School of Computer Sciences

Semester 2, Academic Session 2017/2018

CPM213: Programming Language Methodology and Data Structures

Assignment 1

Name : Noor Ameera Anas Binti Renie

Matric ID : 126090

Group : G

Lecturer : Dr. Mohd Halim Bin Mohd Noor

Table of Contents

1.0 Introduction 2

2.0 Specification of Requirements 3

2.1 System Processes 3

2.2 System Constraints 3

3.0 Program Design 4

3.1 UML Diagram 4

3.2 Class Descriptions 5

4.0 Program Listing 8

Class GeneralInfor Declaration 8

Class DetailInfor Declaration 9

Class BirthDate Declaration 10

Class GeneralInfor Definition 11

Class DetailInfor Definition 13

Class BirthDate Definition 16

Function Definition 17

Main Function Definition 22

5.0 Test Data and Print Screen of Output 28

5.1 Input Data 28

5.2 Sample Output 29

Figure 1: Flowchart of the process formula for PreDAVD system 3
Figure 2: UML Class Diagram 4
Figure 3: Screenshot of Input Data Text File 28
Figure 4: Main Menu 29
Figure 5: Enter new voter interface 29
Figure 6: Enter new voter interface (1) 30
Figure 7: Search voter interface 30
Figure 8: Search voter interface (1) 31
Figure 9: Overall distribution analysis interface 31
Figure 10: Overall distribution analysis interface (1) 32
Figure 11: Gender distribution analysis interface 32
Figure 12: Gender distribution analysis interface (1) 33
Figure 13: Age distribution analysis interface 33
Figure 14: Age distribution analysis interface (1) 34
Figure 15: Display voters' information interface 34
Figure 16: Display voter's information interface (1) 35
Figure 17: End of program 35

1.0 Introduction

Social demographic factors such as gender and age group have known to be a large determinant for party popularities in Malaysian general elections. Our company, *MyFuture.com* is assigned to develop a Preliminary Data Analysis on Voting Distribution System (PreDAVD) using an object-oriented C++ programming language approach which purpose is to perform a data analysis in predicting the popularities of the current parties in Malaysia.

The PreDAVD system should be able to read and record voters general and detailed information from an input file or a manually entered information from the user of the system. General information of voters should include name, Identification Card (I.C.) Number, gender and state meanwhile detailed information should include parliament number, DUN number, contact information, date of birth and party of interest. Corresponding to the main objective of the system, the system should perform an overall distribution analysis for all parties and demographic distribution according to gender and age groups of a specific parliament. Other main functions also include searching for a voter's information from an input I.C. Number and displaying available voter's information.

2.0 Specification of Requirements

2.1 System Processes

Figure 1 below shows the flowchart which describes the processes in the PreDAVD system.

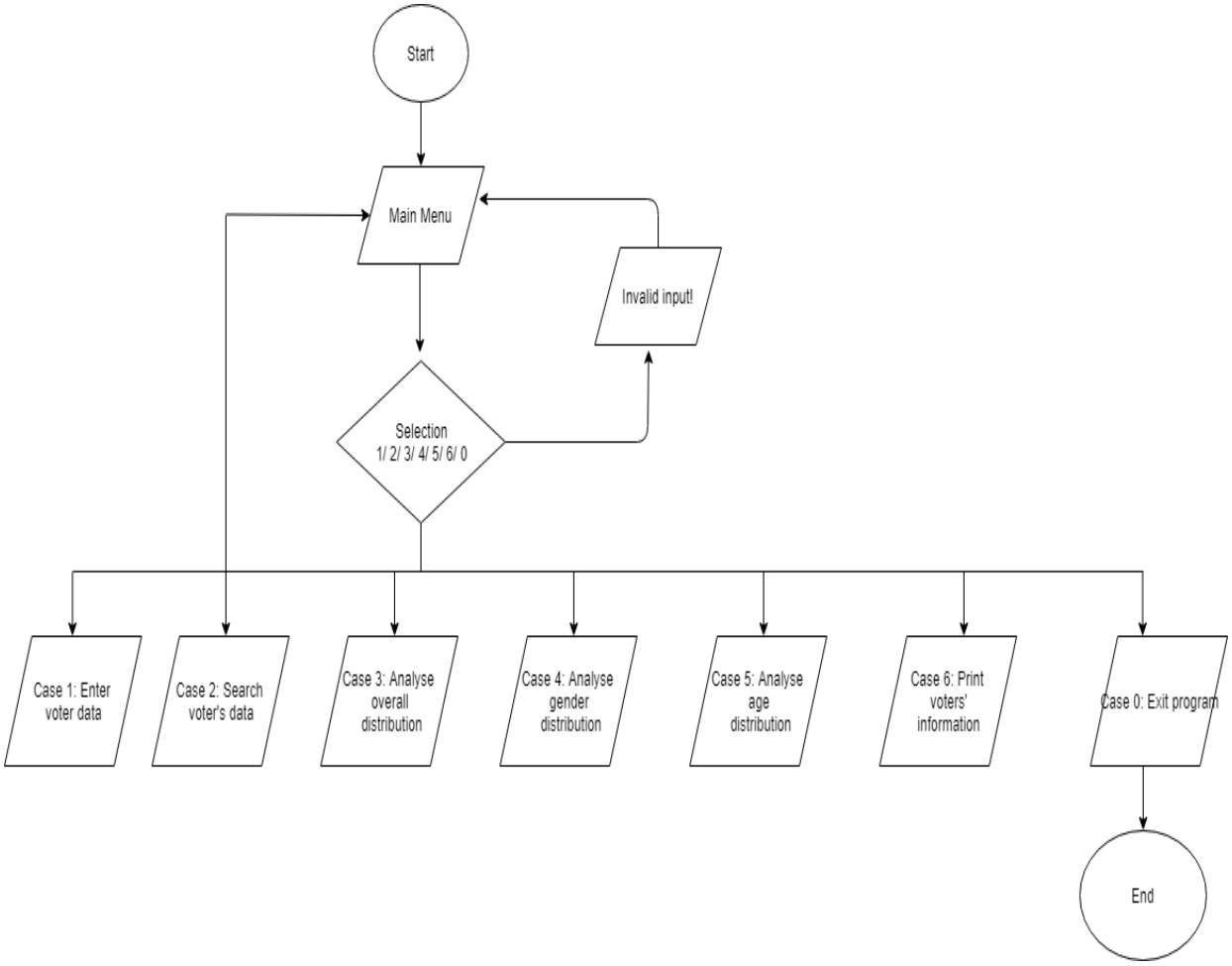

Figure 1: Flowchart of the process formula for PreDAVD system

2.2 System Constraints

Constraints of this system includes :

- Input data constraint due to *string* data type used to store general and detailed voter's information.
- Data security constraint due to data being easily accessible by anyone who uses the system.

3.0 Program Design

3.1 UML Diagram

Unified Modelling Language (UML) is a standard notation for the modelling of real-world objects as a first step in developing an object-oriented program. Figure 2 below shows the UML diagram of the class *generalInfor*, class *detailInfor* and class *birthDate*. In this program, class *generalInfor* is inherited by the class *detailInfor* and has a relationship with class *birthDate* by composition.

