PROJECT WORK

ON

JAVA Program for maintaining a Phonebook

BY

IKEDINOBI ONYEKA

1300512

ISAAC NKWAZEMA

1304793

February 2022

JAVA Program for maintaining a Phonebook

This is to clarify that:

Mr. Onyeka Ikedinobi

Mr. Isaac Nkwazema

Has Successfully Developed:

A Java program for maintaining a phone book

Submitted By:

Mr. Onyeka Ikedinobi

Mr. Isaac Nkwazema

Date Of Issue:

January 23rd 2022

DECLARATION

This is to certify that this project work was done by **Ikedinobi Onyeka** and **Isaac Nkwazema** and to declare that this project represents our original work which has not been previously submitted by anyone.

ACKNOWLEDGEMENTS

I will like to acknowledge all those who have given support and helped make this project a success I also wish to express gratitude to my faculty for his valuable guidance and support for the completion of this project I finally would like to thank my colleague for the valuable roles they played to make the completion of this project a success.

TABLE OF CONTENT

CHAPTER ONE

INTRODUCTION

1.1) Background to study

A database is generally used for storing related, structured data, with well defined data formats, in an efficient manner for insert, update and/or retrieval (depending on application).

On the other hand, a file system is a more unstructured data store for storing arbitrary, probably unrelated data. The file system is more general, and databases are built on top of the general data storage services provided by file systems.

The aim of the project is to develop an application to take in contacts details and store them in a text file.

1.2) Problem of definition

Your Program should allow user to "ADD", "DELETE", "MODIFY", "SEARCH", "Display All Records" and "COUNT"

records.

All data should be stored in text file and should be available whenever the program is executed.

1.3) Aim & Objectivies

We will be using the File system since it's more efficient than database. For simple operations, read, write and delete file operations are faster and simple with the file system

Objectives

- To add contact details
- To delete contact details
- To modify contact details
- To search for contact details using an inputted term
- To display all contact details
- To count number of contacts in the text file

1.4) Scope of study

Using a text file with the Java programming language

CHAPTER TWO

ANALYSIS AND DESIGN

Analysis emphasizes an investigation of the problem and requirements, rather than a solution. For example, if a new computerized library information system is desired, how will it be used?

"Analysis" is a broad term, best qualified, as in requirements analysis (an investigation of the requirements) or object analysis (an investigation of the domain objects).

Design emphasizes a conceptual solution that fulfills the requirements, rather than its implementation. For example, a description of a database schema and software objects. Ultimately, designs can be implemented.

As with analysis, the term is best qualified, as in object design or database design.

2.1) Analysis

File should have following data:

First Name

Last Name

Address

City

Phone Number

Search should be available on Last Name, City and Phone Number or combination of either two. Search should

display complete record.

Records should be properly displayed either GRID WISE [Page Scroll should be maintained] or One Record at a

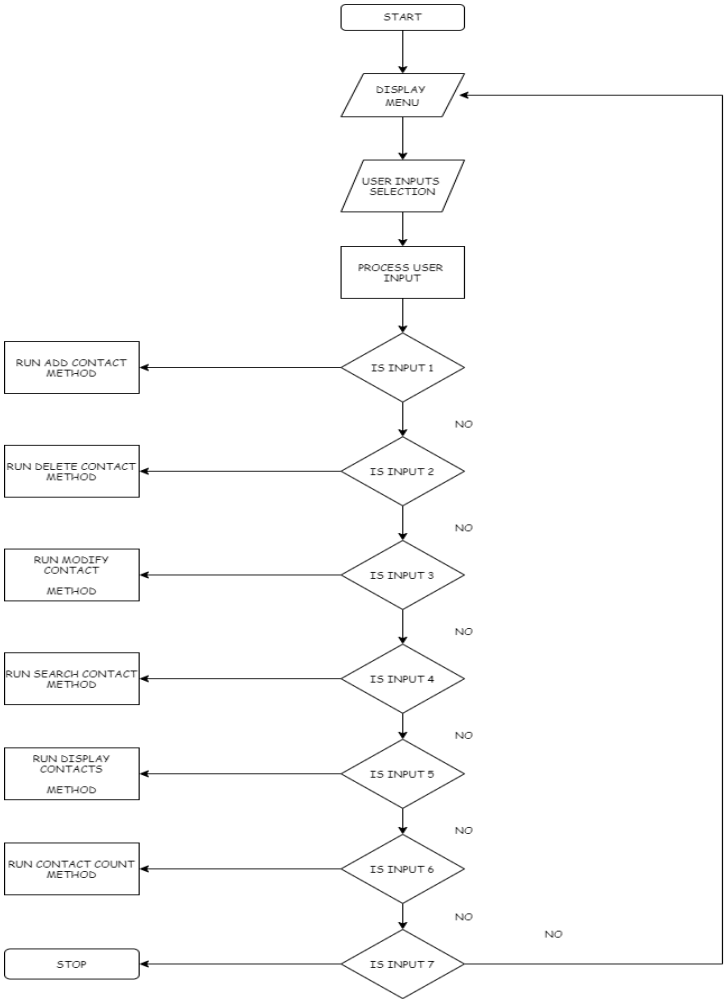
time with option to move to next record or quit.

