**ABCDLV Phonebook Pseudo code.**

**Phone Book Main function**

PhoneBookMain {

main() {

ReadWriteFile\_1 data = new ReadWriteFile\_1();

data.fileReader();

LinkedListNode.add(data);

}

}

**Read/Write data to the .txt document.**

ReadWriteFile\_1(){

fileWriter(String data) {

try

{

writeDetails = new BufferedWriter(new FileWriter("phoneBookInfo.txt", true)); //access the file and write information to it. The "true" allow the writing of now info without overwriting current info

current = LinkedListNode.getHead();

for (int i = 0; i < current.data.length; i++ ){

if (i <= 2) {

writeDetails.write(current.data[i] + ","); //write the first three data parts and seperate then with a comma

}

else

{

writeDetails.write(current.data[i] + "\n"); //ensure data writes to a new line every time after the last data is added

}

}

writeDetails.close(); //close the file after writing to it

}

catch (IOException e)

{

JOptionPane.showMessageDialog(null, "Could not save data"); //message to indicate that the data could not be saved.

}

}

public static void fileReader(){

try

{

BufferedReader readDetails = new BufferedReader(new FileReader("phoneBookInfo.txt"));

String line;

while ((line = readDetails.readLine()) != null) //read from the file as until the end of the document

{

System.out.println(line);

}

readDetails.close(); //close the file after reading it

}

catch (IOException e)

{

JOptionPane.showMessageDialog(null, e);

}

}

**Insert**

START

private void createButtonActionPerformed(java.awt.event.ActionEvent evt) // Action event for the insert button

lastname = get text from lastnameInput

name = get text from nameInput

number = get text from numberInput

address = get text from addressInput

nodeData = array containing lastname, name, number, address// Create an array ofdetails

LinkedListNode.add(nodeData) // Add contact to linked list

ReadWriteFile\_1.fileWriter(nodeData) // Write contact details to file

End

**Search**

START

function SearchContact();

function jButton1ActionPerformed(evt) // Action event for the search button

lastname = get text from textField2

name = get text from textField1

number = get text from textField3

foundContacts = searchContactsInFile(lastname, name, number)

IF foundContacts is not empty then

displayContactsInTable(foundContacts)

show message "Contacts found!" with info icon

ELSE

show message "No contacts found." with warning icon

END IF

END

function searchContactsInFile(lastname, name, number) // Search for contacts in the file

foundContacts = empty list

WHILE (line = read line from file) is not null do

details = split line by comma

IF details has 4 elements then

fileLastname = details[0].trim()

fileName = details[1].trim()

fileNumber = details[2].trim()

matches = true

IF lastname is not empty and fileLastname is not equal to lastname then

matches = false

END IF

IF name is not empty and fileName is not equal to name then

matches = false

END IF

IF number is not empty and fileNumber is not equal to number then

matches = false

END IF

IF matches then

add details to foundContacts

END IF

END IF

END WHILE

return found Contacts

END

function displayContactsInTable(contacts) // Display found contacts in a table

model = get model from jTable1

clear existing rows in model

entryNo = 1

FOR each contact in contacts do

add a new row to model with entryNo, contact details

entryNo = entryNo + 1

END FOR

END

**Update**

START

function updateContact(surName, Name, PhoneNumber,address) // Action event for the search button

IF index >= 0 and index < userDetail.size() then// Check if index is valid

print "Enter new user detail: "

current = head // Initialize current to the head of the contact list

WHILE current is not null do // Traverse the linked list of contacts

IF current.name is equal to name then // Check if the current name matches the given

name

current.name = newName // Update the contact's name and phone number

current.phone = newPhone

return true // Contact updated successfully

END IF

current = current.next // Move to the next contact

END WHILE

return false // Contact not found

ELSE

print "Invalid index. Please run the program again."

END IF

END

**Sorting**

START

function ContactSorting()

current = head // Start from the head of the linked list

WHILE current is not null do

minNode = current // Assume the current node is the minimum

prevMin = null // No previous node for minNode yet

pointer = current // Start checking from the current node

WHILE pointer.next is not null do

IF Alphabetical(pointer.next.data, minNode.data) then

minNode = pointer.next // Update minNode

prevMin = pointer // Update previous minNode

pointer = pointer.next // Move to the next node

IF minNode is not current then // Swap data between current and minNode if

different

temp = current.data

current.data = minNode.data

minNode.data = temp

current = current.next // Move to the next node in the list

END WHILE

END

function Alphabetical(node1, node2)

IF node1 is equal to node2 then

return false // They are the same, no need to sort

ELSE

// Compare the first element (assuming it's a string)

return node1[0] < node2[0] // True if node1 comes before node2 alphabetically

ENDIF

END

**Delete**

START

function deleteContact()

prompt "Enter the number to delete:" // Prompt user for the number to delete

number = get user input

if (number exists in contact list) then // Search for the contact number

remove contact with number // Remove the contact

display "Contact number deleted successfully"// Display success message

update ReadWrite file // Update the ReadWrite file

ELSE

display "Contact not found" // Display not found message

END IF

END