

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering Semester: (Fall, Year:2024), B.Sc. in CSE (Day)**

**Assignment NO – 01**

**Course Title: Data Structure Course Code: CSE205 Section: D2**

**Submission Date : 18/09/2024**

**Course Teacher’s Name: Prof. Dr. Saiful Azad**

**Name: Rukonuzzaman Topu**

**ID : 232002280**

# Code:

#include <stdio.h>

#include <string.h>

// Define the structure for PropertySale, which contains details about a property sale.

typedef struct {

int uid;

char address[100];

int zip;

int size;

int year;

int price;

} PropertySale;

// Define the structure for SalesDatabase, which stores multiple sales and the count of sales.

typedef struct {

PropertySale sales[100];

int salesCount;

} SalesDatabase;

// Function declarations

void Sales(SalesDatabase \*db); // Insert a new sale into the database

void Erase(SalesDatabase \*db); // Delete a sale from the database based on UID

void Search(SalesDatabase \*db); // Search for a sale in the database by UID

void PrintDB(SalesDatabase \*db); // Print all sales in the database

void GetZIP(SalesDatabase \*db); // Get the ZIP code of a sale based on UID

void GetPrice(SalesDatabase \*db); // Get the price of a sale based on UID

int SalesCount(SalesDatabase \*db); // Get the total number of sales in the database

float AveragePrice(SalesDatabase \*db); // Calculate the average price of all sales

int main() {

SalesDatabase db; // Create a new sales database

db.salesCount = 0; // Initialize the sales count to 0 (empty database)

int choice;

// Main loop to display the menu and get the user's choice

while (1) {

printf("\n1. Insert new flat sale\n2. Delete flat sale\n3. Search flat sale\n4. Print all sales\n5. Get ZIP code\n6. Get Price\n7. Average Price\n8. Total Sales\n9. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

// Process the user's choice

switch (choice) {

case 1:

Sales(&db);

break;

case 2:

Erase(&db);

break;

case 3:

Search(&db);

break;

case 4:

PrintDB(&db);

break;

case 5:

GetZIP(&db);

break;

case 6:

GetPrice(&db);

break;

case 7:

printf("\nAverage price: %.2f\n", AveragePrice(&db));

break;

case 8:

printf("\nTotal sales: %d\n", SalesCount(&db));

break;

case 9:

return 0; // Exit the program

default:

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

}

}

return 0;

}

// Function to insert a new sale into the database

void Sales(SalesDatabase \*db) {

// Check if the database is full

if (db->salesCount >= 100) {

printf("Database is full!\n");

return;

}

PropertySale new\_sale; // Create a new sale record

// Get the details of the new sale from the user

printf("Enter UID: ");

scanf("%d", &new\_sale.uid);

printf("Enter Address: ");

scanf(" %[^\n]s", new\_sale.address); // Read the entire line for the address

printf("Enter ZIP: ");

scanf("%d", &new\_sale.zip);

printf("Enter size of Square Feet: ");

scanf("%d", &new\_sale.size);

printf("Enter construction year: ");

scanf("%d", &new\_sale.year);

printf("Enter price in taka: ");

scanf("%d", &new\_sale.price);

// Insert the new sale into the database

db->sales[db->salesCount] = new\_sale;

db->salesCount++; // Increment the sales count

printf("Sale added successfully.\n");

}

// Function to delete a sale from the database based on UID

void Erase(SalesDatabase \*db) {

int uid;

printf("Enter the UID to delete: ");

scanf("%d", &uid);

int found = 0; // Variable to check if the sale was found

// Loop through the sales to find the sale with the given UID

for (int i = 0; i < db->salesCount; i++) {

if (db->sales[i].uid == uid) {

found = 1;

// Shift the remaining sales to "delete" the entry

for (int j = i; j < db->salesCount - 1; j++) {

db->sales[j] = db->sales[j + 1];

}

db->salesCount--; // Decrease the sales count

printf("Sale with UID %d deleted successfully.\n", uid);

break;

}

}

// If the sale was not found, notify the user

if (!found) {

printf("Sale with UID %d not found.\n", uid);

}

}

// Function to search for a sale in the database by UID

void Search(SalesDatabase \*db) {

int uid;

printf("Enter the UID to search: ");

scanf("%d", &uid);

// Loop through the sales to find the sale with the given UID

for (int i = 0; i < db->salesCount; i++) {

if (db->sales[i].uid == uid) {

// Print the details of the sale if found

printf("Sale found: UID=%d, Address=%s, ZIP=%d, Size=%d, Year=%d, Price=%d\n",

db->sales[i].uid, db->sales[i].address, db->sales[i].zip,

db->sales[i].size, db->sales[i].year, db->sales[i].price);

return;

}

}

printf("Sale with UID %d not found.\n", uid);

}

// Function to print all sales in the database

void PrintDB(SalesDatabase \*db) {

// Check if the database is empty

if (db->salesCount == 0) {

printf("No sales available.\n");

return;

}

// Loop through the sales and print each one

for (int i = 0; i < db->salesCount; i++) {

printf("UID: %d, Address: %s, ZIP: %d, Size: %d, Year: %d, Price: %d\n",

db->sales[i].uid, db->sales[i].address, db->sales[i].zip,

db->sales[i].size, db->sales[i].year, db->sales[i].price);

