#include <stdio.h>

#include <string.h>

#define MAX\_ITEMS 100 // Maximum number of grocery items

#define NAME\_LENGTH 100 // Maximum length for item name

#define CATEGORY\_LENGTH 50 // Maximum length for category name

// Union to represent the stock status

typedef union {

int inStock; // 1 = in stock, 0 = out of stock

char stockText[10]; // Text representation, for example "In Stock"

} StockStatus;

// Structure to represent a grocery item

typedef struct {

char name[NAME\_LENGTH];

char category[CATEGORY\_LENGTH];

int quantity;

float price;

StockStatus status; // Use union for stock status

} GroceryItem;

// Global variables

GroceryItem inventory[MAX\_ITEMS]; // Array of grocery items

int itemCount = 0; // Number of total items in the inventory

// Defining function prototypes

void addGroceryItem();

void listGroceryItems();

void updateItemQuantity();

void removeGroceryItem();

void displayMenu();

int main() {

int choice = 0;

// A loop of the menu for the user to manage grocery item information and item availability

while (choice != 5) {

displayMenu();

printf("Enter your choice: ");

scanf("%d", &choice);

getchar(); // Remove newline character

if (choice == 1) {

addGroceryItem();

} else if (choice == 2) {

listGroceryItems();

} else if (choice == 3) {

updateItemQuantity();

} else if (choice == 4) {

removeGroceryItem();

} else if (choice == 5) {

printf("Exiting...\n");

} else {

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

}

}

return 0;

}

// Function to display the menu

void displayMenu() {

printf("\nGrocery Inventory Management System\n");

printf("1. Add Grocery Item\n");

printf("2. List All Grocery Items\n");

printf("3. Update Quantity\n");

printf("4. Remove Grocery Item\n");

printf("5. Exit\n");

}

// Function to add a new grocery item

void addGroceryItem() {

if (itemCount >= MAX\_ITEMS) {

printf("Inventory is full. Cannot add more grocery items.\n");

return;

}

GroceryItem newItem; // Declare a new variable of type GroceryItem to store details of a single grocery item

// Prompt the user for the input of item name

printf("Enter Item Name: ");

fgets(newItem.name, NAME\_LENGTH, stdin);

newItem.name[strcspn(newItem.name, "\n")] = '\0'; // Remove newline character

// Prompt the user for the input of category

printf("Enter Category: ");

fgets(newItem.category, CATEGORY\_LENGTH, stdin);

newItem.category[strcspn(newItem.category, "\n")] = '\0'; // Remove newline character

printf("Enter Quantity: "); // Prompt the user for the input of quantity of item

scanf("%d", &newItem.quantity);

printf("Enter Price: "); // Prompt the user for the input price of item

scanf("%f", &newItem.price);

getchar(); // Remove newline character

// Update the stockText field of the union to reflect the stock status

if (newItem.quantity > 0) {

strcpy(newItem.status.stockText, "In Stock");

} else {

strcpy(newItem.status.stockText, "Out of Stock");

}

// Add the new item to the inventory

inventory[itemCount++] = newItem;

printf("Grocery item added successfully!\n");

}

// Function to list all grocery items information

void listGroceryItems() {

if (itemCount == 0) {

printf("No grocery items in inventory.\n");

return;

}

printf("\nGrocery Inventory:\n");

for (int i = 0; i < itemCount; i++) {

printf("Name: %s\n", inventory[i].name);

printf("Category: %s\n", inventory[i].category);

printf("Quantity: %d\n", inventory[i].quantity);

printf("Price: %.2f\n", inventory[i].price);

printf("Status: %s\n", inventory[i].status.stockText);

printf("\n"); // Add a blank line between items

}

}

// Function to update the quantity of an item

void updateItemQuantity() {

if (itemCount == 0) {

printf("No grocery items in inventory.\n");

return;

}

char itemName[NAME\_LENGTH];

printf("Enter name of the grocery item to update quantity: "); // Prompt the user for the input of name of the item whose quantity needs to be updated

fgets(itemName, NAME\_LENGTH, stdin);

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

for (int i = 0; i < itemCount; i++) {

if (strcmp(inventory[i].name, itemName) == 0) {

// Prompt the user for the input of the new quantity of the item

printf("Enter new quantity for %s: ", inventory[i].name);

scanf("%d", &inventory[i].quantity);

getchar(); // Remove newline character

// Update the stock status based on the new quantity

if (inventory[i].quantity > 0) {

strcpy(inventory[i].status.stockText, "In Stock"); // Set the status as "In Stock"

} else {

strcpy(inventory[i].status.stockText, "Out of Stock"); // Set the status as "Out of Stock"

}

printf("Quantity updated successfully!\n");

return;

}

}

printf("Grocery item not found in inventory.\n");

}

// Function to remove a grocery item

void removeGroceryItem() {

if (itemCount == 0) {

printf("No grocery items in inventory.\n");

return;

}

char itemName[NAME\_LENGTH];

printf("Enter the name of the grocery item to remove: "); // Prompt the user for the input of the name of the item to be removed

fgets(itemName, NAME\_LENGTH, stdin);

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

for (int i = 0; i < itemCount; i++) {

if (strcmp(inventory[i].name, itemName) == 0) {

// Shift items to overwrite the removed item

for (int j = i; j < itemCount - 1; j++) {

inventory[j] = inventory[j + 1]; // Overwrite the current item with the next item

}

itemCount--; // Decrease item count

printf("Grocery item '%s' removed successfully!\n", itemName);

return;

}

}

printf("Grocery item not found in inventory.\n");

}