

OOP Lab Assignment 01



Muhammad Anas Ramzan

231609

Sir Dr. Ashfaq Hussain Farooqi

February 11, 2024

Voting Management system has been created as per required in lab assignment 01.It deals with management of:

1. NA Candidate   
2. PA Candidate   
3. Voters

The project consists of 3 files out of which 2 are header files (Candidate.h, Voter.h) while the other is C++ file (Voting Management System.cpp)

This project also involves file handling to store data in files and three files are created in this regard. They are:

1. NA\_Candidates.txt

2. PA \_Candidates.txt

3. Voters.txt

**Code snippets with their outputs**

**Main**

|  |
| --- |
|  |

**Main menu**

|  |
| --- |
|  |
|  |

**NA Candidate menu**

|  |
| --- |
|  |
|  |

**PA Candidate menu**

|  |
| --- |
|  |
|  |

**Voters menu**

|  |
| --- |
|  |
|  |

**Structure for NA record and PA record as well**

Same structure works for both NA record and PA record

|  |
| --- |
|  |

**Add function for NA menu and PA menu as well**

Same function works for both NA menu and PA menu

|  |  |
| --- | --- |
|  | |
|  |  |

**View function for NA menu and PA menu as well**

Same function works for both NA menu and PA menu

|  |  |
| --- | --- |
|  | |
|  |  |

**Save function**

Used in add function (for NA menu and PA menu) to save record into respective files

|  |  |
| --- | --- |
|  | |
|  |  |

**Add function for Voter menu**

|  |
| --- |
|  |
|  |

**View function for Voter menu**

|  |
| --- |
|  |
|  |

**Search function for Voter menu**

|  |  |
| --- | --- |
|  | |
|  |  |

**Save function for Voter menu**

Used in add function to save record into file

|  |
| --- |
|  |
|  |

CODE

|  |
| --- |
| Voting Management System.cpp |
| #include <iostream>  #include <fstream>  #include"Candidate.h"  #include"Voter.h"  using namespace std;  void menu();  void NA\_CandidateMenu(candidate na\_candidates[], int naEntries);  void PA\_CandidateMenu(candidate pa\_candidates[], int paEntries);  void voterMenu(voter voters[], int voterEntries);  int main()  {  cout << "Voting Management System\n";  menu();  }  void menu()  {  const int naSize = 5;  candidate na\_candidates[naSize];  ifstream fin;  fin.open("NA\_Candidates.txt");  try {  if (fin.fail())  throw runtime\_error("There is no previously added NA Candidate record\n");  }  catch (runtime\_error & e)  {  cout << e.what();  }  int naEntries = 0;  if (!fin.fail()) {  while (!fin.eof())  {  fin >> na\_candidates[naEntries].areaNo >> na\_candidates[naEntries].cnic >> na\_candidates[naEntries].name >> na\_candidates[naEntries].party;  naEntries++;  }  naEntries -= 1;  }  const int paSize = 5;  candidate pa\_candidates[paSize];  ifstream inputFile;  inputFile.open("PA\_Candidates.txt");  try {  if (inputFile.fail())  throw runtime\_error("There is no previously added PA Candidate record\n");  }  catch (runtime\_error & e)  {  cout << e.what();  }  int paEntries = 0;  if (!inputFile.fail())  {  while (!inputFile.eof())  {  inputFile >> pa\_candidates[paEntries].areaNo >> pa\_candidates[paEntries].cnic >> pa\_candidates[paEntries].name >> pa\_candidates[paEntries].party;  paEntries++;  }  paEntries -= 1;  }  const int votersSize = 5;  voter voters[votersSize];  ifstream fileIn;  fileIn.open("Voters.txt");  try {  if (fileIn.fail())  {  throw runtime\_error("There is no previously added voter record\n");  }  }  catch (runtime\_error & e)  {  cout << e.what();  }  int voterEntries = 0;  if (!fileIn.fail())  {  while (!fileIn.eof())  {  fileIn >> voters[voterEntries].cnic >> voters[voterEntries].na >> voters[voterEntries].pa >> voters[voterEntries].block >> voters[voterEntries].pollingStation;  voterEntries++;  }  voterEntries -= 1;  }  while (1)  {  cout << "1.Manage N.A Candidate\n2.Manage P.A Candidate\n3.Manage Voter Detail\n4.Exit\n";  int choice;  cin >> choice;  switch (choice)  {  case 1:  cout << "N.A Candidate Management System\n";  NA\_CandidateMenu(na\_candidates, naEntries);  break;  case 2:  cout << "P.A Candidate Management System\n";  PA\_CandidateMenu(pa\_candidates, paEntries);  break;  case 3:  cout << "Voter Details Management System\n";  voterMenu(voters, voterEntries);  break;  case 4:  cout << "Program Ends\n";  exit(0);  break;  default:  cout << "You have entered invalid option!\n";  }  }  }  void NA\_CandidateMenu(candidate na\_candidates[], int naEntries)  {  while (1)  {  cout << "1.Add\n2.View\n3.Back to main menu\n";  int choice;  cin >> choice;  switch (choice)  {  case 1:  cout << "Add N.A Candidate Details(Area = N.A Number)\n";  add(na\_candidates, naEntries, 1);  break;  case 2:  cout << "Viewing N.A Candidates' Details(Area = N.A Number)\n";  view(na\_candidates, naEntries);  break;  case 3:  cout << "Going back to main menu!\n";  menu();  break;  default:  cout << "You have entered invalid option!\n";  }  }  }  void PA\_CandidateMenu(candidate pa\_candidates[], int paEntries)  {  while (1)  {  cout << "1.Add\n2.View\n3.Back to main menu\n";  int choice;  cin >> choice;  switch (choice)  {  case 1:  cout << "Add P.A Candidate Details(Area = P.A Number)\n";  add(pa\_candidates, paEntries, 2);  break;  case 2:  cout << "Viewing P.A Candidates' Details(Area = P.A Number)\n";  view(pa\_candidates, paEntries);  break;  case 3:  cout << "Going back to main menu!\n";  menu();  break;  default:  cout << "You have entered invalid option!\n";  }  }  }  void voterMenu(voter voters[], int voterEntries)  {  while (1)  {  cout << "1.Add\n2.View\n3.Search\n4.Back to main menu\n";  int choice;  cin >> choice;  switch (choice)  {  case 1:  cout << "Add Voter Details\n";  add(voters, voterEntries);  break;  case 2:  cout << "Viewing Voter Details\n";  view(voters, voterEntries);  break;  case 3:  cout << "Search the Voter\n";  cout << "Enter voter's CNIC:";  uint64\_t searchCNIC;  cin >> searchCNIC;  bool flag;  flag = search(voters, voterEntries, searchCNIC);  if (flag)  {  cout << "Record of this voter has already been entered!\n";  }  else  {  cout << "Record of this voter does not exist!\n";  }  break;  case 4:  cout << "Going back to main menu!\n";  menu();  default:  cout << "You have entered invalid option\n";  }  }  } |