Proper validations should be maintained on all data fields.

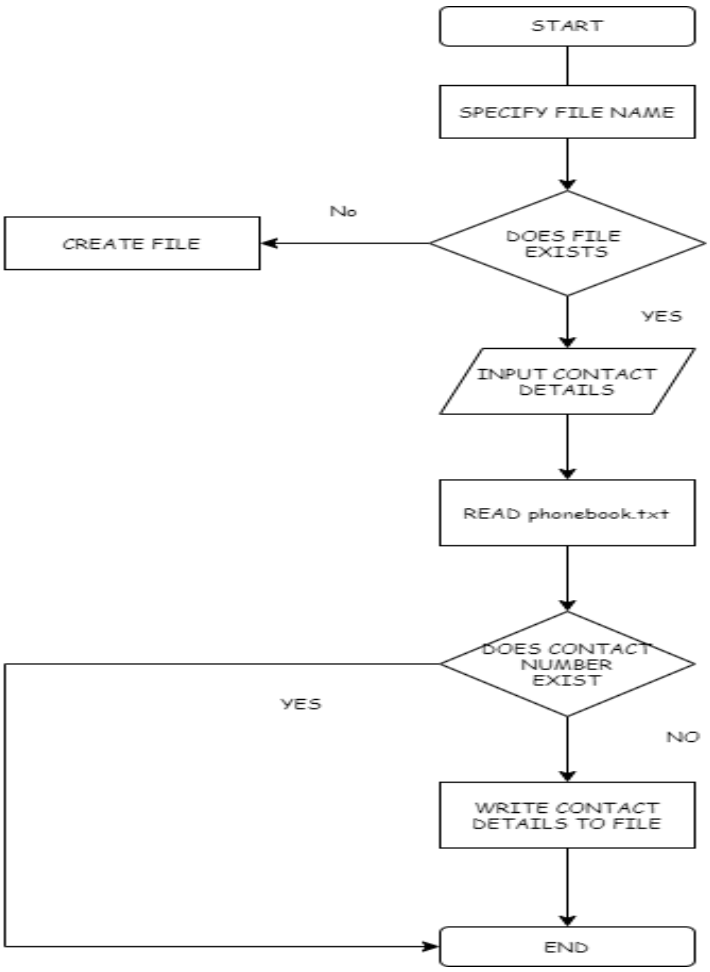
Key Field should be Phone Number, hence no two records can have same phone number

2.1) Design

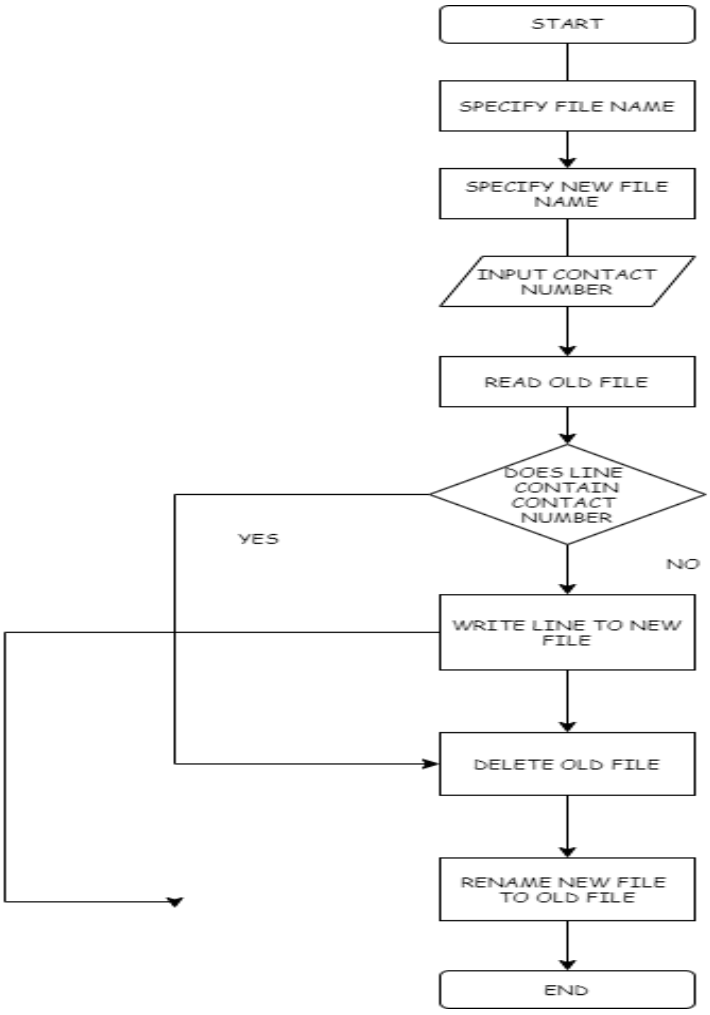
Flowchart for the Menu:



