```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_USERS 10
#define MAX_ELEMENTS 100

// User structure
typedef struct {
    char username[20];
    char password[20];
    int isAdmin;
} User;

// Element structure
typedef struct {
    int data[MAX_ELEMENTS];
    int size;
} ElementList;

// Function prototypes
int login(User users[], int *loggedInUser);
void adminMenu(ElementList *list);
void customerMenu(ElementList *list);
void addElement(ElementList *list, int element);
void displayElements(ElementList list);
int linearSearch(ElementList list, int target);
int binarySearch(ElementList list, int target);

int main() {
    User users[MAX_USERS];
    ElementList elementList;
    int loggedInUser = -1;

    // Initialize users (username, password, isAdmin)
    strcpy(users[0].username, "admin");
    strcpy(users[0].password, "admin123");
    users[0].isAdmin = 1;

    strcpy(users[1].username, "customer");
    strcpy(users[1].password, "customer123");
    users[1].isAdmin = 0;

    // Initialize element list
```

```
    elementList.size = 0;

    while (1) {
        int choice = login(users, &loggedInUser);

        if (loggedInUser == -1) {
            printf("Login failed. Please try again.\n");
            continue;
        }

        if (users[loggedInUser].isAdmin) {
            adminMenu(&elementList);
        } else {
            customerMenu(&elementList);
        }
    }

    return 0;
}

int login(User users[], int *loggedInUser) {
    char username[20];
    char password[20];

    printf("Login\n");
    printf("Username: ");
    scanf("%s", username);
    printf("Password: ");
    scanf("%s", password);

    for (int i = 0; i < MAX_USERS; i++) {
        if (strcmp(users[i].username, username) == 0 && strcmp(users[i].password, password) ==
0) {
            *loggedInUser = i;
            return i;
        }
    }

    return -1;
}

void adminMenu(ElementList *list) {
    int choice;
    int element;
```

```
    while (1) {
        printf("\nAdmin Menu\n");
        printf("1. Add Element\n");
        printf("2. Display Elements\n");
        printf("3. Logout\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter an element to add: ");
                scanf("%d", &element);
                addElement(list, element);
                break;
            case 2:
                displayElements(*list);
                break;
            case 3:
                return;
            default:
                printf("Invalid choice. Try again.\n");
        }
    }
}

void customerMenu(ElementList *list) {
    int choice;
    int target;
    int result;

    while (1) {
        printf("\nCustomer Menu\n");
        printf("1. Linear Search\n");
        printf("2. Binary Search\n");
        printf("3. Logout\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter the element to search for: ");
                scanf("%d", &target);
                result = linearSearch(*list, target);
```

```
            if (result != -1) {
                printf("Element %d found at index %d\n", target, result);
            } else {
                printf("Element %d not found\n", target);
            }
            break;
        case 2:
            printf("Enter the element to search for: ");
            scanf("%d", &target);
            result = binarySearch(*list, target);
            if (result != -1) {
                printf("Element %d found at index %d\n", target, result);
            } else {
                printf("Element %d not found\n", target);
            }
            break;
        case 3:
            return;
        default:
            printf("Invalid choice. Try again.\n");
    }
  }
}

void addElement(ElementList *list, int element) {
  if (list->size < MAX_ELEMENTS) {
    list->data[list->size++] = element;
    printf("Element %d added successfully.\n", element);
  } else {
    printf("Element list is full. Cannot add more elements.\n");
  }
}

void displayElements(ElementList list) {
  printf("Elements: ");
  for (int i = 0; i < list.size; i++) {
    printf("%d ", list.data[i]);
  }
  printf("\n");
}

int linearSearch(ElementList list, int target) {
  for (int i = 0; i < list.size; i++) {
    if (list.data[i] == target) {
```

```
            return i;
        }
    }
    return -1;
}

int binarySearch(ElementList list, int target) {
    int left = 0;
    int right = list.size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;

        if (list.data[mid] == target) {
            return mid;
        }

        if (list.data[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }

    return -1;
}
```