|  |
| --- |
| Candidate.h |
| #pragma once  using namespace std;  struct candidate  {  int areaNo;  string cnic;  string name;  string party;  };  void add(candidate na\_candidates[], int naEntries, int identifier);  void save(candidate na\_candidates[], int naEntries, int identifier);  void view(candidate na\_candidates[], int naEntries);  void add(candidate na\_candidates[], int naEntries, int identifier)  {  cout << "Enter Area Number:";  cin >> na\_candidates[naEntries].areaNo;  cout << "Enter Candidate's CNIC:";  cin >> na\_candidates[naEntries].cnic;  cout << "Enter Candidate's Name:";  cin >> na\_candidates[naEntries].name;  cout << "Enter Candidate's Political Party:";  cin >> na\_candidates[naEntries].party;  save(na\_candidates, naEntries, identifier);  }  void save(candidate na\_candidates[], int naEntries, int identifier)  {  ofstream fout;  if (identifier == 1)  {  fout.open("NA\_Candidates.txt");  }  else  {  fout.open("PA\_Candidates.txt");  }  for (int i = 0; i < naEntries + 1; i++)  {  fout << na\_candidates[i].areaNo << " " << na\_candidates[i].cnic << " "  << na\_candidates[i].name << " " << na\_candidates[i].party << endl;  }  fout.close();  }  void view(candidate na\_candidates[], int naEntries)  {  cout << "Area\tName\tParty\tCNIC\n";  for (int i = 0; i < naEntries; i++)  {  cout << na\_candidates[i].areaNo << "\t" << na\_candidates[i].name << "\t"  << na\_candidates[i].party << "\t" << na\_candidates[i].cnic << endl;  }  } |

|  |
| --- |
| Voter.h |
| #pragma once  using namespace std;  struct voter  {  uint64\_t cnic;  int na;  int pa;  int block;  string pollingStation;  };  void add(voter voters[], int voterEntries);  void save(voter voters[], int voterEntries);  void view(voter voters[], int voterEntries);  bool search(voter voters[], int voterEntries, uint64\_t target);  void add(voter voters[], int voterEntries)  {  cout << "Enter CNIC of Voter:";  cin >> voters[voterEntries].cnic;  cout << "Enter N.A Number of Voter:";  cin >> voters[voterEntries].na;  cout << "Enter P.A Number of Voter:";  cin >> voters[voterEntries].pa;  cout << "Enter Block Number of Voter:";  cin >> voters[voterEntries].block;  cout << "Enter Polling Station for Voter:";  cin >> voters[voterEntries].pollingStation;  save(voters, voterEntries);  }  void save(voter voters[], int voterEntries)  {  ofstream fout;  fout.open("Voters.txt");  for (int i = 0; i < voterEntries + 1; i++)  {  fout << voters[i].cnic << " " << voters[i].na << " " << voters[i].pa  << " " << voters[i].block << " " << voters[i].pollingStation << endl;  }  fout.close();  }  void view(voter voters[], int voterEntries)  {  cout << "CNIC\t\tN.A\tP.A\tBlock\tPolling\_Station\n";  for (int i = 0; i < voterEntries; i++)  {  cout << voters[i].cnic << "\t" << voters[i].na << "\t" << voters[i].pa  << "\t" << voters[i].block << "\t" << voters[i].pollingStation << endl;  }  }  bool search(voter voters[], int voterEntries, uint64\_t target)  {  for (int i = 0; i < voterEntries; i++)  {  if (voters[i].cnic == target)  {  return 1;  }  }  return 0;  } |