

EE321 Lab 2

Reliable Client Server Communication

Exercise One

Aim:

The aim of this exercise is to create a server that holds a text file containing a phonebook. It allows clients to connect to the server and request a person's number. It then returns the number. The goal is to learn how to receive messages from a client and use these messages as required.

Procedure:

I began with the code that I had from the previous assignment; so, I had server code that allowed a client to connect. I then implemented a reader to read both an input from the client and from the phonebook txt file. This was done using the buffered reader java class.

```
BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
FileReader fileReader = new FileReader(fileName); //reads file
BufferedReader bufferedReader = new BufferedReader(fileReader);
```

The bufferedReader for the text file is so that the reader reads only a line at a time.

Then I wrote the code that would take the input of the client and compare it to the text file. If the name is found it will then print out the number that follows it.

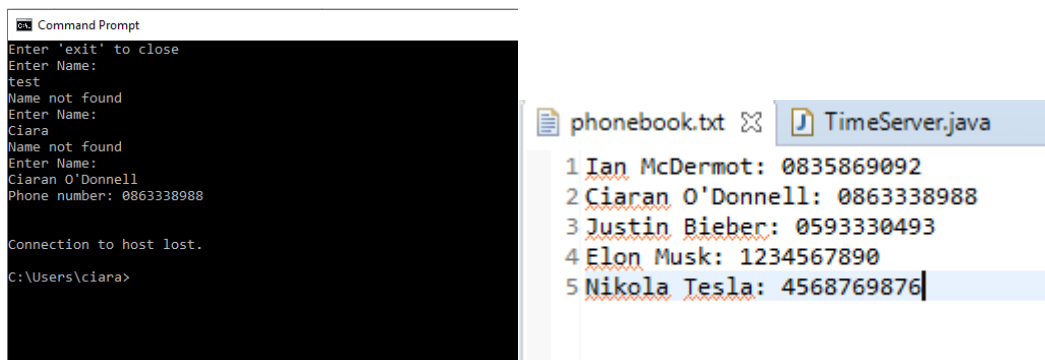
```
while((line = bufferedReader.readLine()) != null) { //reads through entire list
    if((requestname.equals(line.substring(0, requestname.length())) && (line.charAt(requestname.length()) == ':')) {
        out.println("Phone number: " + line.substring(requestname.length()+2)); //print number
    }
}
```

I had to also check that the character after the name was ':' in order to be sure a full name is entered.

With this the basic functionality of the server was working. With this done I decided to add some quality of life implementations. Such as not being kicked off the server after entering an incorrect name but rather being allowed enter another, as well as including an exit command to disconnect with the server.

Results:

With all this done my server worked as expected, found the numbers of all inputs in the file with no errors.

The image shows a screenshot of a Java application window titled 'TimeServer.java'. On the left, there is a 'Command Prompt' window showing the interaction with the server. The user enters 'test' and 'Ciaran', both of which result in 'Name not found'. Then, the user enters 'Ciaran O'Donnell', and the server responds with 'Phone number: 0863338988'. The prompt then shows 'Connection to host lost.' and the user's directory 'C:\Users\ciara>'. On the right, the 'phonebook.txt' file is displayed, containing a list of names and phone numbers: 1 Ian McDermot: 0835869092, 2 Ciaran O'Donnell: 0863338988, 3 Justin Bieber: 0593330493, 4 Elon Musk: 1234567890, and 5 Nikola Tesla: 4568769876.

With this working I moved on to the second exercise of this assignment.

