DSA Assignment

The program is a linked list application of a phonebook that allows for the insertion, deletion, searching and updating of contact details. It also allows all the contacts to be displayed. We chose to use a linked list, because linked lists are dynamic and the increase and decrease to fit usage application.

Contributions

SN	Name	Student Number	Role played in the project
1	France Lugambo **(GL)**	221134247	Insert function pseudo code
2	Gabriel Justin	222103191	Search function
3	Abed B Indongo	224021850	Delete function
4	Mathias Indongo	224062840	Update function
5	Redemptus Muyeu Jr	223124249	Main method
6	Stern Nyambe	221107363	ListNode constructor and flowchart

Modules

1. Insert contact: inserts new contact details into the linked list.
2. Search contact: searches for a specific contact in the Linked list.
3. Display all contacts: displays all the names and phone numbers in the linked list.
4. Delete contact: deletes contact details from the linked list.
5. Update contact: updates the contact details that are in the list.

Analysis of search algorithm:

Algoritm used: Linear Search

Time Complexity

Worst-case scenario: If the element is at the end of the list or is not present at all, it requires checking all elements, resulting in a time complexity of (O(n)).

Best-case scenario: if the element is at the beginning of the list, it requires only one comparison, resulting in a time complexity of (O(1)).

Average-case scenario: if the element is somewhere in the middle, leading to an average time complexity of (O(n)).

Functions

1. insert (name, phoneNumber)
2. search (name, phoneNumber)
3. display ()
4. delete (name, number)
5. update (name, newName, phonenumber, newPhoneNumber)

PSEUDOCODE

Start

```
head = null    //Setting empty list
tail = null   //Setting empty list
userInput = null  //initializing variable
DO  //do while loop to repeat the code segment
    Display "Choose among the options below using the option numbers:
        1. Insert contact
        2. Search contact
        3. Display all contacts
        4. Delete contact
        5. Update contact"  //prompt user for which action they want to take
    Get option   //get user input
    name = null
    phoneNumber = 0

    Case of (option) {    //case selection for the user's desired action
        1: Prompt user for name   //prompt user for the name to add to the linked list
           Get name
           Prompt user for phoneNumber    // prompt user for the phone number to add linked list
           Get phoneNumber
           insertLast (name, phoneNumber)  //call function to insert contact details into linked list
       //prompt user whether they want to use a phone number or name to search for contact details
       2: Display "Search by phone number or name, enter 1 for phone number or 2 for name: "
           Get searchOption
           If (searchOption == 1)
               Prompt user for phone number   //prompt the user for the phone number to search for
               Get phoneNumber
               search (name, phonenumber)   //call function to search for contact details
           Else if (searchOption == 2)
               Prompt user for name    //prompt the user for the name to search for
               Get name
               search (name, phonenumber)   //call function to search for contact details
          Else
```

