Mini Voting system

Micro-project Report

Department of Basic Science

Submitted By

Mr. Niranjan Dnyaneshwar Joshi

URN: 1024091005

Roll no:1105

Guided by

Prof. Rutuja Patil

Annasaheb Dange College of Engineering and Technology, Ashta.

1 Introduction

Voting is a crucial part of democratic systems, ensuring representation and decision-making. This project presents a simple Electronic Voting System implemented in C programming, designed to automate the voting process for three candidates. The system is user-friendly, modular, and ensures accurate vote counting and result display

2. Problem Statement

Traditional voting methods often involve manual counting, which can lead to errors and inefficiencies. This project aims to create a basic, automated voting system to:

- Simplify the voting process.
- Ensure accurate and real-time vote counting.
- Reduce human errors in result compilation.

3. Algorithm

Step 1: Start

Step 2: Initialize: Declare arrays to store candidate names and votes.

Step 3: Input Candidates: Prompt the user to enter the names of three candidates.

Step 4: Display Candidates: Show the list of candidates with numbered options.

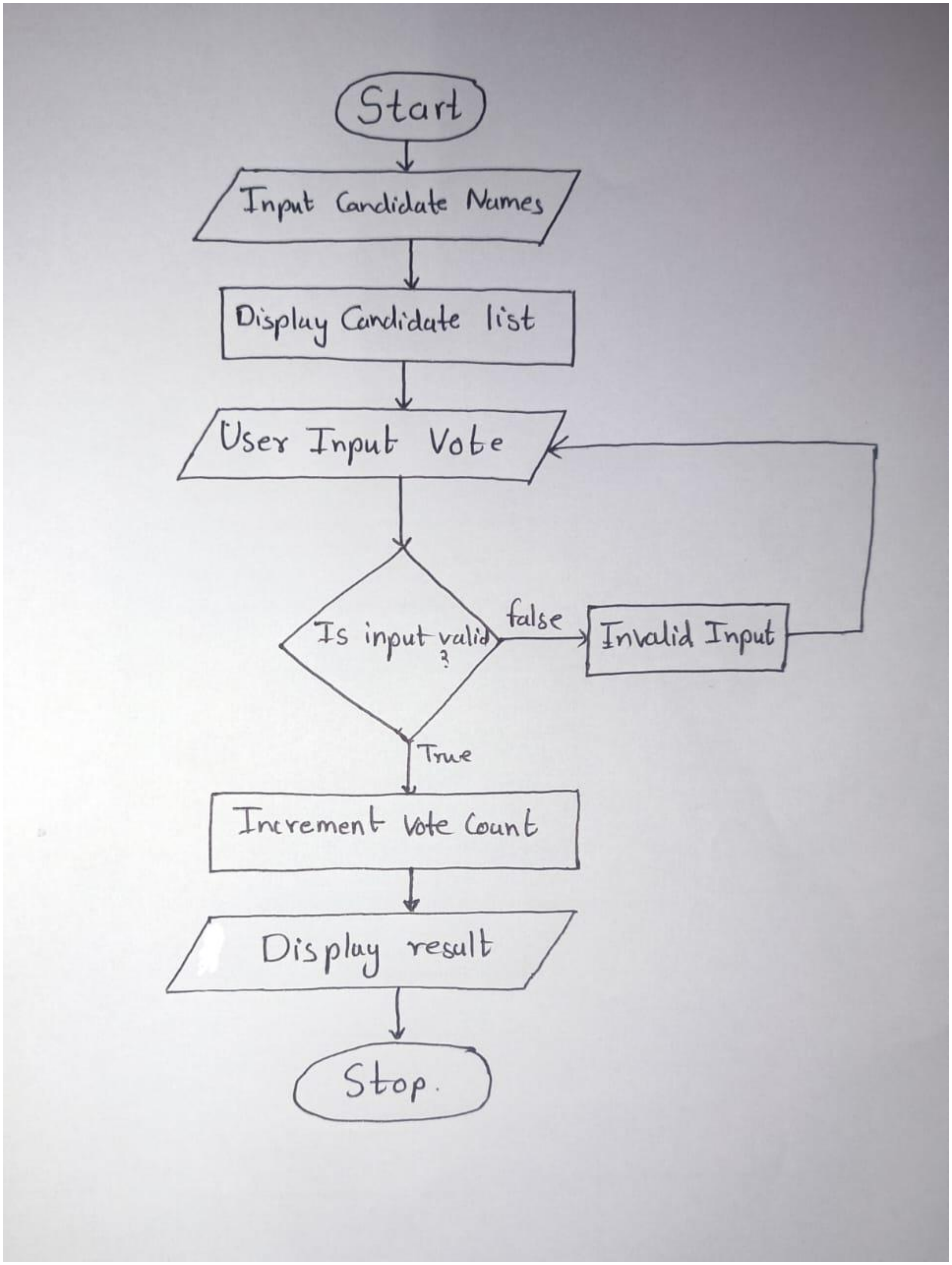
Step 5: Accept Votes: Allow users to vote by selecting a candidate or stop voting by entering 0.