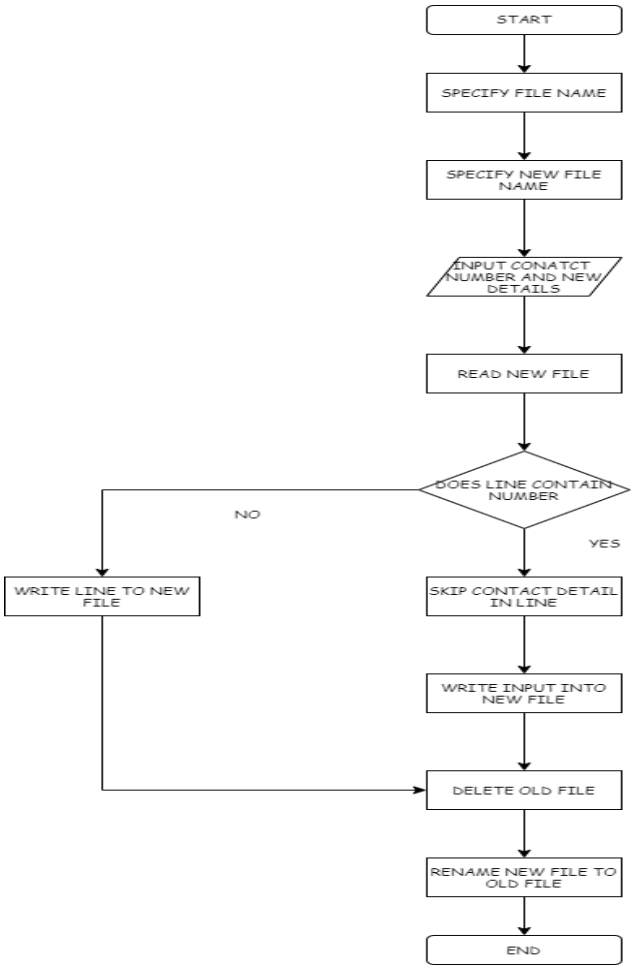
Flowchart for Add Contact:



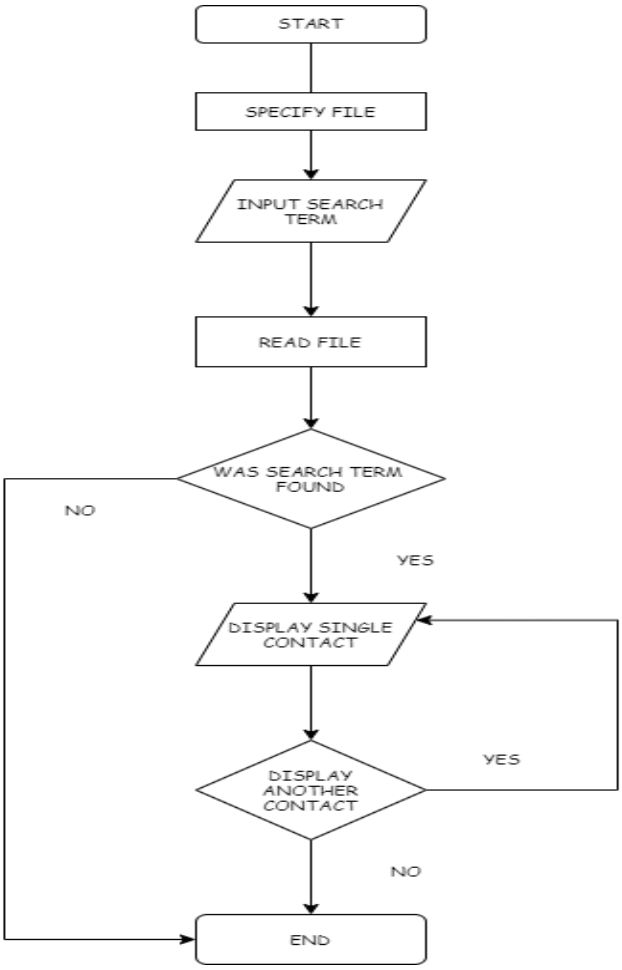
Flowchart for Delete Contact:



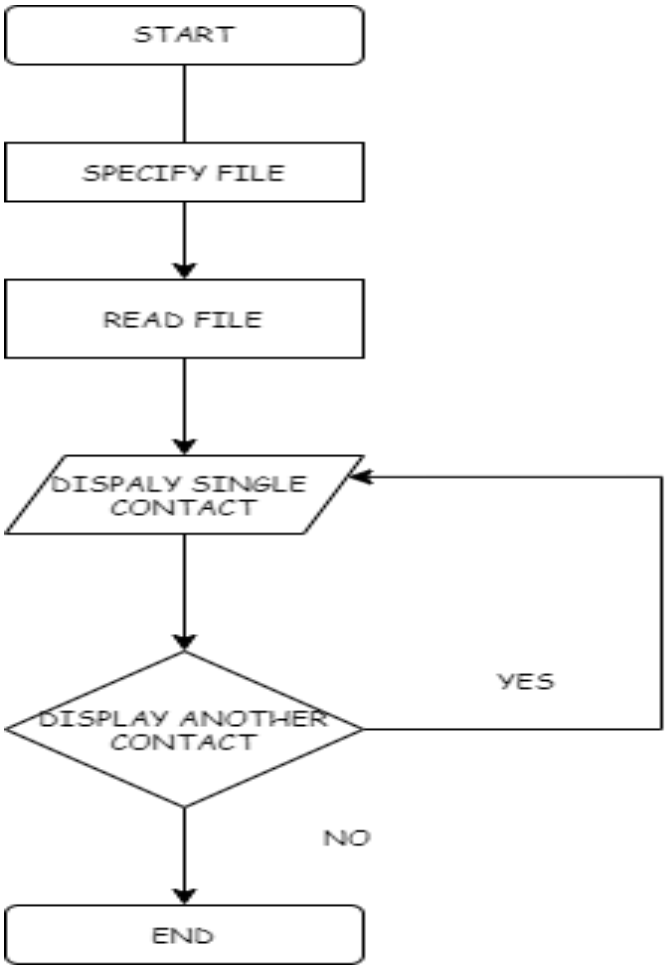
Flowchart for Modify Contact:



Flowchart for Search Contact:



Flowchart for Display Contact:


START
SPECIFY FILE
READ FILE
DISPALY SINGLE CONTACT
DISPLAY ANOTHER CONTACT
YES
NO
END

CHAPTER THREE

IMPLEMENTATION

3.1 Language Justification

Java has significant advantages over other languages that makes it suitable for just about any programming task

The following are advantages of using Java

- Java is object-oriented which allows us to create reusable codes
- Java is platform-independent, The ability to run the same program on many different systems

3.2 System Requirement

The requirement needed for the program to run are:

3.2.1 Hardware Requirements

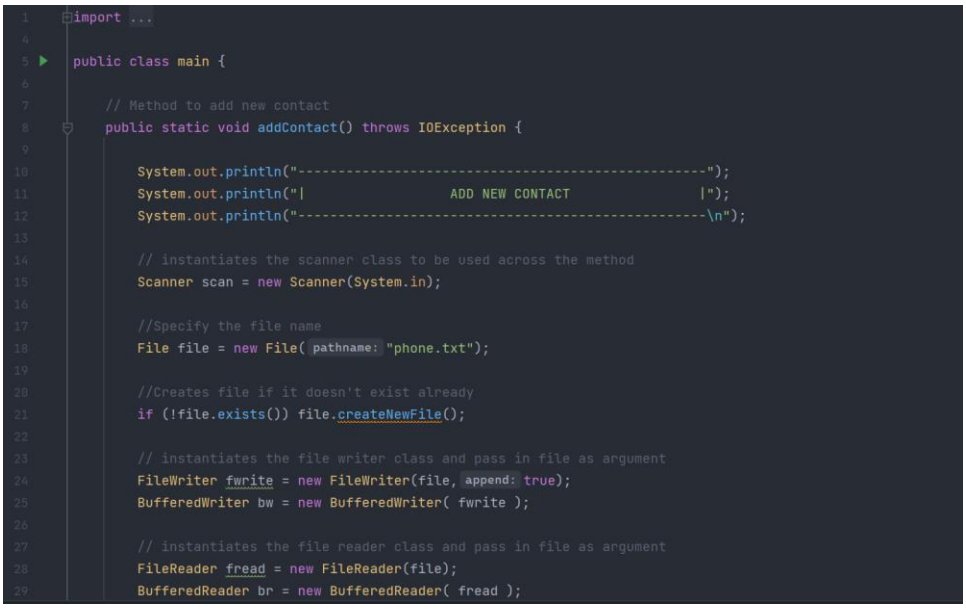
- A minimum computer system that will help you access all the tools in the program is a Pentium 166 or better
- 128 Megabytes of RAM or better

