printf("Invalid input.\n");  
 // Clear input buffer  
 while (getchar() != '\n');  
 continue;  
 }  
  
 // Perform action based on user choice  
 switch (choice) {  
 case 1:  
 viewCarsStock();  
 break;  
 case 2:  
 buyCars();  
 break;  
 case 3:  
 viewSalesData();  
 break;  
 case 4:  
 customerFeedback();  
 break;  
 case 5:  
 displayCustomerPurchaseSummary(); // Call new function  
 break;  
 case 6:  
 // Save sales data to file before exiting  
 saveSalesDataToFile();  
 printf("Exiting program.\n");  
 break;  
 default:  
 printf("Invalid choice. Please enter a number between 1 and 6.\n");  
 break;  
 }  
 } while (choice != 6);  
  
 // Free dynamically allocated memory  
 for (int i = 0; i < totalSales; ++i) {  
 free(salesData[i].customerName);  
 free(salesData[i].dateOfPurchase);  
 free(salesData[i].feedbackComment);  
 }  
 free(salesData);  
  
 return 0;  
}  
  
// Function to view cars stock  
void viewCarsStock() {  
 printf("\nViewing Cars Stock:\n");  
 printf("| %-40s | %-10s | %-4s | %-15s |\n", "Model", "Price", "Year", "Amount Remaining");  
 printf("|------------------------------------------|------------|------|-----------------|\n");  
 for (int i = 0; i < TOTAL\_CARS; i++) {  
 printf("| %-40s | $%-9d | %-4d | %-15d |\n", carModels[i], carPrices[i], carYears[i], carAmounts[i]);  
 }  
}  
  
// Function to view sales data  
void viewSalesData() {  
 printf("\nViewing Sales Data:\n");  
 printf("+--------------------------------------------------------------------------------------------------------------------+\n");  
 printf("| Customer Name | Model | Age | Total Price | Discount Given | Discount Value | Date of Purchase |\n");  
 printf("+--------------------------------------------------------------------------------------------------------------------+\n");  
 for (int i = 0; i < totalSales; i++) {  
 printf("| %-20s | %-20s | %3d | $%10.2f | %-14s | $%10.2f | %s |\n",  
 salesData[i].customerName,  
 carModels[salesData[i].numberOfCars - 1], // Corrected to fetch the purchased car model  
 salesData[i].customerAge,  
 salesData[i].totalPrice,  
 (salesData[i].discountGiven == 1) ? "Yes" : "No",  
 salesData[i].discountValue,  
 salesData[i].dateOfPurchase);  
 }  
 printf("+--------------------------------------------------------------------------------------------------------------------+\n");  
}  
  
// Function to receive customer feedback  
void customerFeedback() {  
 printf("\nCustomer Feedback:\n");  
 printf("+-----------------------------------------------------------------------------+\n");  
 printf("| Customer Name | Model | Rating | Comment |\n");  
 printf("+-----------------------------------------------------------------------------+\n");  
 for (int i = 0; i < totalSales; i++) {  
 if (salesData[i].feedbackRating != 0) {  
 printf("| %-20s | %-20s | %6d | %-20s |\n",  
 salesData[i].customerName,  
 carModels[salesData[i].numberOfCars - 1], // Assuming carModels is zero-indexed  
 salesData[i].feedbackRating,  
 salesData[i].feedbackComment);  
 }  
 }  
 printf("+-----------------------------------------------------------------------------+\n");  
}  
  
// Function to buy cars  
void buyCars() {  
 int carChoice, age, quantity;  
 unsigned short giveDiscount = 0;  
  
 // Display car options  
 viewCarsStock();  
  
 // Get user's information  
 char tempName[100], tempDate[20], tempComment[100];  
 printf("\nEnter your name: ");  
 scanf("%s", tempName);  
  
 printf("What is your age? ");  
 scanf("%d", &age);  
  
 if (age >= 25 && age <= 75) {  
 printf("Congratulations!!! You will be getting a discount.\n");  
 giveDiscount = 1;  
 } else {  
 printf("Sorry, we don't have any discount for you.\n");  
 }  
  