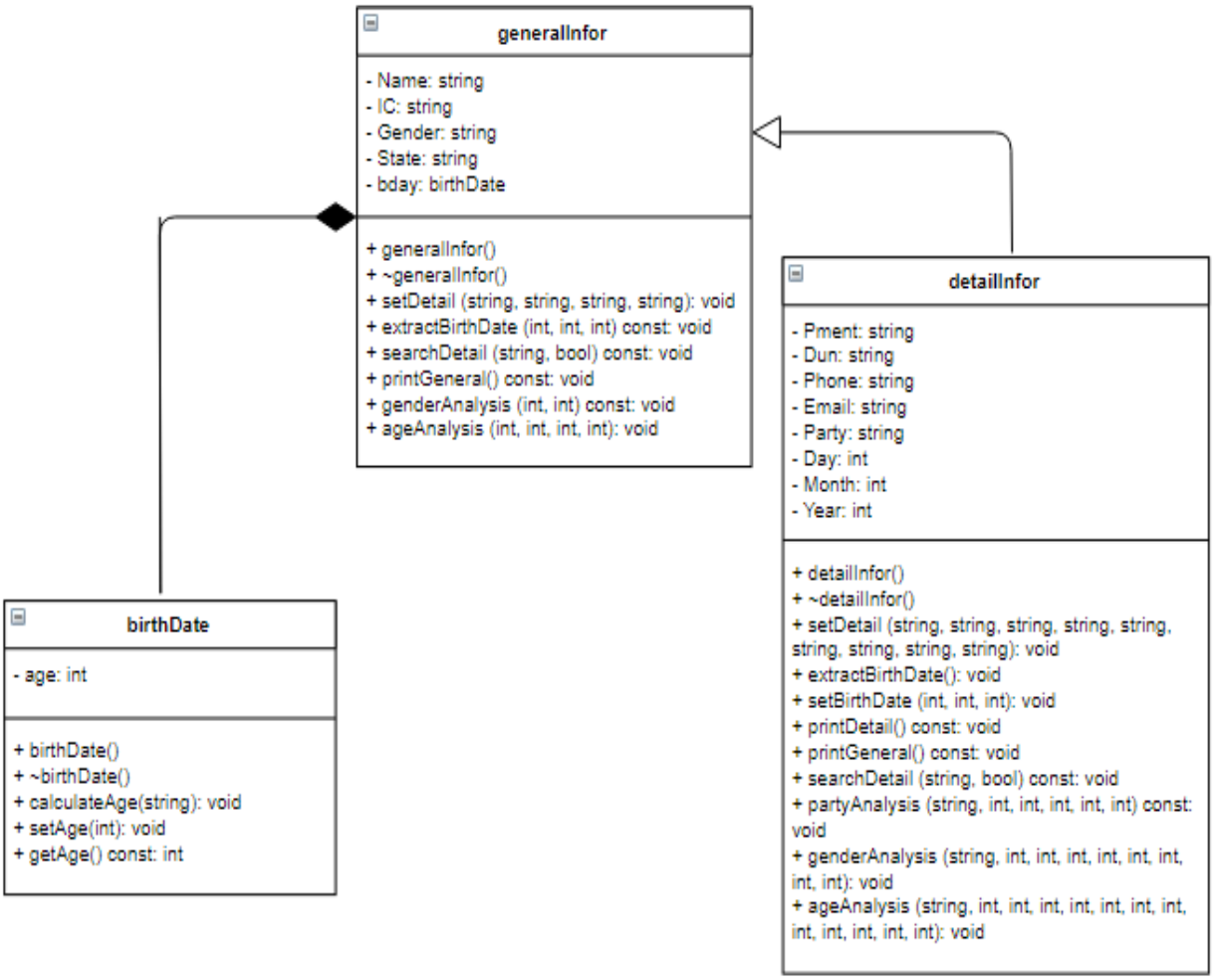

Figure 2: UML Class Diagram

3.2 Class Descriptions

<table>
<tr><td rowspan="3">Class
generalInfor</td><td>Variable Description</td></tr>
<tr><td>• Name
- Store voter’s full name in a string
• IC
- Store voter’s I.C. Number in a string
• Gender
- Store voter’s gender in a string
• State
- Store voter’s state in a string
• bday: birthDate
- object of class birthDate</td></tr>
<tr><td>Function/Method Description
• generalInfor(): void
- Default constructor
• ~generalInfor(): void
- Destructor when the lifetime of a generalInfor object ends
• setDetail(string, string, string, string): void
- Set name, I.C. Number, gender and state which are passed from function setDetail in class detailInfor
• extractBirthDate(int, int, int) const: void
- Extract day, month, and year from voter’s I.C. Number
• searchDetail(string, bool): void
- Search voter from an input I.C. number which is passed from function searchDetail in class detailInfor and return a bool data type by reference
• printGeneral const(): void
- Print name, I.C. number, gender and state
• genderAnalysis(int, int) const: void
- Count voter’s gender and return the counted values by reference
• ageAnalysis(int, int, int, int) const: void
- Count voter’s age according to age groups and return the counted values by</td></tr>
</table>

	reference
Class detailInfor	**Variable Description**
	• Pment: Store parliament number in a string • Dun: Store DUN number in a string • Phone: Store voter's phone number in a string • Email: Store voter's email address in a string • Party: Store voter's party preference in a string • Day: Store voter's date of birth in an integer data type • Month: Store voter's month of birth in an integer data type • Year: Store voter's year of birth in an integer data type
	Function/Method Description
	• detailInfor(): - Default constructor • ~detailInfor(): - Destructor when the lifetime of a detailInfor object ends • setDetail(string, string, string, string, string, string, string, string, string): - Receive voter's detailed information from main function, set voter's detailed information and call for setDetail function in parent class generalInfor • extractBirthDate(): void - Call extractBirthDate function in generalInfor class and setBirthDate function on the same class • setBirthDate(int, int, int): void - Set date, month, and year of birth • printDetail() const: void - Print voter's detailed information • printGeneral() const: void - Call for print voter's general information function in generalInfor class • searchDetail(string, bool) const: void - Call for search detail function in generalInfor class • partyAnalysis(string, int, int, int, int, int) const: void - Count voter's party preference, receive parliament and total voters parameters, and return count party parameters in main function by reference

<table>
<tr><td></td><td>

- genderAnalysis(string, int, int, int, int, int, int, int, int, int, int): void

- Call for gender analysis function in generalInfor, receive and return parliament, total voters, and gender parameters in main function by reference

- ageAnalysis(string, int, int, int, int, int, int, int, int, int, int, int, int, int): void

- Call for age analysis function in generalInforclass, receive and return parliament, total voters, and age parameters in main function by reference

</td></tr>
<tr><td rowspan="3">Class birthDate</td><td>Variable Description</td></tr>
<tr><td>

- Age: Store voter’s age

</td></tr>
<tr><td>Function/Method Description</td></tr>
<tr><td></td><td>

- birthDate():

- Default constructor

- ~birthDate():

- Destructor

- calculateAge(string): void

- Extract age from voter’s I.C. number passed from ageAnalysis function in class generalInfor

- setAge(int): void

- Set voter’s age received from calculateAge function

- getAge() const: int

- Return voter’s age when it is called

</td></tr>
</table>

4.0 Program Listing

Class GeneralInfor Declaration

```
#ifndef GENERALINFOR_H
#define GENERALINFOR_H

#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "birthDate.h"

using namespace std;

class generalInfor
{
    private:
        string Name;
        string IC;
        string Gender;
        string State;
        birthDate bday;

    public:
        generalInfor();
        ~generalInfor();
        void setDetail(string, string, string, string);
        void extractBirthDate(int&, int&, int&) const;
        void searchDetail(string, bool&) const;
        void printGeneral() const;
        void genderAnalysis(int&, int&) const;
        void ageAnalysis(int&, int&, int&, int&);
};

#endif // GENERALINFOR_H
```

Class DetailInfor Declaration

```
#ifndef DETAILINFOR_H
#define DETAILINFOR_H

#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>

using namespace std;

class detailInfor: public generalInfor
{

    private:
        string Pment;
        string Dun;
        string Phone;
        string Email;
        string Party;
        int Day;
        int Month;
        int Year;

    public:
        detailInfor();
        ~detailInfor();
        void setDetail(string, string, string, string, string, string,
                       string, string, string);
        void extractBirthDate();
        void setBirthDate(int, int, int);
        void printDetail() const;
        void printGeneral() const;
        void searchDetail(string, bool&) const;
        void partyAnalysis(string, int&, int&, int&, int&) const;
        void genderAnalysis(string, int&, int&, int&, int&, int&, int&,
                            int&) const;
        void ageAnalysis(string, int&, int&, int&, int&, int&, int&, int&,
                         int&, int&, int&, int&, int&, int&);
};

#endif // DETAILINFOR_H
```

Class BirthDate Declaration

```
#ifndef BIRTHDATE_H
#define BIRTHDATE_H

#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>

using namespace std;

class birthDate
{
    private:
        int age;

    public:
        birthDate();
        ~birthDate();
        void calculateAge(string);
        void setAge(int);
        int getAge() const;
};

#endif // BIRTHDATE_H
```

Class GeneralInfor Definition

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "generalInfor.h"
#include "detailInfor.h"
#include "birthDate.h"

using namespace std;

generalInfor::generalInfor():bday() //constructor
{
    Name    = " ";
    IC      = " ";
    Gender  = " ";
    State   = " ";
}

generalInfor::~generalInfor() //default constructor
{
}

void generalInfor::setDetail(string na, string id, string ge, string st)
{ //set voter's general information
    Name    = na;
    IC      = id;
    Gender  = ge;
    State   = st;
}

void generalInfor::printGeneral() const
{ //print voter's general information
    cout << endl << " Name              : " << Name;
    cout << endl << " I.C. Number       : " << IC;
    cout << endl << " Gender            : " << Gender;
    cout << endl << " State             : " << State;
}

void generalInfor::extractBirthDate(int& day, int& month, int& year) const
{ //extract birth date from I.C number
    day     = atoi(IC.substr(4, 2).c_str());
    month   = atoi(IC.substr(2, 2).c_str());
    year    = atoi(IC.substr(0, 2).c_str());
}

void generalInfor::searchDetail(string id, bool& found) const
{ //search detail's information based on I.C. number
    if (IC==id)
    {
        found = true;
```

```
    }
}

void generalInfor::genderAnalysis(int& female, int& male) const
{ //count voters' gender
    if (Gender == "F")
        female++;
    else if (Gender == "M")
        male++;
}

void generalInfor::ageAnalysis(int& age20, int& age30, int& age50, int&
age60)
{ //count voters' age according to age groups
    int age;

    bday.calculateAge(IC);

    age = bday.getAge();