Source Code:

```
import java.io.*;
import java.net.*;

public class Phone_Server {

    public static void main(String args[]) throws UnknownHostException, IOException {

        String fileName = "phonebook.txt";
        String line = null;
        Boolean found = false;

        try {
            ServerSocket myServerSocket = new ServerSocket(1234); //opens server side socket
            Socket connectedClientSocket = null; //create uninitialized client socket

            while(true) { //keep server on

                connectedClientSocket = myServerSocket.accept();//wait for client to connect
                PrintWriter out = new PrintWriter(connectedClientSocket.getOutputStream(), true);//print writer to client
                BufferedReader in = new BufferedReader(new InputStreamReader(connectedClientSocket.getInputStream())); //reads from client
                out.println("Enter 'exit' to close"); //prints exit message

                while(found==false) { //loops until exit or name found
                    out.println("Enter Name: "); //asks to enter name
                    String requestname = in.readLine(); //reads entry
                    if(requestname.equals("exit")) {
                        break; //if entry exit it exits the loop
                    }

                    FileReader fileReader = new FileReader(fileName); //reads file
                    BufferedReader bufferedReader = new BufferedReader(fileReader); //buffered reader so it reads a line at a time
                    while((line = bufferedReader.readLine()) != null) { //reads through entire list
                        if((requestname.equals(line.substring(0, requestname.length())) && (line.charAt(requestname.length()) == ':')) {

                            //if name is found
                            out.println("Phone number: " + line.substring(requestname.length()+2)); //print number
                            found = true;
                        }
                    }
                    if(found == false) {
                        out.println("Name not found"); //if name is not found
                    }
                }

                connectedClientSocket.close(); //closes connection to the client
                found=false;
            }
        }

        catch(FileNotFoundException ex) {
            System.out.println(ex);
        }

        catch(IOException ex) {
            System.out.println(ex);
        }
    }
}
```

Exercise Two

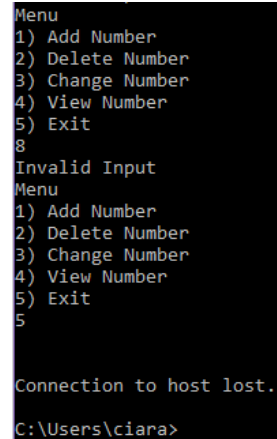
Aim:

The aim of this exercise is to add to our phonebook server to implement added functionality. This functionality includes being able to insert new numbers for a person, delete a number for a person and to change a number, as well as listing there numbers as before.

Procedure:

I began by making a new class to hold this server code and transferring all my old code into this new class. I then wrote a small function to display a menu to the client. This menu used simple number inputs for the user to choose the function they want to use.

```
static void menu(PrintWriter tout) {  
    tout.println("Menu");  
    tout.println("1) Add Number");  
    tout.println("2) Delete Number");  
    tout.println("3) Change Number");  
    tout.println("4) View Number");  
    tout.println("5) Exit");  
}
```



```
Menu  
1) Add Number  
2) Delete Number  
3) Change Number  
4) View Number  
5) Exit  
8  
Invalid Input  
Menu  
1) Add Number  
2) Delete Number  
3) Change Number  
4) View Number  
5) Exit  
5  
  
Connection to host lost.  
C:\Users\ciara>
```

The first method I created was a list numbers method as I knew this would come in handy for the other methods in my code. I did this by simply reading the text file and printing each line to the client until the line was NULL.

```
while((line = bufferedReader.readLine()) != null) { //reads through entire list  
    tout.println(tmp + " " + line);  
    tmp++;  
}
```

I then worked on implementing the add number feature, it began by listing the numbers to the client asking which person to add the number to. When selected the server requests the number from the client. It then adds this number to the end of that person's line in the file. It does this by copying the file line by line to a new file until it is at the required line, then appends this line and then overwrites the original file.

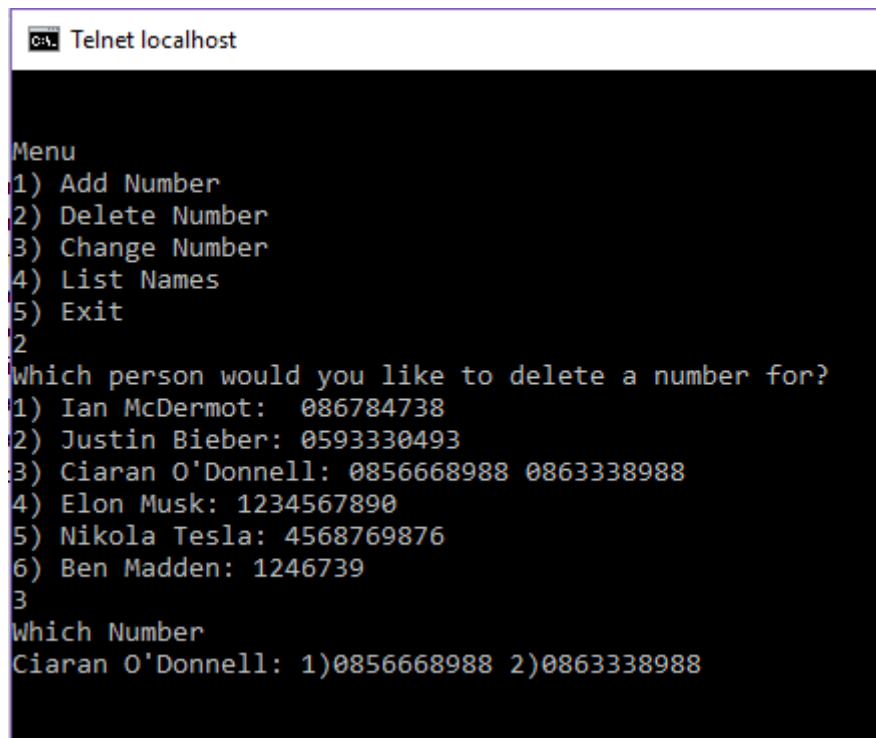
```
while ((line = br.readLine()) != null) {  
    if(lineCount==req-1){  
        pw.println(line + " " + newnum);  
    }  
    else {  
        pw.println(line);  
    }  
}
```

It also allows the user to add a new name. If the client chooses to add a new name it must input the name, then the number, and then the server will append it to the end of the file.

```
FileWriter fw = new FileWriter(fileName,true); //the true will append the new data  
fw.write("\n" + newname + ": " + newnum);//appends the string to the file  
fw.close();
```

The next step was to implement the delete number function. I realised that to deal with names that had more than one number I would need a function to list all their numbers. To do this I created a new function to list numbers, (I changed the name of the previous list numbers function to list names). I did this by stepping through the file until at the required name similar to the add function, then stepping through the line a character at a time to find the start of each number and list them.

```
while((line = bufferedReader.readLine()) != null) { //reads through entire list
if(tmp == lineNum) {
    String newtext;
    newtext = "" + line.charAt(0);
    for(int j=1; j<line.length(); j++) {
        newtext += line.charAt(j);
        if((line.charAt(j) == ' ') && Character.isDigit(line.charAt(j+1))) {
            newtext += phoneind + " ";
            phoneind++;
        }
    }
    tout.println(newtext); //prints the lines to client
}
}
```



The screenshot shows a Telnet session titled 'Telnet localhost'. The user has entered '2' to select the 'Delete Number' option from the menu. The program then displays a list of names and their associated phone numbers, numbered 1 through 6. The user has entered '3' to select the 'Which Number' option. The program then displays the phone numbers for the selected name, Ciaran O'Donnell, numbered 1 and 2.

```
Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
2
Which person would you like to delete a number for?
1) Ian McDermot: 086784738
2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988 0863338988
4) Elon Musk: 1234567890
5) Nikola Tesla: 4568769876
6) Ben Madden: 1246739
3
Which Number
Ciaran O'Donnell: 1)0856668988 2)0863338988
```

With this done I then had to take an input from the client, again I stepped through the line and when I came to the specified number I simply did not add it to the replacement line.

```
for(int j=1; j<line.length(); j++) {
    newline += line.charAt(j);
    if((line.charAt(j) == ' ') && Character.isDigit(line.charAt(j+1))) {
        phoneind++;
        if(phoneind == num) {
            j++;
            while((j<line.length()) && (line.charAt(j) != ' ')) {
                j++;
            }
        }
    }
}
pw.println(newline);
```