 // Get car choice  
 printf("Enter the car number you want to buy: ");  
 scanf("%d", &carChoice);  
  
 // Check if the selected car is available  
 if (carAmounts[carChoice - 1] == 0) {  
 printf("Sorry, the selected car is out of stock.\n");  
 return;  
 }  
  
 // Get quantity of cars to buy  
 printf("How many cars do you want to buy? ");  
 scanf("%d", &quantity);  
  
 // Check if requested quantity is available  
 if (quantity > carAmounts[carChoice - 1]) {  
 printf("Sorry, we only have %d cars available for %s.\n", carAmounts[carChoice - 1], carModels[carChoice - 1]);  
 return;  
 }  
  
 // Allocate memory for customer name and date of purchase  
 salesData = realloc(salesData, (totalSales + 1) \* sizeof(Sale));  
 if (salesData == NULL) {  
 printf("Memory allocation failed. Exiting program.\n");  
 exit(1);  
 }  
 salesData[totalSales].customerName = strdup(tempName);  
 if (salesData[totalSales].customerName == NULL) {  
 printf("Memory allocation failed. Exiting program.\n");  
 exit(1);  
 }  
 salesData[totalSales].dateOfPurchase = malloc(20);  
 if (salesData[totalSales].dateOfPurchase == NULL) {  
 printf("Memory allocation failed. Exiting program.\n");  
 exit(1);  
 }  
 salesData[totalSales].feedbackComment = malloc(100);  
 if (salesData[totalSales].feedbackComment == NULL) {  
 printf("Memory allocation failed. Exiting program.\n");  
 exit(1);  
 }  
  
 // Get current date and time  
 time\_t now;  
 struct tm \*local;  
 now = time(NULL);  
 local = localtime(&now);  
 strftime(tempDate, 20, "%Y-%m-%d", local);  
  
 // Copy customer name and date of purchase  
 salesData[totalSales].dateOfPurchase = strdup(tempDate);  
  
 // Calculate total price based on the selected car and quantity  
 float totalPrice = carPrices[carChoice - 1] \* quantity;  
  
 // Apply discount if applicable  
 if (giveDiscount)  
 totalPrice \*= (100 - 20) / 100.0;  
  
 // Record the sale data  
 salesData[totalSales].customerAge = age;  
 salesData[totalSales].totalPrice = totalPrice;  
 salesData[totalSales].discountGiven = giveDiscount;  
 if (giveDiscount)  
 salesData[totalSales].discountValue = 20;  
 else  
 salesData[totalSales].discountValue = 0;  
 salesData[totalSales].numberOfCars = quantity;  
  
 // Get customer feedback  
 printf("\nPlease rate your experience (1-5 stars): ");  
 scanf("%d", &salesData[totalSales].feedbackRating);  
 printf("Leave a short comment: ");  
 getchar(); // Consume the newline character left in the buffer  
 fgets(tempComment, sizeof(tempComment), stdin);  
 tempComment[strcspn(tempComment, "\n")] = '\0'; // Remove newline character  
 salesData[totalSales].feedbackComment = strdup(tempComment);  
  
 // Update customer purchase summary  
 strcat(customerNameSummary, tempName);  
 strcat(customerNameSummary, ", ");  
 strcat(carModelSummary, carModels[carChoice - 1]);  
 strcat(carModelSummary, ", ");  
 strcat(dateSummary, tempDate);  
 strcat(dateSummary, ", ");  
 totalPriceSummary += totalPrice;  
 numberOfCarsSummary += quantity;  
  
 totalSales++;  
  
 // Update remaining amount of the selected car model  
 carAmounts[carChoice - 1] -= quantity;  
  
 // Inform the user about the purchase  
 printf("\nThank you, %s!\n", salesData[totalSales - 1].customerName);  
 printf("You are purchasing %d %s, %d.\n", quantity, carModels[carChoice - 1], carYears[carChoice - 1]);  
 printf("Total cost is $%.2f.\n", totalPrice);  
 printf("\nThere are %d cars remaining for %s.\n", carAmounts[carChoice - 1], carModels[carChoice - 1]);  
 printf("\nHave a great day!!!\n");  
}  
  