    if (age >= 21 && age <= 30)
        age20++;
    else if (age >= 31 && age <= 49)
        age30++;
    else if (age >= 50 && age <= 60)
        age50++;
    else if (age > 60)
        age60++;
}
```

Class DetailInfor Definition

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "generalInfor.h"
#include "detailInfor.h"

using namespace std;

detailInfor::detailInfor():generalInfor() //default constructor
{
    string Pment    = " ";
    string Dun      = " ";
    string Phone    = " ";
    string Email    = " ";
    string Party    = " ";
    int Day         = 00;
    int Month       = 00;
    int Year        = 1900;
}

detailInfor::~detailInfor() //destructor
{
}

void detailInfor::setDetail(string na, string id, string ge, string st,
string pm, string du,
                            string ph, string em, string pa)
{ //set voter's general and detailed information
    generalInfor::setDetail(na, id, ge, st);
    extractBirthDate();

    Pment   = pm;
    Dun     = du;
    Phone   = ph;
    Email   = em;
    Party   = pa;
}

void detailInfor::extractBirthDate()
{ //extract birth date from voter's I.C. number
    int d,m,y;

    generalInfor::extractBirthDate(d, m, y);

    setBirthDate(d, m, y);
}

void detailInfor::setBirthDate(int d, int m, int y)
{ //set voter's birth date
    Day     = d;
    Month   = m;
```

```
    Year    = y;
}

void detailInfor::printDetail() const
{ //print voter's detailed information
    cout << endl << " Parliament Number : " << Pment;
    cout << endl << " DUN Number        : " << Dun;
    cout << endl << " Date of Birth     : " << Day << ":" << Month << ":19"
         << Year;
    cout << endl << " Phone number      : " << Phone;
    cout << endl << " Email address     : " << Email;
    cout << endl << " Party preference  : " << Party;
}

void detailInfor::printGeneral() const
{ //print voter's general information
    generalInfor::printGeneral();
}

void detailInfor::searchDetail(string id, bool& found) const
{ //call search voter's information function in base detailInfor class
    generalInfor::searchDetail(id, found);

    if  (found == true)
    {
        printDetail();
    }
}

void detailInfor::partyAnalysis(string pm, int& teratai, int& melor, int&
dahlia, int& total) const
{ //analyse overall parties distribution based on selected parliament
    if(Pment == pm)
    {
        total++;

        if (Party == "T")
            teratai++;
        else if(Party == "M")
            melor++;
        else if (Party == "D")
            dahlia++;
    }
}

void detailInfor::genderAnalysis(string pm, int& female_T, int& male_T,
                                 int& female_M, int& male_M, int& female_D,
                                 int& male_D, int& total) const
{ //analyse demographic distribution based on genders for selected
parliament
    if(Pment == pm)
    {
        total++;

        if (Party == "T")
            generalInfor::genderAnalysis(female_T, male_T);
        else if(Party == "M")
```

```
            generalInfor::genderAnalysis(female_M, male_M);
        else if (Party == "D")
            generalInfor::genderAnalysis(female_D, male_D);
    }
}


void detailInfor::ageAnalysis(string pm, int& total, int& age20s_T, int&
                              age20s_M, int& age20s_D, int& age30s_T, int&
                              age30s_M, int& age30s_D, int& age50s_T, int&
                              age50s_M, int& age50s_D, int& age60s_T, int&
                              age60s_M, int& age60s_D)
{ //analyse demographic distribution based on age groups for selected
parliament
    if(Pment == pm)
    {
        total++;

        if (Party == "T")
        {
            generalInfor::ageAnalysis(age20s_T, age30s_T, age50s_T,
                                      age60s_T);
        }
        else if (Party == "M")
        {
            generalInfor::ageAnalysis(age20s_M, age30s_M, age50s_M,
                                      age60s_M);
        }
        else if (Party == "D")
        {
            generalInfor::ageAnalysis(age20s_D, age30s_D, age50s_D,
                                      age60s_D);
        }

    }

}
```

Class BirthDate Definition

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "generalInfor.h"
#include "detailInfor.h"
#include "birthDate.h"

using namespace std;

birthDate::birthDate() //default constructor
{
    age = 0;
}

birthDate::~birthDate() //destructor
{
}

void birthDate::calculateAge(string ID) //calculate voter's age on year
2018
{
    int Age;

    Age = 118 - atoi(ID.substr(0, 2).c_str());

    setAge(Age);
}

void birthDate::setAge(int Age) //set voter's age
{
    age = Age;
}

int birthDate::getAge() const //return voter's age
{
    return age;
}
```

Function Definition

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "generalInfor.h"
#include "detailInfor.h"
#include "birthDate.h"

using namespace std;

void Welcome()
{ //Display Main Header
    cout << endl << setw(90) << "PreDAVD - Preliminary Data Analysis on
    Voting Distribution System";
    cout << endl << setw(120) << setfill('*') << " "  << setfill(' ');
}

void DisplayMain()
{ //Display Main Menu
    cout << endl << " Main Menu:" << endl;
    cout << endl << " (1) Enter new voter's information.";
    cout << endl << " (2) Search voter's information based on I.C.
    number.";
    cout << endl << " (3) View analysis on overall distribution based on
    party preferences.";
    cout << endl << " (4) View analysis on demographic distribution based
    on genders.";
    cout << endl << " (5) View analysis on demographic distribution based
    on age groups.";
    cout << endl << " (6) View all voters\' information.";
    cout << endl << " (0) Exit the program.";
    cout << endl << endl << "     >> Your selection (1/2/3/4/5/6/0) : ";
}

void AddVoter(string& name, string& ID, string& gender, string& state,
              string& pment, string& dun, string& phone, string& email,
              string& party)
{ //Request and read new voter's information
    bool check;

    cout << endl << endl << setw(75) << "Entering new voter's
    information..." << endl << endl;

    cout << " 1. Full Name\t\t\t\t\t\t: ";
    getline(cin,name,'\n');

    cout << " 2. I.C. Number (Enter without space i.e. 701017101234) : ";
    getline(cin,ID,'\n');

        check = false;
        while(check == false) //Input validation for I.C. Number
        {
```

```
            if (ID.length()==12)
            {
                check = true;
                break;
            }
            cout << endl << "     >> Please re-enter the voter's I.C number
            (12 digits).";
            cout << endl << "     >> I.C. Number: ";
            getline(cin,ID,'\n');
        }

    cout << " 3. Gender (M / F)\t\t\t\t\t: ";
    getline(cin,gender,'\n');

        check = false;
        while(check == false)//Input validation for gender
        {
            if (gender.length()==1)
            {
                gender = toupper(gender[0]);

                if (gender == "M" || gender== "F")
                {
                    check = true;
                    break;
                }
            }
            cout << endl << "     >> Please re-enter the voter's gender (M
            / F).";
            cout << endl << "     >> Gender: ";
            getline(cin,gender,'\n');

        }

    cout << " 4. State\t\t\t\t\t\t: ";
    getline(cin,state,'\n');

    cout << " 5. Parliament number (i.e. 101)\t\t\t: ";
    getline(cin,pment,'\n');

        check = false;
        while(check == false)//Input validation for Parliament Number
        {
            if (pment.length()==3)
            {
                check = true;
                break;
            }
            cout << endl << "     >> Please re-enter the voter's parliament
            number (3 digits).";
            cout << endl << "     >> Parliament Number: ";
            getline(cin,pment,'\n');
        }

    cout <<  " 6. DUN number (i.e. 101-101)\t\t\t\t: ";
    getline(cin,dun,'\n');
```

```
        check = false;
        while(check == false) //Input validation for DUN Number
        {
            if (dun.length()==7)
            {
                //
                check = true;
                break;
            }
            cout << endl << "     >> Please re-enter the voter's DUN number
            (Parliament number followed by " << "DUN number, separated with
            a \'-\')."; 
            cout << endl << "     >> DUN Number: ";
            getline(cin,dun,'\n');
        }

    cout << " 7. Phone number (Enter without space i.e. 0123456789)  : ";
    getline(cin,phone,'\n');

    cout << " 8. Email address (i.e abc@gmail.com)\t\t\t: ";
    getline(cin,email,'\n');

    cout << " 9. Party preference (T/ M/ D)\t\t\t\t: ";
    getline(cin,party,'\n');

        check = false;
        while(check == false)//Input validation for party preference
        {
            if (party.length()==1)
            {
                party = toupper(party[0]);

                if (party == "T" || party == "M" || party == "D")
                {
                    check = true;
                    break;
                }
            }
            cout << endl << "     >> Please re-enter the voter's party
            preference (T/ M/ D).";
            cout << endl << "     >> Parliament Number: ";
            getline(cin,party,'\n');
        }


    cout << endl << endl << "     >> Would you like to add a new voter's
    information? (Y/N) : ";
}

void VoterSearch(string& ID)
{ //Request and read voter's I.C. Number for search purpose
    cout << endl << endl << setw(85) << "Searching a voter's information
    based on I.C number...";
    cout << endl << endl << " Voter's I.C. number (Enter without space i.e.
    701017101234): ";
    getline(cin, ID,'\n');
}
```

```
void HeaderAnalysis()
{ //Print analysis' header
    cout << endl << endl << setw(75) << "Political Parties Distribution
    Analysis";
    cout << endl << setw(120) << setfill('*') << " "  << setfill(' ');
}

void ParliamentSearch(string& pment)
{ //Request and read voter's chosen parliament for analysis purpose
    cout << endl << endl << " Enter parliament code (i.e. 101): ";
    getline(cin, pment, '\n');

}

void PrintPartiesAnalysis(string pment, int t, int m, int d, int total)
{ //Display the overall distribution analysis of all parties for selected
parliament
    cout << endl << setw(75) << "# Overall Distribution For All Parties #";
    cout << endl << endl << " Parliament Code: " << pment;
    cout << endl << " Total number of voters: " << total;
    cout << endl << setw(70) << "Teratai \t Melor \t Dahlia";

    cout << endl << " No. of voters" << setw(37) << t << "\t   " << m <<
         "\t   " << d;
    cout << setprecision(2);
    cout << endl << " % of voters" << setw(40) <<
         (double)t/double(total)*100 << "\t   " <<
         (double)m/double(total)*100 << "\t   " <<
         (double)d/double(total)*100;
}

void PrintGenderAnalysis(string pment, int female_T, int male_T, int
female_M, int male_M, int female_D,
                         int male_D, int total)
{ //Display the demographic distribution analysis based on genders for
selected parliament
    cout << endl << setw(75) << "# Demographic Distribution Based On
    Genders #";
    cout << endl << endl << " Parliament Code: " << pment;
    cout << endl << " Total number of voters: " << total;
    cout << endl << setw(70) << "Teratai \t Melor \t Dahlia";

    cout << setprecision(2);
    cout << endl << " % of male" << setw(40)
    << (double)male_T/double(total)*100 << "\t   " <<
    (double)male_M/double(total)*100 << "\t   " <<
    (double)male_D/double(total)*100;
    cout << endl << " % of female" << setw(38) <<
    (double)female_T/double(total)*100 << "\t   "
    << (double)female_M/double(total)*100 << "\t   " <<
    (double)female_D/double(total)*100;
}

void PrintAgeAnalysis(string pment, int total, int age20s_T, int age20s_M,
                      int age20s_D, int age30s_T, int age30s_M, int
                      age30s_D, int age50s_T, int age50s_M, int age50s_D,
```

```
                        int age60s_T, int age60s_M, int age60s_D)
{ //Display the demographic distribution analysis based on age groups for
selected parliament
    cout << endl << setw(80) << "# Demographic Distribution Based On Age
    Groups #";
    cout << endl << endl << " Parliament Code: " << pment;
    cout << endl << " Total number of voters: " << total;
    cout << endl << setw(65) << "% \t\t Teratai   Melor   Dahlia";

    cout << setprecision(2);
    cout << endl << setw(40) << "21-30" << setw(13) <<
         (double)age20s_T/double(total)*100 << setw(9)
         << (double)age20s_M/double(total)*100 << setw(9) <<
         (double)age20s_D/double(total)*100;
    cout << endl << setw(40) << "31-49" << setw(13) <<
         (double)age30s_T/double(total)*100 << setw(9)
         << (double)age30s_M/double(total)*100 << setw(9) <<
         (double)age30s_D/double(total)*100;
    cout << endl << setw(40) << "50-60" << setw(13) <<
         (double)age50s_T/double(total)*100 << setw(9)
         << (double)age50s_M/double(total)*100 << setw(9) <<
         (double)age50s_D/double(total)*100;
    cout << endl << setw(40) << ">60" << setw(13) <<
         (double)age60s_T/double(total)*100 << setw(9)
         << (double)age60s_M/double(total)*100 << setw(9) <<
         (double)age60s_D/double(total)*100;

}

void DisplayInformation()
{ //Display header for voter's information
    cout << endl << endl << setw(80) << "Voters' General and Detailed
    Information";
    cout << endl << setw(120) << setfill('*') << " "  << setfill(' ');
}
```