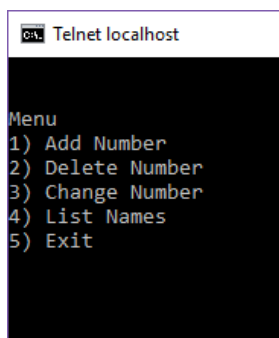
Finally, I had to implement a function to allow the client to change the number of a user. I did this by again reusing previous code. I took the code for deleting a number from a user and allowed the client to instead input a number to replace the number being removed. This simply meant reading an extra line from the client (the new number) and when the number to be replaced is found in the line simply add the new number to the string instead of the old.

```
tout.println("Enter correct number:");
newnum = tin.readLine();

for(int j=1; j<line.length(); j++) {
    newline += line.charAt(j);
    if((line.charAt(j) == ' ')&&Character.isDigit(line.charAt(j+1))) {
        phoneind++;
        if(phoneind == num) {
            j++;
            newline += newnum; //this is where the new number is added
            while((j<line.length()) && (line.charAt(j) != ' ')) {
                j++;
            }
        }
    }
}
```

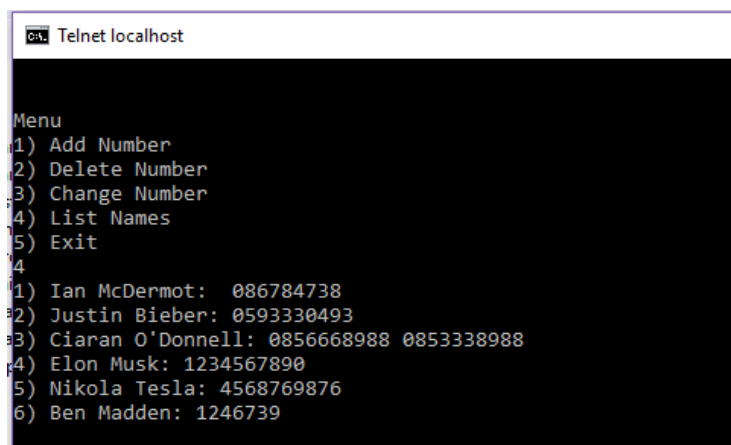
Results:

With these functions wrote it was time to test. Again, I connected using the telnet feature of windows. I was greeted with the menu I wrote earlier:

A screenshot of a Telnet session window titled "Telnet localhost". The window shows a menu with five options: 1) Add Number, 2) Delete Number, 3) Change Number, 4) List Names, and 5) Exit.

```
Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
```

I decided to first simply have the code print out the current names and numbers on the file:

A screenshot of a Telnet session window titled "Telnet localhost". The window shows the same menu as the previous screenshot, but with an additional option, 6) Ben Madden: 1246739, added to the list. The list of names and numbers is displayed as follows:

```
Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
6) Ben Madden: 1246739
```

This worked as expected. Next, I decided to use the add number feature, First adding a number to a name already in the phonebook and then a completely new entry, both worked as expected:

Telnet localhost	Telnet localhost
6) Ben Madden: 1246739	Menu
Menu	1) Add Number
1) Add Number	2) Delete Number
2) Delete Number	3) Change Number
3) Change Number	4) List Names
4) List Names	5) Exit
5) Exit	1
1	Which person would you like to add a number for?
Which person would you like to add a number for?	1) Ian McDermot: 086784738
1) Ian McDermot: 086784738	2) Justin Bieber: 0593330493
2) Justin Bieber: 0593330493	3) Ciaran O'Donnell: 0856668988 0853338988
3) Ciaran O'Donnell: 0856668988 0853338988	4) Elon Musk: 1234567890
4) Elon Musk: 1234567890	5) Nikola Tesla: 4568769876 3295837620
5) Nikola Tesla: 4568769876	6) Ben Madden: 1246739
6) Ben Madden: 1246739	7) Add new name
7) Add new name	7
5	Enter Name:
Enter Number:	Axel Rose
3295837620	Enter Number:
	478932795
Menu	Menu
1) Add Number	1) Add Number
2) Delete Number	2) Delete Number
3) Change Number	3) Change Number
4) List Names	4) List Names
5) Exit	5) Exit
4	4
1) Ian McDermot: 086784738	1) Ian McDermot: 086784738
2) Justin Bieber: 0593330493	2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988 0853338988	3) Ciaran O'Donnell: 0856668988 0853338988
4) Elon Musk: 1234567890	4) Elon Musk: 1234567890
5) Nikola Tesla: 4568769876 3295837620	5) Nikola Tesla: 4568769876 3295837620
6) Ben Madden: 1246739	6) Ben Madden: 1246739
	7) Axel Rose: 478932795
Menu	Menu
1) Add Number	1) Add Number
2) Delete Number	2) Delete Number
3) Change Number	3) Change Number
4) List Names	4) List Names
5) Exit	5) Exit