// Function to save sales data to file  
void saveSalesDataToFile() {  
 FILE \*file = fopen(FILENAME, "wb");  
 if (file == NULL) {  
 printf("Error opening file for writing.\n");  
 return;  
 }  
  
 // Encrypt and write sales data to file  
 for (int i = 0; i < totalSales; i++) {  
 char \*encryptedName = encryptDecrypt(salesData[i].customerName);  
 char \*encryptedDate = encryptDecrypt(salesData[i].dateOfPurchase);  
 char \*encryptedComment = encryptDecrypt(salesData[i].feedbackComment);  
 fwrite(encryptedName, sizeof(char), strlen(encryptedName), file);  
 fwrite(&salesData[i].customerAge, sizeof(int), 1, file);  
 fwrite(&salesData[i].totalPrice, sizeof(float), 1, file);  
 fwrite(&salesData[i].discountGiven, sizeof(int), 1, file);  
 fwrite(&salesData[i].discountValue, sizeof(float), 1, file);  
 fwrite(&salesData[i].numberOfCars, sizeof(int), 1, file);  
 fwrite(encryptedDate, sizeof(char), strlen(encryptedDate), file);  
 fwrite(&salesData[i].feedbackRating, sizeof(int), 1, file);  
 fwrite(encryptedComment, sizeof(char), strlen(encryptedComment), file);  
 free(encryptedName);  
 free(encryptedDate);  
 free(encryptedComment);  
 }  
  
 fclose(file);  
}  
  
// Function to load sales data from file  
void loadSalesDataFromFile() {  
 FILE \*file = fopen(FILENAME, "rb");  
 if (file == NULL) {  
 printf("No previous sales data found.\n");  
 return;  
 }  
  
 // Read and decrypt sales data from file  
 while (1) {  
 Sale sale;  
 char tempName[100], tempDate[20], tempComment[100];  
 fread(tempName, sizeof(char), 100, file);  
 if (feof(file)) break;  
 fread(&sale.customerAge, sizeof(int), 1, file);  
 fread(&sale.totalPrice, sizeof(float), 1, file);  
 fread(&sale.discountGiven, sizeof(int), 1, file);  
 fread(&sale.discountValue, sizeof(float), 1, file);  
 fread(&sale.numberOfCars, sizeof(int), 1, file);  
 fread(tempDate, sizeof(char), 20, file);  
 fread(&sale.feedbackRating, sizeof(int), 1, file);  
 fread(tempComment, sizeof(char), 100, file);  
  
 sale.customerName = encryptDecrypt(tempName);  
 sale.dateOfPurchase = encryptDecrypt(tempDate);  
 sale.feedbackComment = encryptDecrypt(tempComment);  
  
 // Reallocate memory for salesData array  
 salesData = realloc(salesData, (totalSales + 1) \* sizeof(Sale));  
 if (salesData == NULL) {  
 printf("Memory allocation failed. Exiting program.\n");  
 exit(1);  
 }  
  
 salesData[totalSales] = sale;  
 totalSales++;  
 }  
  
 fclose(file);  
}  
  
// Function to encrypt/decrypt a string using a simple XOR cipher  
char \*encryptDecrypt(const char \*input) {  
 char \*output = strdup(input);  
 for (int i = 0; i < strlen(input); i++) {  
 output[i] = input[i] ^ ENCRYPTION\_KEY;  
 }  
 return output;  
}  
  
// Function to display customer purchase summary  
void displayCustomerPurchaseSummary() {  
 printf("\nCustomer Purchase Summary:\n");  
 printf("Name: %s\n", customerNameSummary);  
 printf("Car Model: %s\n", carModelSummary);  
 printf("Date: %s\n", dateSummary);  
 printf("Total Price: $%.2f\n", totalPriceSummary);  
 printf("Number of Cars Purchased: %d\n", numberOfCarsSummary);  
}