3.2.2 Software Requirements

- Java / J2EE / .NET / C / C++
- Notepad or any Java enabled IDE

Appendix a – Screenshots

Code:



```
import ...

public class main {

    // Method to add new contact
    public static void addContact() throws IOException {

        System.out.println("-----------------------------------------------------");
        System.out.println("|                  ADD NEW CONTACT                  |");
        System.out.println("-----------------------------------------------------\n");

        // instantiates the scanner class to be used across the method
        Scanner scan = new Scanner(System.in);

        //Specify the file name
        File file = new File( pathname: "phone.txt");

        //Creates file if it doesn't exist already
        if (!file.exists()) file.createNewFile();

        // instantiates the file writer class and pass in file as argument
        FileWriter fwrite = new FileWriter(file, append: true);
        BufferedWriter bw = new BufferedWriter( fwrite );

        // instantiates the file reader class and pass in file as argument
        FileReader fread = new FileReader(file);
        BufferedReader br = new BufferedReader( fread );
```

```

        // boolean variable to be used to check if contact exists
        boolean contactExists = false;

        // take in contact details
        System.out.println("Enter First Name: ");
        fname = scan.nextLine();

        System.out.println("Enter Last Name: ");
        lname = scan.nextLine();

        System.out.println("Enter Address: ");
        address = scan.nextLine();

        System.out.println("Enter Phone Number: ");
        number = scan.nextLine();

        System.out.println("Enter City: ");
        city = scan.nextLine();

        // loop through the file to check if contact already exists
        while( ( record = br.readLine() ) != null ) {
            // instantiate the stringtokenizer class to be used to split details from the line been read from
            StringTokenizer st = new StringTokenizer(record, delim: ",");

            // convert boolean contactexists to true when a number in the file matches the number inputted
            if (st.nextToken().equals(number)) {
```

```
57
58              // convert boolean contactexists to true when a number in the file matches the number
59              if (st.nextToken().equals(number)) {
60                  contactExists = true;
61              }
62          }
63
64          // add contact to file if it doesn't exist in the file
65          if (!contactExists) {
66              // write to file
67              bw.write( str: number+","+fname+","+lname+","+address+","+city);
68              bw.flush();
69              bw.newLine();
70              // close buffered writer
71              bw.close();
72              System.out.println("Contact saved successfully!");
73          } else {
74              System.out.println("Phone Number already exists!!!");
75          }
76
77          System.out.println("");
78          System.out.println("Exiting Contact Save...");
79
80      }
81
82      // Method to delete contact
83      public static void deleteContact() throws IOException {
```

```
419
420     public static void main(String[] args) throws IOException {
421
422         // instantiates the scanner class to be used across the method
423         Scanner scan = new Scanner(System.in);
424
425         // while loop to display option for user to pick from
426         mainLoop: while(true) {
427             System.out.println("-----------------------------------------------------");
428             System.out.println("|                  E-PROJECT PHONEBOOK               |");
429             System.out.println("-----------------------------------------------------\n");
430
431             System.out.println("1 ===> Add contact \n" +
432                     "2 ===> Delete contact \n" +
433                     "3 ===> Modify contact \n" +
434                     "4 ===> Search for contact \n" +
435                     "5 ===> Display contacts \n" +
436                     "6 ===> Get number of contacts \n" +
437                     "7 ===> Exit");
438
439
440             System.out.println("-----------------------------------------------------\n");
441
442             // take in selected option
443             System.out.println("Enter selection: ");
444             int option = scan.nextInt();
445             System.out.println("");
```

```
439
440             System.out.println("-------------------------------------------------------\n");
441
442             // take in selected option
443             System.out.println("Enter selection: ");
444             int option = scan.nextInt();
445             System.out.println("");
446
447             // run methods as per user selection
448             switch (option) {
449                 case 1:
450                     addContact();
451                     break;
452                 case 2:
453                     deleteContact();
454                     break;
455                 case 3:
456                     modifyContact();
457                     break;
458                 case 4:
459                     searchContact();
460                     break;
461                 case 5:
462                     displayContacts();
463                     break;
464                 case 6:
465                     contactCount();
```

Interface :

```
"C:\Program Files\Java\jdk1.8.0_40\bin\java.exe" ...
-----------------------------------------------------
|                E-PROJECT PHONEBOOK              |
-----------------------------------------------------

1 ===> Add contact
2 ===> Delete contact
3 ===> MAodify contact
4 ===> Search for contact
5 ===> Display contacts
6 ===> Get number of contacts
7 ===> Exit
-----------------------------------------------------

Enter selection:
```

```
1 ===> Add contact
2 ===> Delete contact
3 ===> MAodify contact
4 ===> Search for contact
5 ===> Display contacts
6 ===> Get number of contacts
7 ===> Exit
--------------------------------------------------

Enter selection:
1

--------------------------------------------------
|                 ADD NEW CONTACT                |
--------------------------------------------------

Enter First Name:
Dave
Enter Last Name:
James
Enter Address:
Aiport Road
Enter Phone Number:
09065443567
Enter City:
```

```
7 ===> Exit
--------------------------------------------------

Enter selection:
4

--------------------------------------------------
|                SEARCH FOR CONTACTS              |
--------------------------------------------------

Enter search term
dave

--------------------
|   CONTACT INFO   |
--------------------

Number: 09065443567
First name: Dave
Last name: James
Address: Aiport Road
City: Achina

Show next contact? (y/n)
```

Appendix a – Source Code

ADD CONTACT FUNCTION

```
public static void addContact() throws IOException {

    System.out.println("-----------------------------------------------------");
    System.out.println("|                  ADD NEW CONTACT                  |");
    System.out.println("-----------------------------------------------------\n");

    // instantiates the scanner class to be used across the method
    Scanner scan = new Scanner(System.in);

    //Specify the file name
    File file = new File("phone.txt");

    //Creates file if it doesn't exist already
    if (!file.exists()) file.createNewFile();

    // instantiates the file writer class and pass in file as argument
    FileWriter fwrite = new FileWriter(file,true);
    BufferedWriter bw = new BufferedWriter( fwrite );
```

```
    // instantiates the file reader class and pass in file as argument
    FileReader fread = new FileReader(file);
    BufferedReader br = new BufferedReader( fread );

    // declare variables to be used for the operation
    String fname, lname, address, city , number, record;

    // boolean variable to be used to check if contact exists
    boolean contactExists = false;

    // take in contact details
    System.out.println("Enter First Name: ");
    fname = scan.nextLine();

    System.out.println("Enter Last Name: ");
    lname = scan.nextLine();

    System.out.println("Enter Address: ");
    address = scan.nextLine();

    System.out.println("Enter Phone Number: ");
    number = scan.nextLine();

    System.out.println("Enter City: ");
    city = scan.nextLine();

    // loop through the file to check if contact already exists
    while( ( record = br.readLine() ) != null ) {
        // instantiate the stringtokenizer class to be used to split details from the
line been read from the file
        StringTokenizer st = new StringTokenizer(record,",");

        // convert boolean contactexists to true when a number in the file matches the
number inputted
        if (st.nextToken().equals(number)) {
            contactExists = true;
        }
    }

    // add contact to file if it doesn't exist in the file
    if (!contactExists) {
        // write to file
        bw.write(number+","+fname+","+lname+","+address+","+city);
        bw.flush();
        bw.newLine();
        // close buffered writer
        bw.close();
        System.out.println("Contact saved successfully!");
    } else {
        System.out.println("Phone Number already exists!!!");
    }
```

```
    System.out.println("");
    System.out.println("Exiting Contact Save...");

}
```

DELETE CONTACT FUNCTION

```
public static void deleteContact() throws IOException {

    System.out.println("-----------------------------------------------------");
    System.out.println("|                   DELETE CONTACT                  |");
    System.out.println("-----------------------------------------------------\n");

    // instantiates the scanner class to be used across the method
    Scanner scan = new Scanner(System.in);

    // declare variables to be used for operation
    String number, record;

    // boolean variable to be used to check if contact exists
    boolean found = false;

    // create new file for the operation
    File newFile = new File("new-phone.txt");

    // specify file name
    File file = new File("phone.txt");

    // instantiates the file reader class and pass in file as argument
    FileReader fread = new FileReader(file);
    BufferedReader br = new BufferedReader( fread );

    // instantiates the file writer class and pass in file as argument
    FileWriter fwrite = new FileWriter(newFile);
    BufferedWriter bw = new BufferedWriter( fwrite );

    System.out.println("Enter Contact Number: ");
    number = scan.nextLine();

    // loop through to search for number to be deleted
    while( ( record = br.readLine() ) != null ) {
        // instantiate the stringtokenizer class to be used to split details from the
line been read from the file
        StringTokenizer st = new StringTokenizer(record,",");
```

```
        // check if inputted number matches number of current contact that's being looped
through
        if (st.nextToken().equals(number)) {
            // convert boolean found to true when contact to be deleted is found
            found = true;

            // continue (skip the current loop) the loop when number is found, so it
won't be written to the new file
            continue;
        }

        // write records that don't match the number to be deleted to the new file
        bw.write(record);
        bw.flush();
        bw.newLine();
    }

    // close buffered reader
    br.close();
    // close buffered writer
    bw.close();

    // delete the old file
    file.delete();
    // rename new file to old file
    newFile.renameTo(file);

    // print success or error message if number was found
    if (!found) System.out.println("No contact with number "+number+" found!!");
    else System.out.println(number+" deleted successfully!");

    System.out.println("");
    System.out.println("Exiting Contact Delete...");
}
```