The next test was to delete a number, I deleted the first of Nikola Tesla's numbers and the second of Ciaran's to test both deleting at both ends of a line.

Telnet localhost	Telnet localhost
Menu	Menu
1) Add Number	1) Add Number
2) Delete Number	2) Delete Number
3) Change Number	3) Change Number
4) List Names	4) List Names
5) Exit	5) Exit
2	2
Which person would you like to delete a number for?	Which person would you like to delete a number for?
1) Ian McDermot: 086784738	1) Ian McDermot: 086784738
2) Justin Bieber: 0593330493	2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988 0853338988	3) Ciaran O'Donnell: 0856668988 0853338988
4) Elon Musk: 1234567890	4) Elon Musk: 1234567890
5) Nikola Tesla: 4568769876 3295837620	5) Nikola Tesla: 3295837620
6) Ben Madden: 1246739	6) Ben Madden: 1246739
7) Axel Rose: 478932795	7) Axel Rose: 478932795
5	3
Which Number	Which Number
Nikola Tesla: 1)4568769876 2)3295837620	Ciaran O'Donnell: 1)0856668988 2)0853338988
1	2
Menu	Menu
1) Add Number	1) Add Number
2) Delete Number	2) Delete Number
3) Change Number	3) Change Number
4) List Names	4) List Names
5) Exit	5) Exit
4	4
1) Ian McDermot: 086784738	1) Ian McDermot: 086784738
2) Justin Bieber: 0593330493	2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988 0853338988	3) Ciaran O'Donnell: 0856668988
4) Elon Musk: 1234567890	4) Elon Musk: 1234567890
5) Nikola Tesla: 3295837620	5) Nikola Tesla: 3295837620
6) Ben Madden: 1246739	6) Ben Madden: 1246739
7) Axel Rose: 478932795	7) Axel Rose: 478932795

I needed to test the change number function again this worked as I intended:

```
Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
3
Which person would you like to edit a number for?
1) Ian McDermot: 086784738
2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988
4) Elon Musk: 1234567890
5) Nikola Tesla: 3295837620
6) Ben Madden: 1246739
7) Axel Rose: 478932795
1
Which Number would you like to edit?
Ian McDermot: 1)086784738
1
Enter correct number:
785903768930

Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
4
1) Ian McDermot: 785903768930
2) Justin Bieber: 0593330493
3) Ciaran O'Donnell: 0856668988
4) Elon Musk: 1234567890
5) Nikola Tesla: 3295837620
6) Ben Madden: 1246739
7) Axel Rose: 478932795

Menu
```

Finally, I used the exit command and then tried to reconnect to ensure the server allowed multiple connections one after the other:

```
Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
5

Connection to host lost.

C:\Users\ciara>telnet localhost 1234
```

```
C:\> Telnet localhost

Menu
1) Add Number
2) Delete Number
3) Change Number
4) List Names
5) Exit
```

With all of these tests done I was happy with the functionality of my server and deemed it complete.