Step 6: Validate and Count: Validate the user's choice and update the vote count for the chosen candidate.

Step 7: Display Results: After voting ends, show the total votes received by each candidate.

Step 8: End

4. Flowchart


Start
Input Candidate Names
Display Candidate list
User Input Vote
Is input valid ?
false
Invalid Input
True
Increment Vote Count
Display result
Stop.

5. Methodology (Used Concepts)

This project leverages the following programming concepts:

File Modularity:

main.c: Handles user interaction and the program's main flow.

voting.c: Implements core functionalities like vote counting and result display.

voting.h: Declares functions for modularity.

Arrays: Store candidate names and votes dynamically.

Functions:

display_candidates: Display the list of candidates.

cast_vote: Validate and record votes.

display_results: Display voting results.

Control Flow: Loops and conditional statements for iterative and decision-based operations.

6. Results (Output)

Input:

Favorite Teacher of The Year

Candidate Names: Rutuja Patil mam, Joshi sir , Shinde sir

Votes: 1, 2, 3, 1, 1,0

Output:

Voting Results:

Rutuja Patil mam: 3 votes

Joshi sir: 1 vote

Shinde sir: 1 vote

7. Future Scope

Enhanced Features:

- Add more candidates dynamically.
- Implement voter authentication for secure voting.
- Provide real-time vote monitoring with graphical representation.

Platform Expansion:

- Develop a GUI-based version for better usability.
- cale to support online voting systems.

8. Conclusion

The Electronic Voting System is a practical demonstration of how programming can solve real-world problems. This project efficiently automates voting and result computation, providing a solid foundation for more complex systems in the future.

9. References

- Book: Programming in C by Dennis Ritchie
- Online Resources: www.cprogramming.com
- Lecture notes.

10. Implemented Code

main.c: Handles user interaction and the program's main flow.

```
#include <stdio.h>

#include "voting.h"

int main()

{

    char candidates[3][50];

    int choice;
```

```
printf("Enter the names of 3 candidates:\n");
for (int i = 0; i < 3; i++)
{
    printf("Candidate %d: ", i + 1);
    fgets(candidates[i], sizeof(candidates[i]), stdin);
}

while (1)
{
    display_candidates(candidates);
    printf("\nEnter your vote (1-3) or 0 to stop: ");
    scanf("%d", &choice);

    if (choice == 0)
    {
        break;
    }
    cast_vote(choice);
}
```

```
    display_results(candidates);
    return 0;
}
```

voting.c: Implements core functionalities like vote counting and result display.

```
#include "voting.h"
#include <stdio.h>
#include <string.h>
int votes[3] = {0, 0, 0};
void display_candidates(char candidates[3][50])
{
    printf("Candidates:\n");
    for (int i = 0; i < 3; i++)
    {
        printf("%d. %s", i + 1, candidates[i]);
    }
}
void cast_vote(int choice)
{
    if (choice >= 1 && choice <= 3)
    {
        votes[choice - 1]++;
        printf("Vote recorded!\n");
    }
    else
    {
        printf("Invalid choice. Try again.\n");
    }
}
void display_results(char candidates[3][50])
{
    printf("\nVoting Results:\n");
    for (int i = 0; i < 3; i++)
    {
        printf("%s: %d votes\n", candidates[i], votes[i]);
    }
}
```

voting.h: Declares functions for modularity.

```
#ifndef VOTING_H
#define VOTING_H

void display_candidates(char candidates[3][50]);
void cast_vote(int choice);
void display_results(char candidates[3][50]);

#endif
```