MODIFY CONTACT FUNCTION

```
public static void modifyContact () throws IOException{

    System.out.println("-----------------------------------------------------");
    System.out.println("|                   MODIFY CONTACT                  |");
    System.out.println("-----------------------------------------------------\n");

    // instantiates the scanner class to be used across the method
    Scanner scan = new Scanner(System.in);

    // declare variables to be used for operation
    String fname, lname, address, city, number, record;
```

```
    // boolean variable to be used to check if contact exists
    boolean found = false;

    // specify file name
    File file = new File("phone.txt");

    // create new file for the operation
    File newFile = new File("new_phone.txt");

    // instantiates the file reader class and pass in file as argument
    FileReader fread = new FileReader(file);
    BufferedReader br = new BufferedReader( fread );

    // instantiates the file writer class and pass in file as argument
    FileWriter fwrite = new FileWriter(newFile);
    BufferedWriter bw = new BufferedWriter( fwrite );

    // take in contact details
    System.out.println("Enter contact number to be modified: ");
    number = scan.nextLine();

    System.out.println("Enter new Firstname: ");
    fname = scan.nextLine();

    System.out.println("Enter new Lastname: ");
    lname = scan.nextLine();

    System.out.println("Enter new Address: ");
    address = scan.nextLine();

    System.out.println("Enter new City: ");
    city = scan.nextLine();

    // loop through to search for number to be modified
    while( ( record = br.readLine() ) != null ) {
        // instantiate the stringtokenizer class to be used to split details from the
line been read from the file
        StringTokenizer st = new StringTokenizer(record,",");

        // check if inputted number matches number of current contact that's being looped
through
        if (st.nextToken().equals(number)) {
            // convert boolean found to true when contact to be modified is found
            found = true;

            // write new contact details to the new file created
            bw.write(number+","+fname+","+lname+","+address+","+city);
            bw.flush();
            bw.newLine();

            // continue (skip the current loop) the loop when number is found, so it
```

```
won't be written to the new file
            continue;
        }

        // write records that don't match the number to be modified to the new file
        bw.write(record);
        bw.flush();
        bw.newLine();
    }

    // close buffered reader
    br.close();
    // close buffered writer
    bw.close();

    // delete the old file
    file.delete();
    // rename new file to old file
    newFile.renameTo(file);

    // print success or error message if number was found
    if (!found) System.out.println("No contact with number "+number+" found!!");
    else System.out.println(number+" Modified successfully!");

    System.out.println("");
    System.out.println("Exiting Contact Update...");
}
```

ADD CONTACT FUNCTION

```
public static void searchContact() throws IOException {

    System.out.println("-----------------------------------------------------------");
    System.out.println("|                 SEARCH FOR CONTACTS                     |");
    System.out.println("-----------------------------------------------------------\n");

    // instantiates the scanner class to be used across the method
    Scanner scan = new Scanner(System.in);

    // declare variables to be used for operation
    String term, record, choice;

    // boolean variable to be used to check if any contact exists
    boolean found = false;

    // instantiates the file reader class and pass in file as argument
    FileReader fread = new FileReader("phone.txt");
    BufferedReader br = new BufferedReader( fread );
```

```
    // take in search term
    System.out.println("Enter search term");
    term = scan.nextLine();
    System.out.println("");

    // loop through to searching for numbers to be displayed
    subLoop: while( ( record = br.readLine() ) != null) {

        // instantiate the stringtokenizer class to be used to split details from the
line been read from the file
        StringTokenizer st = new StringTokenizer(record,",");

        // check if record contains the search term
        if( record.toLowerCase().contains(term.toLowerCase()) ) {

            // convert boolean found to true when term searched for is found
            found = true;

            // display current contact being looped through info
            System.out.println("--------------------");
            System.out.println("|   CONTACT INFO   |");
            System.out.println("--------------------\n");

            System.out.println("Number: "+st.nextToken());
            System.out.println("First name: "+st.nextToken());
            System.out.println("Last name: "+st.nextToken());
            System.out.println("Address: "+st.nextToken());
            System.out.println("City: "+st.nextToken()+"\n");


            // option to move to next record or quit
            System.out.println("Show next contact? (y/n)");
            choice = scan.nextLine();

            switch (choice) {
                case "y" :
                case "Y" :
                    continue subLoop;
                case "n" :
                case "N" :
                    break subLoop;
                default:
                    System.out.println("Wrong Input entered!");
            }
        }

    }

    // print error message if no contact was found
    if (!found) System.out.println("No contact matches "+term);
```

```
    // close buffered reader
    br.close();

    System.out.println("");
    System.out.println("Exiting Contact Search...");
}
```