}

}

// Function to get the ZIP code of a sale by UID

void GetZIP(SalesDatabase \*db) {

int uid;

printf("Enter UID: ");

scanf("%d", &uid);

// Loop through the sales to find the sale with the given UID

for (int i = 0; i < db->salesCount; i++) {

if (db->sales[i].uid == uid) {

// Print the ZIP code of the sale if found

printf("ZIP Code: %d\n", db->sales[i].zip);

return;

}

}

printf("Sale with UID %d not found.\n", uid);

}

// Function to get the price of a sale by UID

void GetPrice(SalesDatabase \*db) {

int uid;

printf("Enter UID: ");

scanf("%d", &uid);

// Loop through the sales to find the sale with the given UID

for (int i = 0; i < db->salesCount; i++) {

if (db->sales[i].uid == uid) {

// Print the price of the sale if found

printf("Price: %d\n", db->sales[i].price);

return;

}

}

printf("Sale with UID %d not found.\n", uid);

}

// Function to return the total number of sales in the database

int SalesCount(SalesDatabase \*db) {

return db->salesCount;

}

// Function to calculate the average price of all sales

float AveragePrice(SalesDatabase \*db) {

if (db->salesCount == 0) {

return 0.0; // Return 0 if there are no sales

}

float total\_price = 0; // Variable to hold the sum of prices

// Loop through the sales to calculate the total price

for (int i = 0; i < db->salesCount; i++) {

total\_price += db->sales[i].price;

}

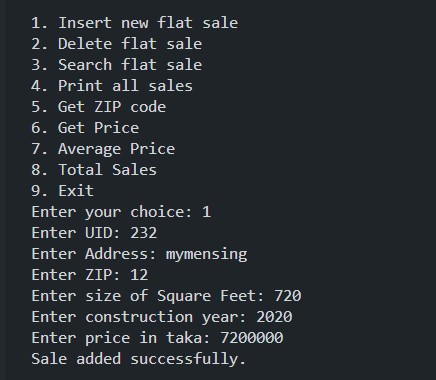
// Return the average price

return total\_price / db->salesCount;

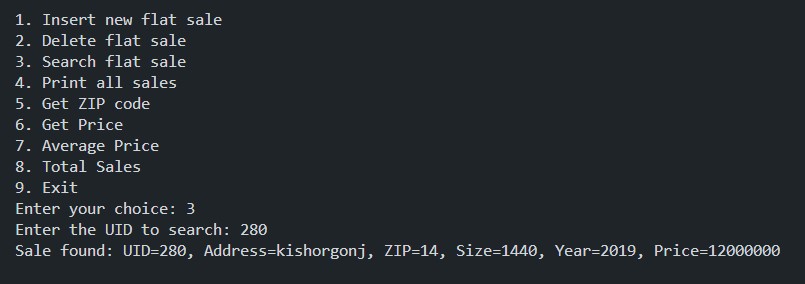
}

# Output:

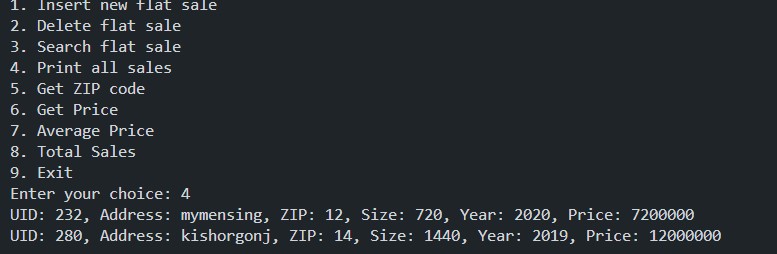
Step 1:



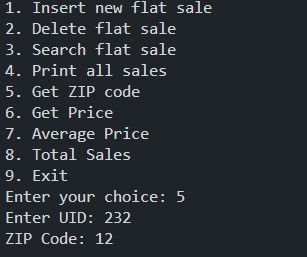
Step 2:



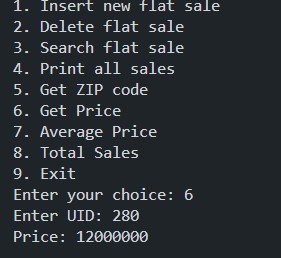
Step 3:



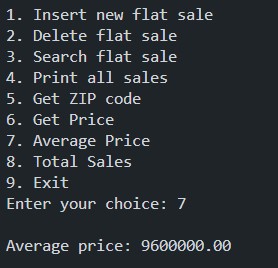
Step 4:

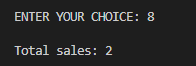


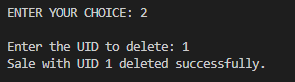
Step 5:

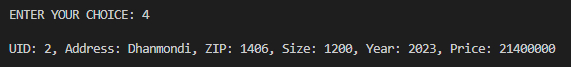


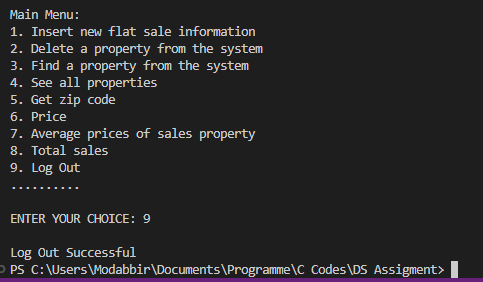
Step 6:



Step 7:

Step 11:

Step 12:

Step 13: