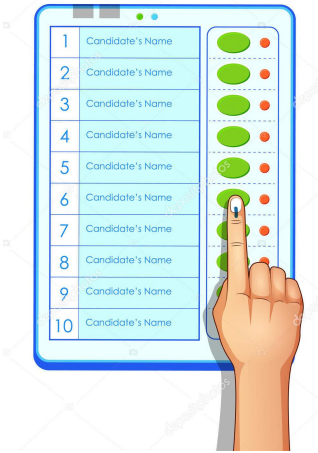
MINI VOTING SYSTEM



Abstract

C program simulates a mini voting system where users can cast votes for one of the three political parties. After all votes are cast, the program displays the election results and declares the winning party based on the highest number of votes.

- **Initialization**: Three political parties are defined, each represented by a structure that includes the party name and an initial vote count set to zero.
- **User Interaction**:The program prompts the user to enter the number of voters participating in the election.For each voter, the program allows them to cast a vote by selecting one of the three parties. The user is presented with a list of parties, and they choose one by entering the corresponding number.

- **Vote Counting:** The program keeps track of the votes each party receives. For every vote cast, the respective party's vote count is incremented.

- **Election Results:** After all votes are cast, the program displays the election results. It shows the number of votes each party received.

- **Winner Determination:** The program identifies the winning party by comparing the vote counts. The party with the highest number of votes is declared the winner.

- **Winner Announcement:** The program prints a message declaring the winning party and indicating the number of votes it received.

Resources

1. Mouse
2. Laptop
3. Printer

Source code

```
#include <stdio.h>
#include <stdlib.h>
#define NUM_PARTIES 3
// Structure to store party details
struct Party {
    char name[30];
    int votes;
};
// Function to display party names
void displayParties(struct Party parties[]) {
    printf("Available Parties:\n");
    for (int i = 0; i < NUM_PARTIES; ++i) {
        printf("%d. %s\n", i + 1, parties[i].name);
    }
}
void castVote(struct Party parties[]) {
    int choice;
    displayParties(parties);
    printf("Enter the number corresponding to your
choice: ");
    scanf("%d", &choice);
```

Source code

```
if (choice >= 1 && choice <= NUM_PARTIES) {
        parties[choice - 1].votes++;
        printf("Vote cast successfully for %s.\n", parties[choice -
1].name);
    } else {
        printf("Invalid choice. Please try again.\n");
    }
}
// Function to display election results
void displayResults(struct Party parties[]) {
    printf("\nElection Results:\n");
    for (int i = 0; i < NUM_PARTIES; ++i) {
        printf("%s: %d votes\n", parties[i].name, parties[i].votes);
    }
}
int main() {
    struct Party parties[NUM_PARTIES] = {
        {"BJP", 0},
        {"INC", 0},
        {"BRS", 0}
    };
```

Source code

```
 int voters;
    printf("Enter the number of voters: ");
    scanf("%d", &voters);

    for (int i = 0; i < voters; ++i) {
        printf("\nVoter %d:\n", i + 1);
        castVote(parties);
    }
    // Display election results
    displayResults(parties);

 int winnerIndex = 0;
    for (int i = 1; i < NUM_PARTIES; ++i) {
        if (parties[i].votes > parties[winnerIndex].votes) {
            winnerIndex = i;
        }
    }
    printf("\nThe winner is %s!\n",
parties[winnerIndex].name);
    return 0;
}
```

Output

Enter the number of voters: 5
Voter 1:
Available Parties:
1. BJP
2. INC
3. BRS
Enter the number corresponding to your choice: 1
Vote cast successfully for BJP.

Voter 2:
Available Parties:
1. BJP
2. INC
3. BRS
Enter the number corresponding to your choice: 2
Vote cast successfully for INC.

Voter 3:
Available Parties:
1. BJP
2. INC
3. BRS
Enter the number corresponding to your choice: 1
Vote cast successfully for BJP.

Output

Voter 4:
Available Parties:
1. BJP
2. INC
3. BRS
Enter the number corresponding to your choice: 1
Vote cast successfully for BJP.

Voter 5:
Available Parties:1. BJP
2. INC
3. BRS
Enter the number corresponding to your choice: 1
Vote cast successfully for BJP.

Election Results:
BJP: 4 votes
INC: 1 votes
BRS: 0 votes

The winner is BJP!