DISPLAY CONTACT FUNCTION

```
public static void displayContacts () throws IOException {

    System.out.println("-----------------------------------------------------");
    System.out.println("|                  DISPLAY CONTACTS                 |");
    System.out.println("-----------------------------------------------------\n");

    // instantiates the scanner class to be used across the method
    Scanner scan = new Scanner(System.in);

    // declare variables to be used for operation
    String record,choice;

    // boolean variable to be used to check if any contact exists
    boolean found = false;

    // instantiates the file reader class and pass in file as argument
    FileReader fread = new FileReader("phone.txt");
    BufferedReader br = new BufferedReader( fread );

    // loop through all numbers to be displayed
    subLoop: while( ( record = br.readLine() ) != null) {

        // instantiate the stringtokenizer class to be used to split details from the
line been read from the file
        StringTokenizer st = new StringTokenizer(record,",");

        // convert boolean found to true when contacts exist
        found = true;

        // display current contact being looped through info
        System.out.println("--------------------");
        System.out.println("|   CONTACT INFO   |");
        System.out.println("--------------------\n");

        System.out.println("Number: "+st.nextToken());
        System.out.println("First name: "+st.nextToken());
        System.out.println("Last name: "+st.nextToken());
        System.out.println("Address: "+st.nextToken());
```

```
            System.out.println("City: "+st.nextToken()+"\n");

            // option to move to next record or quit
            System.out.println("Show next contact? (y/n)");
            choice = scan.nextLine();

            switch (choice) {
                case "y" :
                case "Y" :
                    continue subLoop;
                case "n" :
                case "N" :
                    break subLoop;
                default:
                    System.out.println("Wrong Input entered!");
                    continue subLoop;
            }

        }

        // print error message if no contact was found
        if (!found) System.out.println("Contact List empty");

        // close buffered reader
        br.close();

        System.out.println("");
        System.out.println("Exiting Contacts Display...");
    }
```

COUNT CONTACT FUNCTION

```
public static void contactCount () throws IOException {

        System.out.println("-----------------------------------------------------");
        System.out.println("|                  NUMBER OF CONTACTS                |");
        System.out.println("-----------------------------------------------------\n");

        // instantiates the scanner class to be used across the method
        Scanner scan = new Scanner(System.in);

        // declare a count variable to be increased when contacts are found
        int count = 0;

        // instantiates the file reader class and pass in file as argument
        FileReader fread = new FileReader("phone.txt");
        BufferedReader br = new BufferedReader( fread );
```

```
    // loop through all numbers for counting
    subLoop: while( br.readLine() != null) {

        // increment count whenever a number is found
        count++;

    }

    // display number of contacts found
    System.out.println(count+" Contacts exists");
    br.close();

    System.out.println("");
    System.out.println("Exiting Contact Count...");
}
```

MENU

```
Scanner scan = new Scanner(System.in);

// while loop to display option for user to pick from
mainLoop: while(true) {
    System.out.println("-------------------------------------------------------");
    System.out.println("|                  E-PROJECT PHONEBOOK                 |");
    System.out.println("-------------------------------------------------------\n");

    System.out.println("1 ===> Add contact \n" +
            "2 ===> Delete contact \n" +
            "3 ===> Modify contact \n" +
            "4 ===> Search for contact \n" +
            "5 ===> Display contacts \n" +
            "6 ===> Get number of contacts \n" +
            "7 ===> Exit");


    System.out.println("-------------------------------------------------------\n");

    // take in selected option
    System.out.println("Enter selection: ");
    int option = scan.nextInt();
    System.out.println("");

    // run methods as per user selection
    switch (option) {
        case 1:
            addContact();
            break;
        case 2:
```

```
                deleteContact();
                break;
            case 3:
                modifyContact();
                break;
            case 4:
                searchContact();
                break;
            case 5:
                displayContacts();
                break;
            case 6:
                contactCount();
                break;
            case 7:
                System.out.println("Exiting Program....");
                break mainLoop;
            default:
                System.out.println("Invalid Input");
        }

        System.out.println("");
    }
```

TASK SHEETS

1) Create a menu for user to pick action to perform

2) Create Add contact Method for adding contacts

3) Create Delete contact Method for deleting contact

4) Create Modify contact Method for editing contact

5) Create Search contact Method for searching for contact

6) Create Display contact Method for displaying contact

7) Create Contact count Method for counting number of contacts in file

8) Go through code check for errors and bugs