Main Function Definition

```
#include <iostream>
#include <fstream>
#include <string>
#include <cstdlib>
#include <iomanip>
#include "generalInfor.h"
#include "detailInfor.h"
#include "birthDate.h"

using namespace std;

void Welcome();
void DisplayMain();
void AddVoter(string&, string&, string&, string&, string&, string&,
              string&, string&, string&);
void VoterSearch(string&);
void ParliamentSearch(string&);
void PrintPartiesAnalysis(string, int, int, int, int);
void PrintGenderAnalysis(string, int, int, int, int, int, int, int);
void PrintAgeAnalysis(string, int, int, int, int, int, int, int, int, int,
                      int, int, int, int);
void HeaderAnalysis();
void DisplayInformation();

int main()
{
    ifstream in;
    string name, ID, gender, state, pment, dun, phone, email, party;
    int num = 0, choice, Total;
    char choice1;
    const int SIZE = 500;

    detailInfor info[SIZE]; //declare array of info objects from class
detailInfor with size 1000

    in.open("Data.txt");

    if(in.fail())
    {
        cerr << "Input file could not be opened!" << endl;
        exit(-1);
    }

    while(!in.eof()) //read data from an input file
    {
```

```
        getline(in,name,',');
        getline(in,ID,',');
        getline(in,gender,',');
        getline(in,state,',');
        getline(in,pment,',');
        getline(in,dun,',');
        getline(in,phone,',');
        getline(in,email,',');
        getline(in,party);
        in.ignore(10,'\n');

        info[num].setDetail(name, ID, gender, state, pment, dun, phone,
                            email, party);

        num++;
    }
    in.close(); //close file

    do
    {
        Welcome(); // Display header
        DisplayMain(); // Display main menu
        cin >> choice;
        cin.ignore();

        switch(choice)
        {
            case 0: //User exits the program
                {
                    choice1 = 'N';

                    cout << endl << endl << endl << endl << endl << endl <<
                    endl << endl << endl << endl << endl;

                    break;
                }
            case 1: //User adds new voter's information
                {
                    system("cls");

                    choice1 = 'Y';

                    while(choice1 == 'Y' || choice1 == 'y')
                    {
                        AddVoter(name, ID, gender, state, pment, dun,
                                 phone, email, party);
                        cin >> choice1;
                        cin.ignore();
                        info[num].setDetail(name, ID, gender, state, pment,
                                            dun, phone, email, party);
                        num++;
                    }

                    choice1 = 'Y';
                    break;
                }
            case 2: //User searches for a voter's information based I.C.
```

```
Number
                {
                    system("cls");

                    choice1 = 'Y';

                    while(choice1 == 'Y' || choice1 == 'y')
                    {
                        int j = 0; bool found = false;

                        VoterSearch(ID);

                        while(j < num)
                        {
                            info[j].searchDetail(ID, found);
                            j++;

                            if(found == true)
                                break;
                        }

                        if(found == false)
                            cout << endl << " The voter's information does
                            not exist!";

                        cout << endl << endl << endl << "     >> Would you
                        like to search for another voter's" << "
                        information? (Y/N) : ";
                        cin >> choice1;
                        cin.ignore();
                    }

                    choice1 = 'Y';
                    break;
                }
            case 3: //User views parties overall distribution for selected
parliament
                {
                    system("cls");

                    HeaderAnalysis();

                    choice1 = 'Y';

                    while(choice1 == 'Y' || choice1 == 'y')
                    {
                        int teratai=0, melur=0, dahlia=0;
                        Total=0;

                        ParliamentSearch(pment);

                        for(int j=0; j<num; j++)
                        {
                            info[j].partyAnalysis(pment, teratai, melur,
                                                  dahlia, Total);
                        }
```

```
                        PrintPartiesAnalysis(pment, teratai, melur, dahlia,
                                             Total);

                        cout << endl << endl << endl << "     >> Would you
                        like to view the overall"
                        << " distribution for different parliament? (Y/N):
                        ";
                        cin >> choice1;
                        cin.ignore();
                    }

                    choice1 = 'Y';
                    break;
                }
            case 4: //User views demographic distribution based on genders
for selected parliament
                {
                    system("cls");

                    HeaderAnalysis();

                    choice1 = 'Y';

                    while(choice1 == 'Y' || choice1 == 'y')
                    {
                        int female_T=0, male_T=0, female_M=0, male_M=0,
                        female_D=0, male_D=0;

                        Total=0;

                        ParliamentSearch(pment);

                        for(int j=0;j<num;j++)
                        {
                            info[j].genderAnalysis(pment, female_T, male_T,
                                                   female_M, male_M,
                                                   female_D, male_D,
                                                   Total);
                        }

                        PrintGenderAnalysis(pment, female_T, male_T,
                                            female_M, male_M, female_D,
                                            male_D, Total);

                        cout << endl << endl << endl << "     >> Would you
                        like to view the distribution for"
                        << " different parliament? (Y/N): ";
                        cin >> choice1;
                        cin.ignore();
                    }
                    choice1 = 'Y';
                    break;
                }
            case 5: //User views demographic distribution based on age
groups for selected parliament
                {
                    system("cls");
```

```
                HeaderAnalysis();

                choice1 = 'Y';

                while(choice1 == 'Y' || choice1 == 'y')
                {
                    int age20s_T=0, age20s_M=0, age20s_D=0, age30s_T=0,
                    age30s_M=0, age30s_D=0, age50s_T=0, age50s_M=0,
                    age50s_D=0, age60s_T=0, age60s_M=0, age60s_D=0;

                    ParliamentSearch(pment);

                    for(int j=0;j<num;j++)
                    {
                        info[j].ageAnalysis(pment, Total, age20s_T,
                                            age20s_M, age20s_D,
                                            age30s_T, age30s_M,
                                            age30s_D, age50s_T,
                                            age50s_M, age50s_D,
                                            age60s_T, age60s_M,
                                            age60s_D);

                    }

                    PrintAgeAnalysis(pment, Total, age20s_T, age20s_M,
                                     age20s_D, age30s_T, age30s_M,
                                     age30s_D, age50s_T, age50s_M,
                                     age50s_D, age60s_T, age60s_M,
                                     age60s_D);

                    cout << endl << endl << endl << "     >> Would you
                    like to view the distribution for"
                    << " different parliament? (Y/N): ";
                    cin >> choice1;
                    cin.ignore();
                }
                choice1 = 'Y';
                break;
            }
        case 6: //User views voters' information
            {
                system("cls");

                DisplayInformation();

                for(int j=0; j<num ; j++)
                {
                    info[j].printGeneral();
                    info[j].printDetail();
                    cout << endl << endl;
                }
                cout << " Total number of voters in the Pre-DAVD
                system: " << num << endl << endl;

                choice1 = 'Y';
                break;
```

```
                }
            default: //User enters none of the available option
                {
                    cout << endl << "     >> You've entered the wrong
input! Press any key to continue. ";
                    cin.get();

                    system("cls");

                    choice1 = 'Y';
                }
        }

        if (choice!=0) //Display return-to-main-menu-message
        {
            cout << endl << "     >> Returning to main menu. Press any key
            to continue.";
            cin.get();
            system("cls");
        }

    } while(choice1 !='N' && choice1 !='n' );

} //end of main function
```

5.0 Test Data and Print Screen of Output

5.1 Input Data

In this program, voter's information is read from an input text file which is shown in figure below.

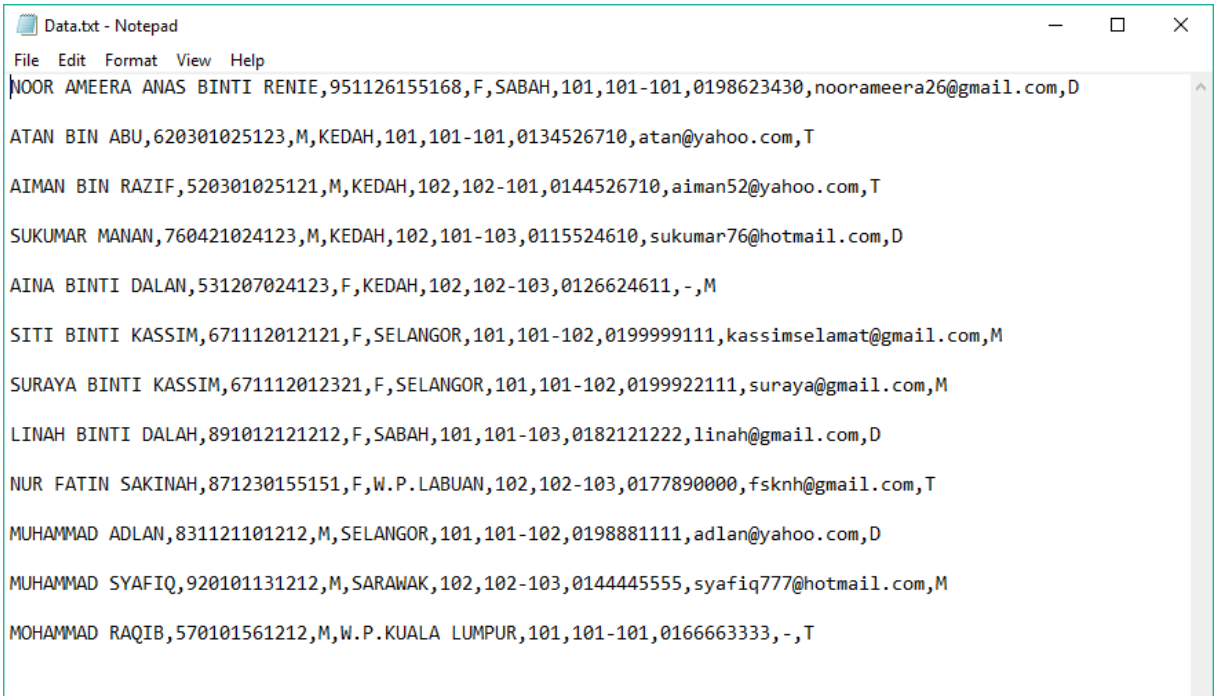

Figure 3: Screenshot of Input Data Text File

5.2 Sample Output

The screenshots of sample outputs are shown in below figures.

Scenario 1: Main Menu Selection

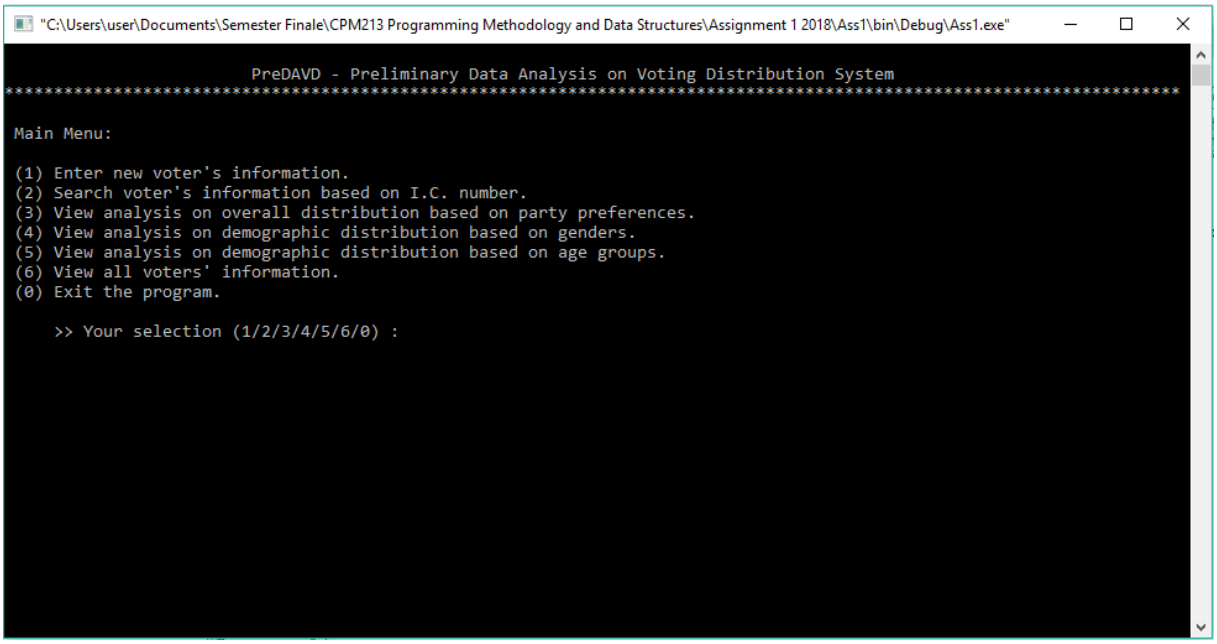

Figure 4: Main Menu

Scenario 2: User chooses to enter new voter's information (main menu selection: 1)

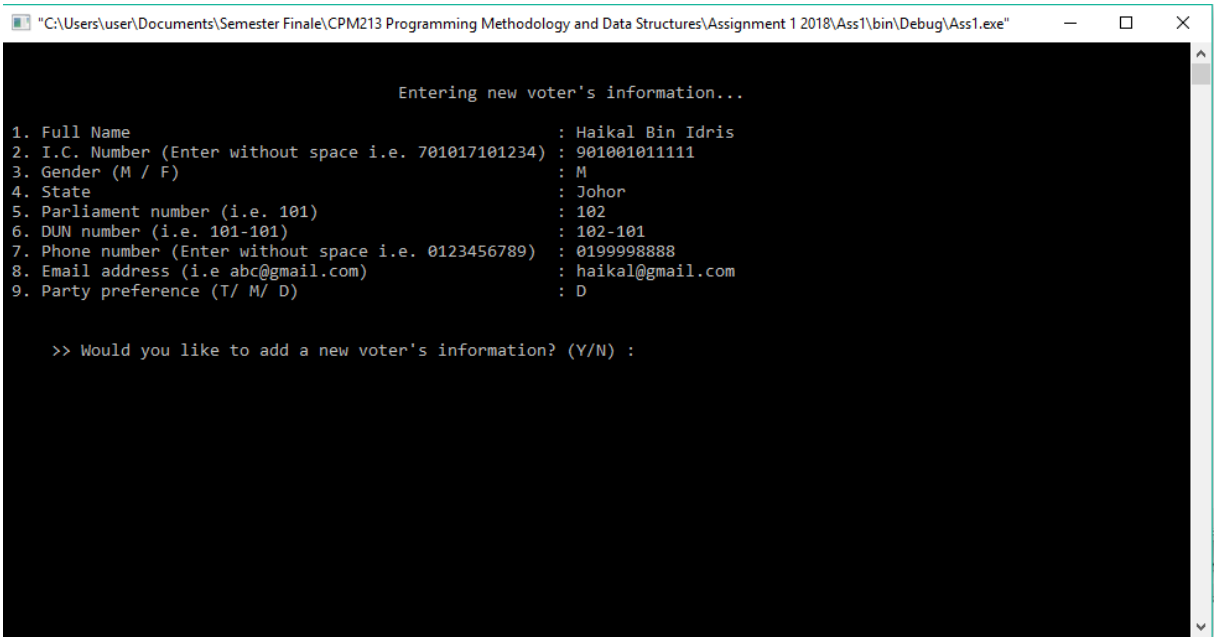

Figure 5: Enter new voter interface

Scenario 2.1: User chooses to add another voter's information (with input validation)

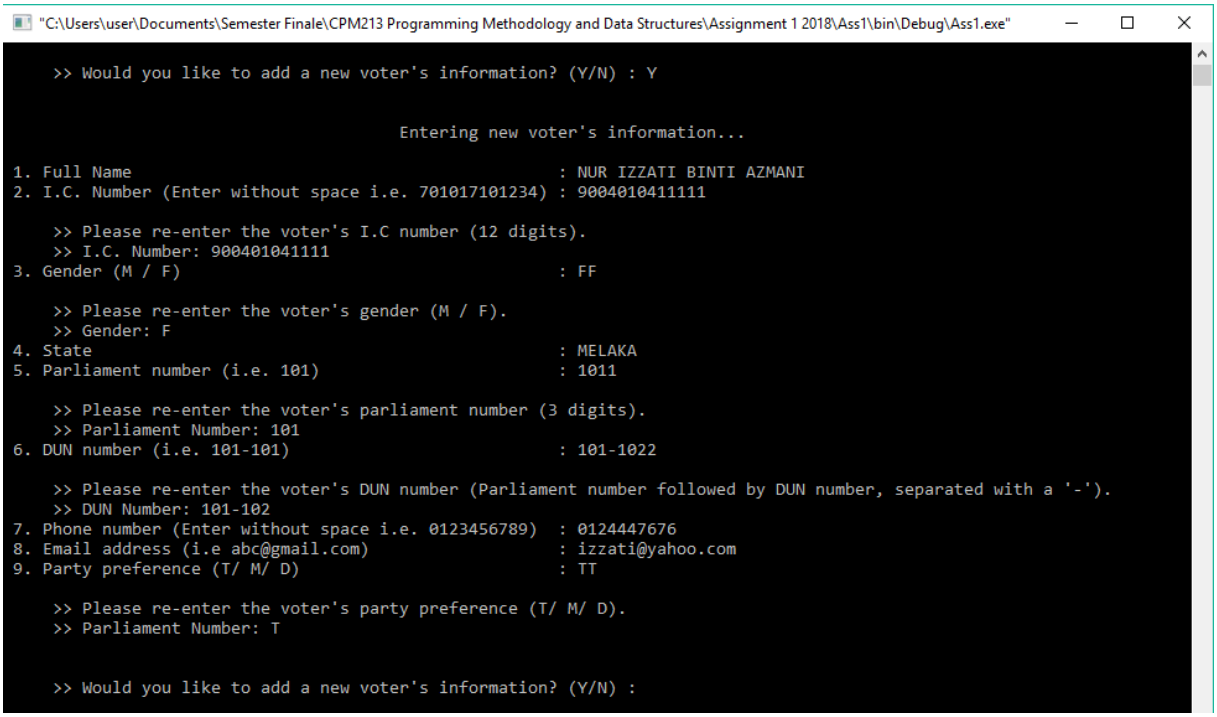

Figure 6: Enter new voter interface (1)

Scenario 3: User chooses to search for a voter's information (main menu selection: 2)

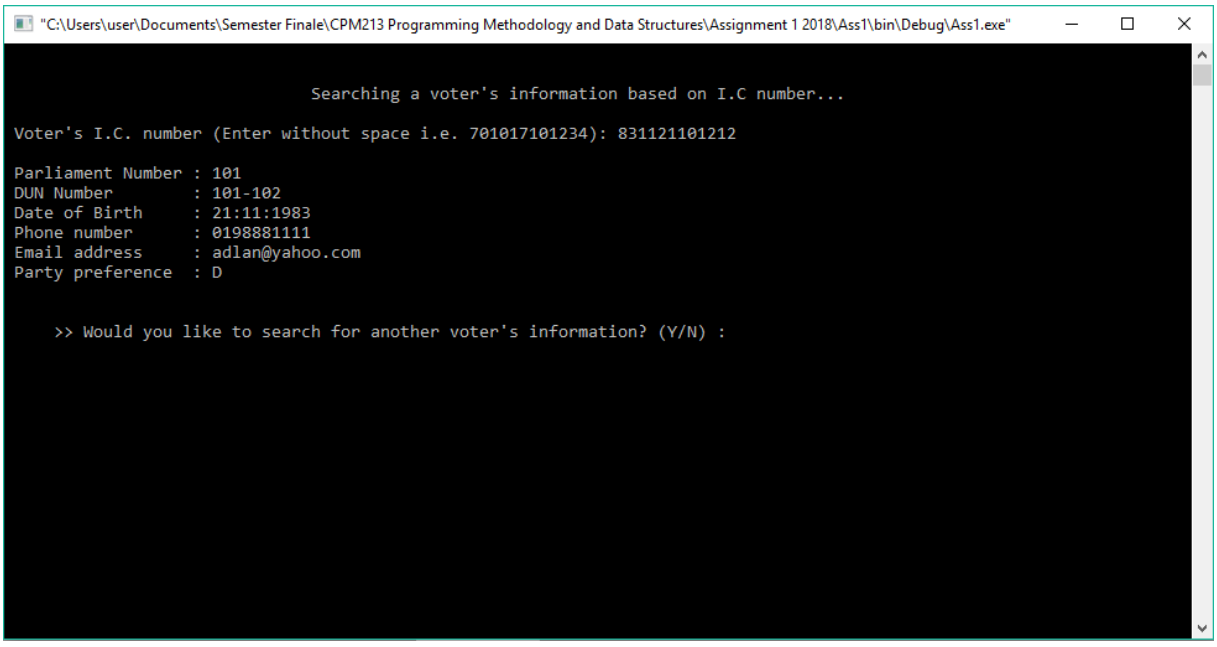

Figure 7: Search voter interface

Scenario 3.1: User chooses to search for another voter's information

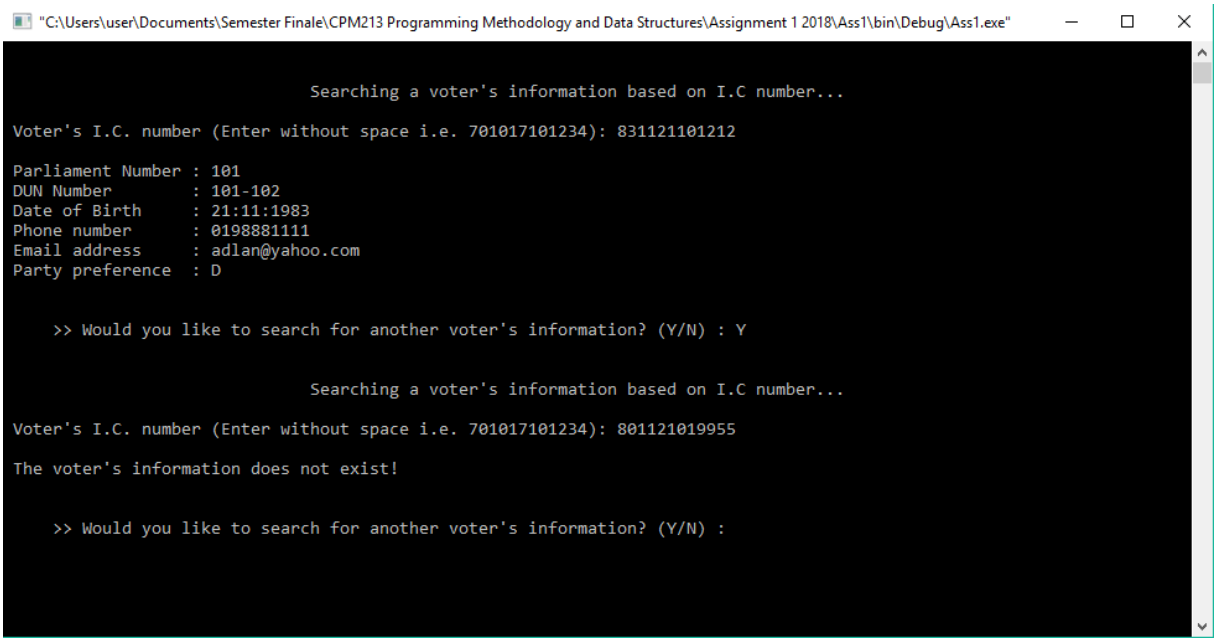

Figure 8: Search voter interface (1)

Scenario 4: User chooses to view the overall distribution for a parliament (main menu selection: 3)

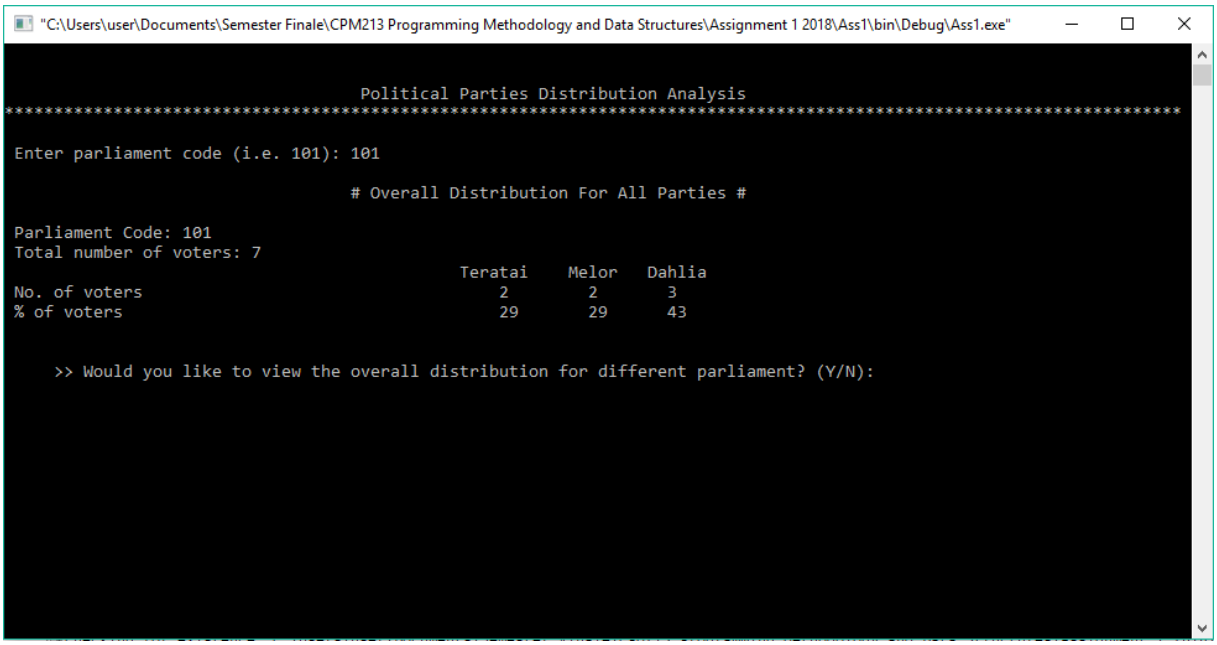

Figure 9: Overall distribution analysis interface

Scenario 4: User chooses to view the overall distribution for another parliament

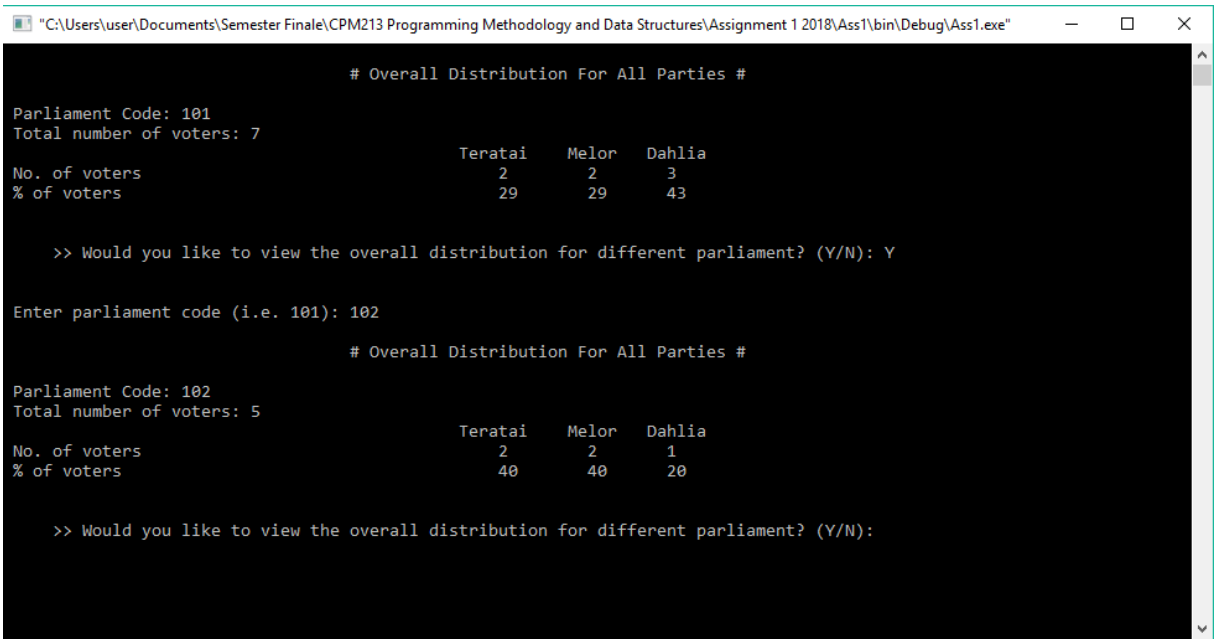

Figure 10: Overall distribution analysis interface (1)

Scenario 5: User chooses to view the gender distribution for a parliament (main menu selection: 4)

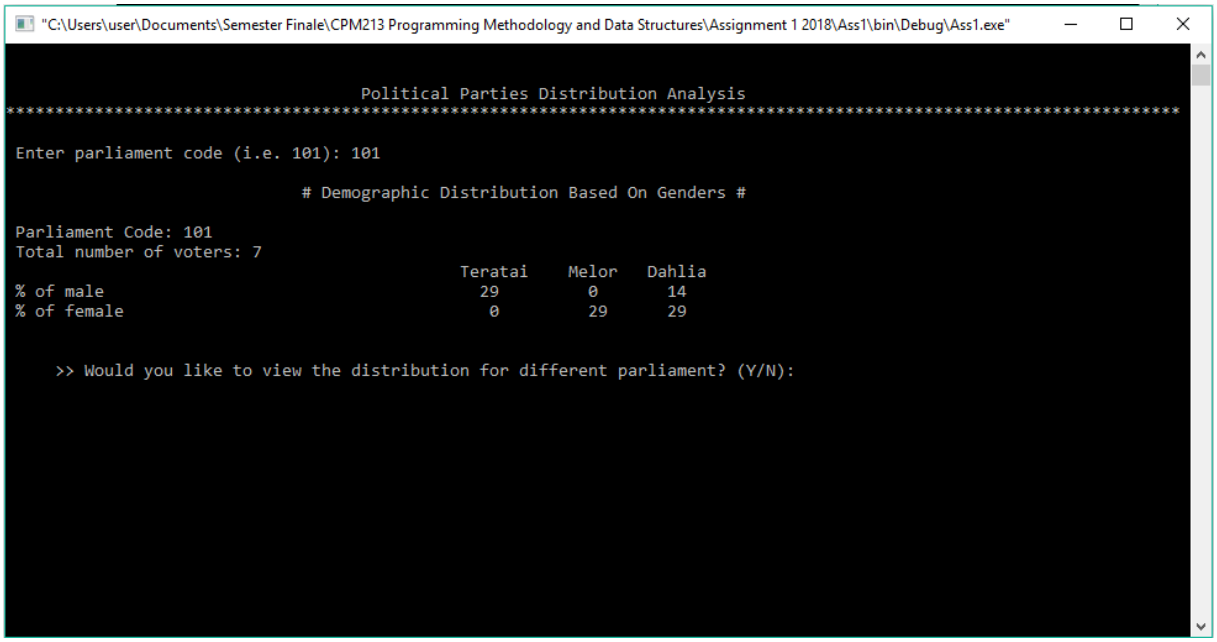

Figure 11: Gender distribution analysis interface

Scenario 5.1: User chooses to view the gender distribution for another parliament

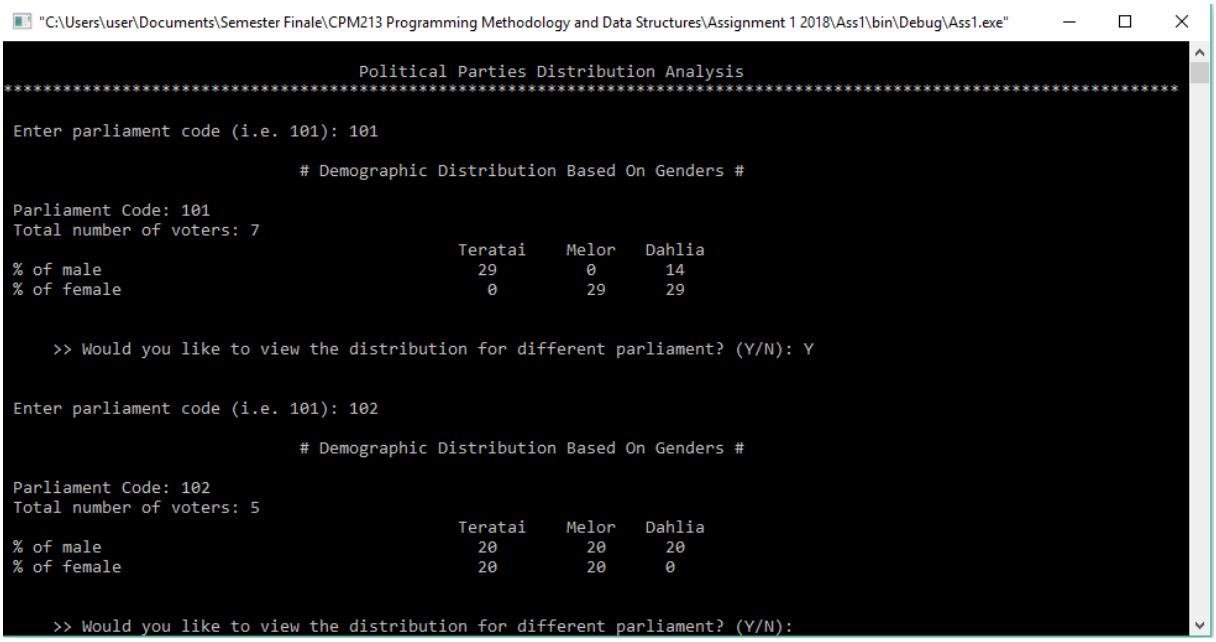

Figure 12: Gender distribution analysis interface (1)

Scenario 6: User chooses to view the age distribution for a parliament (main menu selection: 5)

