#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_CANDIDATES 5

#define MAX\_NAME\_LEN 50

#define MAX\_VOTERS 100

typedef struct {

char name[MAX\_NAME\_LEN];

int votes;

} Candidate;

typedef struct {

int id;

int hasVoted;

} Voter;

Candidate candidates[MAX\_CANDIDATES];

Voter voters[MAX\_VOTERS];

int totalCandidates = 0;

int totalVoters = 0;

const char ADMIN\_USERNAME[] = "admin";

const char ADMIN\_PASSWORD[] = "1234";

// Helper to clear input buffer

void clearInputBuffer() {

int c;

while ((c = getchar()) != '\n' && c != EOF);

}

void saveCandidates() {

FILE \*fp = fopen("candidates.txt", "w");

if (!fp) {

printf("Error saving candidates.\n");

return;

}

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

fprintf(fp, "%s %d\n", candidates[i].name, candidates[i].votes);

}

fclose(fp);

}

void loadCandidates() {

FILE \*fp = fopen("candidates.txt", "r");

if (!fp) return;

totalCandidates = 0;

while (fscanf(fp, "%s %d", candidates[totalCandidates].name, &candidates[totalCandidates].votes) == 2) {

totalCandidates++;

if (totalCandidates >= MAX\_CANDIDATES) break;

}

fclose(fp);

}

void saveVoters() {

FILE \*fp = fopen("voters.txt", "w");

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

fprintf(fp, "%d %d\n", voters[i].id, voters[i].hasVoted);

}

fclose(fp);

}

void loadVoters() {

FILE \*fp = fopen("voters.txt", "r");

if (!fp) return;

totalVoters = 0;

while (fscanf(fp, "%d %d", &voters[totalVoters].id, &voters[totalVoters].hasVoted) == 2) {

totalVoters++;

if (totalVoters >= MAX\_VOTERS) break;

}

fclose(fp);

}

void addCandidates() {

printf("Enter number of candidates (1 to %d): ", MAX\_CANDIDATES);

scanf("%d", &totalCandidates);

clearInputBuffer();

if (totalCandidates < 1 || totalCandidates > MAX\_CANDIDATES) {

printf("Invalid number of candidates.\n");

totalCandidates = 0;

return;

}

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

printf("Enter name of candidate %d: ", i + 1);

fgets(candidates[i].name, MAX\_NAME\_LEN, stdin);

candidates[i].name[strcspn(candidates[i].name, "\n")] = 0; // remove newline

candidates[i].votes = 0;

}

saveCandidates();

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

}

int hasAlreadyVoted(int id) {

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

if (voters[i].id == id) {

return voters[i].hasVoted;

}

}

return 0;

}

void markVoted(int id) {

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

if (voters[i].id == id) {

voters[i].hasVoted = 1;

saveVoters();

return;

}

}

if (totalVoters < MAX\_VOTERS) {

voters[totalVoters].id = id;

voters[totalVoters].hasVoted = 1;

totalVoters++;

saveVoters();

} else {

printf("Voter limit reached.\n");

}

}

void castVote() {

int id;

printf("Enter your Voter ID (number): ");

scanf("%d", &id);

clearInputBuffer();

if (hasAlreadyVoted(id)) {

printf("You have already voted!\n");

return;

}

if (totalCandidates == 0) {

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

return;

}

printf("\n--- Voting Panel ---\n");

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

printf("%d. %s\n", i + 1, candidates[i].name);

}

int choice;

printf("Enter your choice (1 to %d): ", totalCandidates);

scanf("%d", &choice);

clearInputBuffer();

if (choice >= 1 && choice <= totalCandidates) {

candidates[choice - 1].votes++;

markVoted(id);

saveCandidates();

printf("Vote cast successfully.\n");

} else {

printf("Invalid choice.\n");

}

}

void showResults() {

if (totalCandidates == 0) {

printf("No candidates to display.\n");

return;

}

printf("\n--- Voting Results ---\n");

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

printf("%s - %d votes\n", candidates[i].name, candidates[i].votes);

}

}

void showWinner() {

if (totalCandidates == 0) {

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

return;

}

int maxVotes = -1;

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

if (candidates[i].votes > maxVotes) {

maxVotes = candidates[i].votes;

}

}

if (maxVotes == 0) {

printf("No votes have been cast yet.\n");

return;

}

printf("\n--- Winner(s) ---\n");

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

if (candidates[i].votes == maxVotes) {

printf("%s with %d votes\n", candidates[i].name, maxVotes);

}

}

}

int adminLogin() {

char username[50], password[50];

printf("Enter admin username: ");

scanf("%s", username);

printf("Enter password: ");

scanf("%s", password);

clearInputBuffer();

if (strcmp(username, ADMIN\_USERNAME) == 0 && strcmp(password, ADMIN\_PASSWORD) == 0) {

return 1;

} else {

printf("Access denied.\n");

return 0;

}

}

int main() {

int choice;

loadCandidates();

loadVoters();

while (1) {

printf("\n--- Mini Voting System ---\n");

printf("1. Admin Panel\n");

printf("2. Cast Vote\n");

printf("3. Show Results\n");

printf("4. Show Winner\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

clearInputBuffer();

switch (choice) {

case 1:

if (adminLogin()) {

addCandidates();

}

break;

case 2:

castVote();

break;

case 3:

showResults();

break;

case 4:

showWinner();

break;

case 5:

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

exit(0);

default:

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

}

}

return 0;

}