Source code:

```
import java.io.*;
import java.net.*;

public class Phone_Server2 {
    public static void main(String args[]) throws UnknownHostException, IOException {

        String fileName = "phonebook2.txt";
        String request = null;
        String line = null;
        Boolean exit = false;
        PrintWriter out = null;
        BufferedReader in = null;

        try {
            ServerSocket myServerSocket = new ServerSocket(1234); //opens server side socket
            Socket connectedClientSocket = null; //create uninitialized client socket

            while(true) { //keep server on

                connectedClientSocket = myServerSocket.accept(); //wait for client to connect
                out = new PrintWriter(connectedClientSocket.getOutputStream(), true); //print writer to client
                in = new BufferedReader(new InputStreamReader(connectedClientSocket.getInputStream())); //reads from client

                while(exit == false) { //loops until exit
                    out.println("\n"); //creates a two line gap
                    menu(out);
                    request = in.readLine(); //reads entry
                    switch(request){
                        case "1":
                            addNumber(out, in);
                            break;
                        case "2": //runs the required function
                            delNumber(out, in);
                            break;
                        case "3":
                            chngNumber(out, in);
                            break;
                        case "4":
                            ListNames(out, in);
                            break;
                        case "5":
                            exit = true;
                            break;
                        default:
                            out.println("Invalid Input");
                            break;
                    }
                }

                connectedClientSocket.close(); //closes connection to the client
                exit=false;
            }

            catch(FileNotFoundException ex) {
                System.out.println(ex);
            }

            catch(IOException ex) {
                System.out.println(ex);
            }
        }

        static void menu(PrintWriter tout) {
            tout.println("Menu"); //simple prints menu
            tout.println("1) Add Number");
            tout.println("2) Delete Number");
            tout.println("3) Change Number");
            tout.println("4) List Names");
            tout.println("5) Exit");
        }

        static void addNumber(PrintWriter tout, BufferedReader tin) throws IOException {
            String fileName = "phonebook2.txt";
            tout.println("Which person would you like to add a number for?");
            int listlength = ListNames(tout, tin); //lists names and returns length of list
            listlength++; //increments length for following
            tout.println(listlength + ") Add new name"); //adds option to add name to end of list
            int req = Integer.parseInt(tin.readLine()); //reads what the user would like to do
            if(req==listlength) { //if user adding new name
                tout.println("Enter Name:"); //requests name
                String newname = tin.readLine();
                tout.println("Enter Number:");
                String newnum = tin.readLine(); //requests number
                FileWriter fw = new FileWriter(fileName,true); //the true will append the new data
                fw.write("\n" + newname + ": " + newnum); //appends the string to the file
                fw.close();
            }
            else {
                tout.println("Enter Number:"); //asks for new number
                String newnum = tin.readLine();
                File file = new File(fileName);
                File temp = File.createTempFile("temp-file-name", ".tmp"); //temporary file
                BufferedReader br = new BufferedReader(new FileReader( file ));
                PrintWriter pw = new PrintWriter(new FileWriter( temp ));
            }
        }
    }
}
```



```

        String line;
        int lineCount = 0;
        while ((line = br.readLine()) != null) {
            if (lineCount == req - 1) { //finds line and appends new number to the end of it
                pw.println(line + " " + newnum);
            }
            else {
                pw.println(line); //otherwise just adds line to file
            }
            lineCount++;
        }
        br.close();
        pw.close();
        file.delete(); //replaces file with temp file
        temp.renameTo(file);
    }
}

static void delNumber(PrintWriter tout, BufferedReader tin) throws IOException {
    String fileName = "phonebook2.txt";
    tout.println("Which person would you like to delete a number for?");
    ListNames(tout, tin); //lists all names
    int req = Integer.parseInt(tin.readLine()); //retrieves int input for name
    int phoneind = 0;
    String newline;

    tout.println("Which Number");
    ListNumbers(tout, tin, req); //prints all that users numbers
    int num = Integer.parseInt(tin.readLine()); //retrieves int input for number

    File file = new File(fileName); //opens the file
    File temp = File.createTempFile("temp-file-name", ".tmp"); //opens temp file
    BufferedReader br = new BufferedReader(new FileReader(file)); //reads file line at a time
    PrintWriter pw = new PrintWriter(new FileWriter(temp)); //prints to temp file
    String line; //current line
    int lineCount = 0;
    while ((line = br.readLine()) != null) { //steps through file
        if (lineCount == req - 1) { //finds requested name
            newline = "" + line.charAt(0); //initializes the replacement line
            for (int j = 1; j < line.length(); j++) { //steps through line
                newline += line.charAt(j); //adding each character to newline as it does
                if ((line.charAt(j) == ' ') && Character.isDigit(line.charAt(j + 1))) { //finds a space followed by a digit
                    phoneind++; //increments how many numbers found
                    if (phoneind == num) { //if number is the number requested for del
                        j++;
                        while ((j < line.length()) && (line.charAt(j) != ' ')) { //steps through number not adding
                            j++; //it to replacement file
                        }
                    }
                }
            }
        }
        pw.println(newline); //prints replacement line to file
    }
    else {
        pw.println(line); //prints line to file
    }
    lineCount++;
}
br.close();
pw.close(); //closes reader and writer
file.delete();
temp.renameTo(file); //deletes old file and renames replacement file
}

static void chngNumber(PrintWriter tout, BufferedReader tin) throws IOException {
    String fileName = "phonebook2.txt";
    tout.println("Which person would you like to edit a number for?"); //all same as delete except where specified
    ListNames(tout, tin);
    int req = Integer.parseInt(tin.readLine());
    int phoneind = 0;
    String newline;
    String newnum;

    tout.println("Which Number would you like to edit?");
    ListNumbers(tout, tin, req);
    int num = Integer.parseInt(tin.readLine());

    tout.println("Enter correct number:");
    newnum = tin.readLine(); //enter new number

    File file = new File(fileName);
    File temp = File.createTempFile("temp-file-name", ".tmp");
    BufferedReader br = new BufferedReader(new FileReader(file));
    PrintWriter pw = new PrintWriter(new FileWriter(temp));
    String line;
    int lineCount = 0;
    while ((line = br.readLine()) != null) {
        if (lineCount == req - 1) {
            newline = "" + line.charAt(0);
            for (int j = 1; j < line.length(); j++) {
                newline += line.charAt(j);
                if ((line.charAt(j) == ' ') && Character.isDigit(line.charAt(j + 1))) {