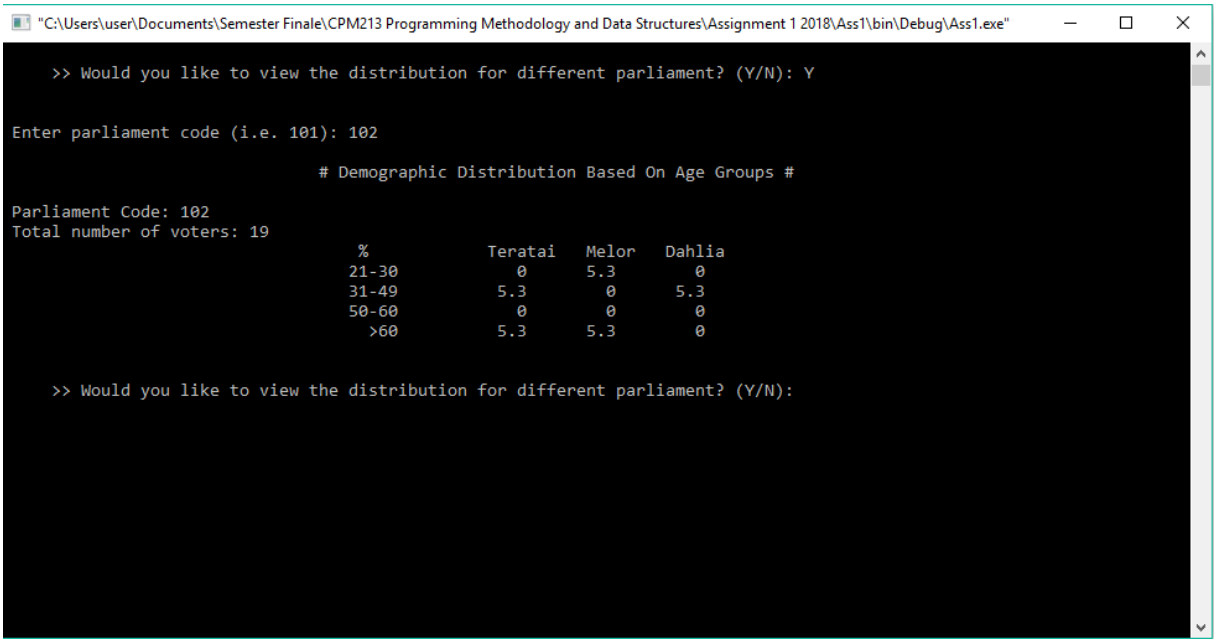

Figure 13: Age distribution analysis interface

Scenario 6.1: User chooses to view the age distribution for another parliament

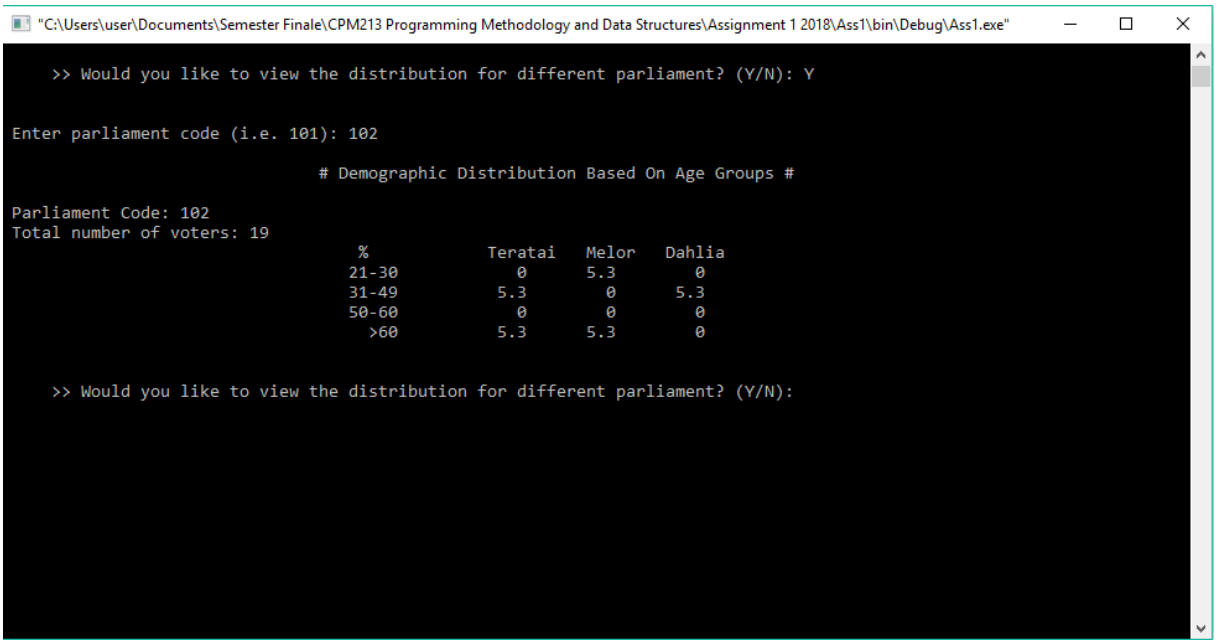

Figure 14: Age distribution analysis interface (1)

Scenario 7: User chooses to view currently stored voters' information (main menu selection: 6)

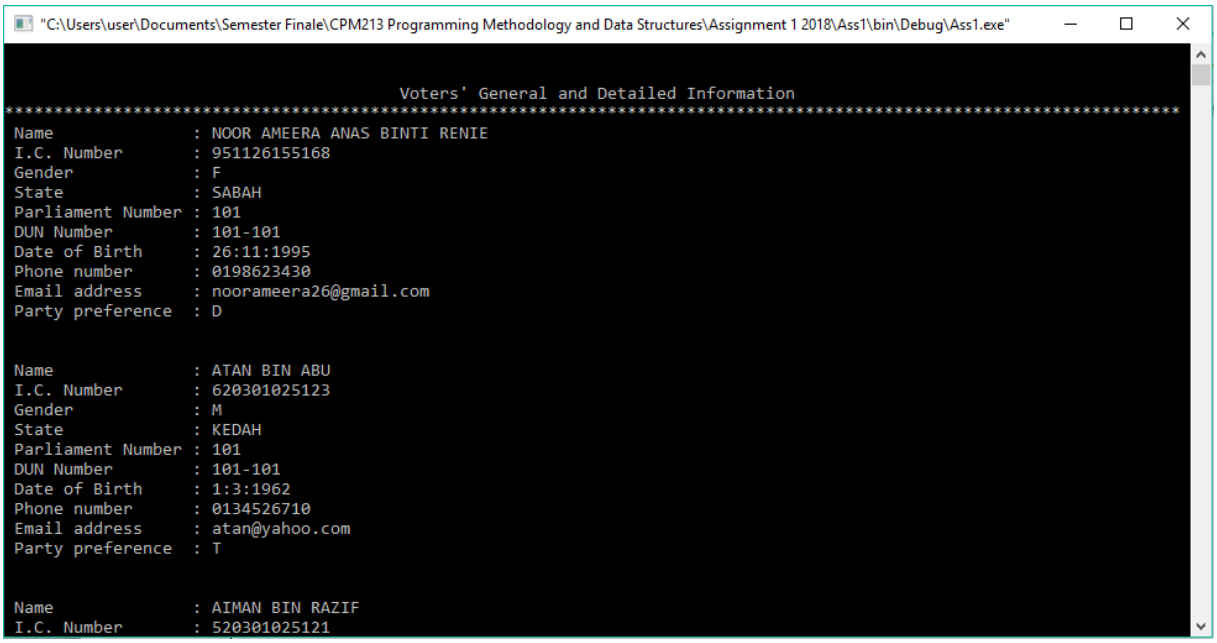

Figure 15: Display voters' information interface

Scenario 7.1: User chooses to view currently stored voters' information (continued)

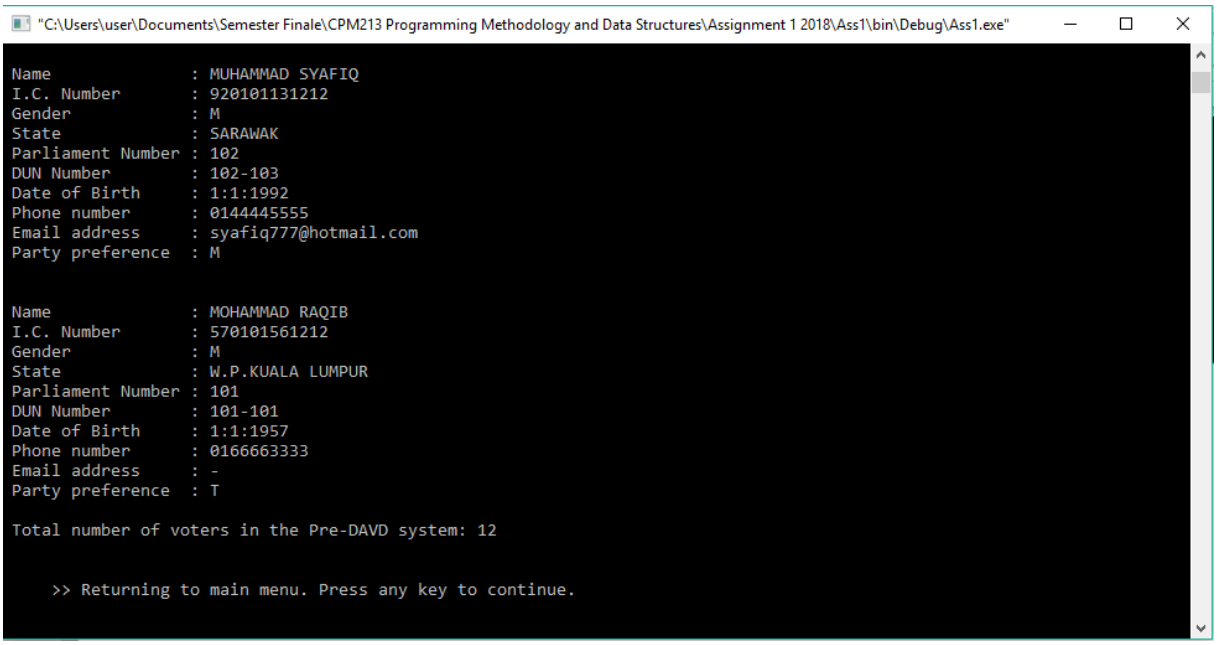

Figure 16: Display voter's information interface (1)

Scenario 8: User chooses to exit the program

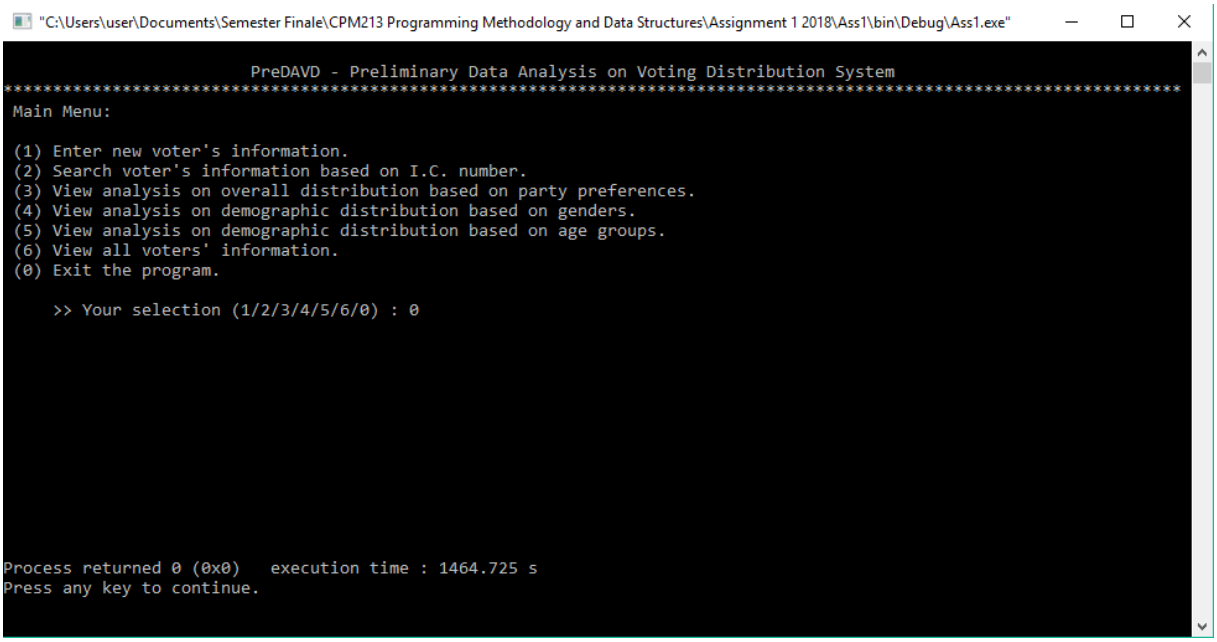

Figure 17: End of program