```

```

        phoneind++;
        if(phoneind == num) {
            j++;
            newline += newnum; //adds new new number then ignores copying old num
            while((j<line.length()) && (line.charAt(j) != ' ')) {
                j++;
            }
        }
    }
    pw.println(newline);
}
else {
    pw.println(line);
}
lineCount++;
}
br.close();
pw.close();
file.delete();
temp.renameTo(file);
}

static int listNumbers(PrintWriter tout, BufferedReader tin, int lineNum) throws IOException {
    String fileName = "phonebook2.txt";
    String line = null;
    int tmp = 1;
    int phoneind = 1;
    FileReader fileReader = new FileReader(fileName); //reads file
    BufferedReader bufferedReader = new BufferedReader(fileReader); //buffered reader so it reads a line at a time
    while((line = bufferedReader.readLine()) != null) { //reads through entire list
        if(tmp == lineNum) {
            String newtext;
            newtext = "" + line.charAt(0);
            for(int j=1; j<line.length(); j++) {
                newtext += line.charAt(j);
                if((line.charAt(j) == ' ' && Character.isDigit(line.charAt(j+1))) {
                    newtext += phoneind + " ";
                    phoneind++;
                }
            }
            tout.println(newtext); //prints the lines to client
        }
        tmp++;
    }
    bufferedReader.close();
    tmp--;
    phoneind--;
    return phoneind; //returns length of list
}

static int listNames(PrintWriter tout, BufferedReader tin) throws IOException {
    String fileName = "phonebook2.txt";
    String line = null;
    int tmp = 1;
    FileReader fileReader = new FileReader(fileName); //reads file
    BufferedReader bufferedReader = new BufferedReader(fileReader); //buffered reader so it reads a line at a time
    while((line = bufferedReader.readLine()) != null) { //reads through entire list
        tout.println(tmp + " " + line); //prints the lines to client
        tmp++;
    }
    bufferedReader.close();
    tmp--;
    return tmp; //returns length of list
}
}

```