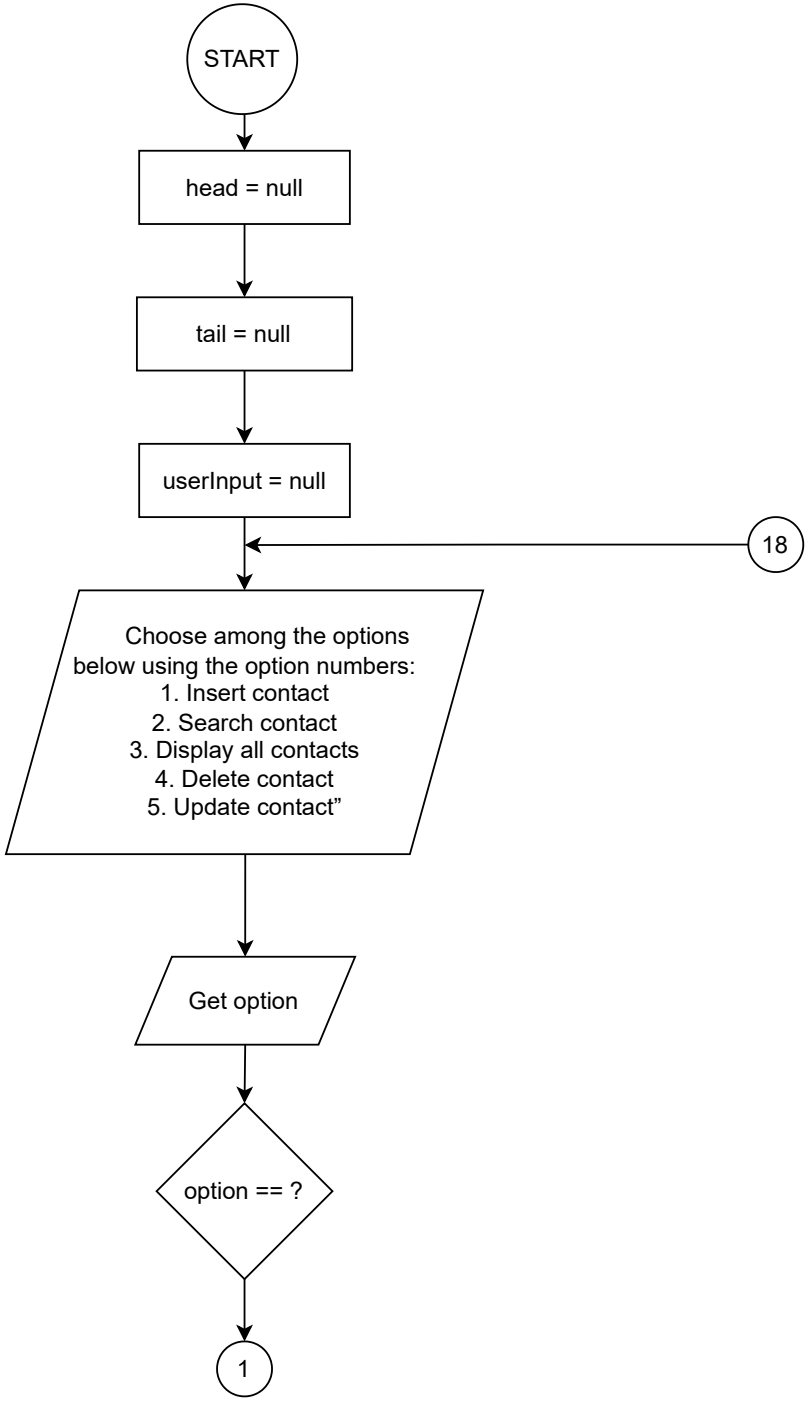
```
            Display “Invalid option”
        Endif
        Endif
    3: display ()   //call function to display all the contact details
    //prompt user whether they want to use a phone number or name to delete contact details
    4: Display “Delete using a phone number or name, enter 1 for phone number or 2 for name:”
        Get deleteOption
        If (deleteOption == 1)
            Prompt user for phone number  //prompt the user for the phone number to delete
            Get phoneNumber
             delete (name, phonenumber)  //call the function to delete contact details
        Else if (deleteOption == 2)
            Prompt user for name   //prompt the user for the name to delete
            Get name
            delete (name, phonenumber)   //call the function to delete contact details
        Else
            Display “Invalid option”
        Endif
        Endif
        //prompt user whether they want to use a phone number or name to update contact details
    5: Display “Update by phone number or name, enter 1 for phone number or 2 for name: ”
        Get updateOption
        If (updateOption == 1)
            Prompt user for phone number   //prompt the user for the phone number to update
            Get phoneNumber
            Display “Enter new phone number, if no changes are to be done enter old phone number”
            Get newPhoneNumber
            Display “Enter new name, if no changes are to be done enter old name”
            Get newName
             update (name, newName, phonenumber, newPhoneNumber)    //call the function to
update contact details
        Else if (updateOption == 2)
            Prompt user for name   //prompt the user for the name to delete
            Get name
```

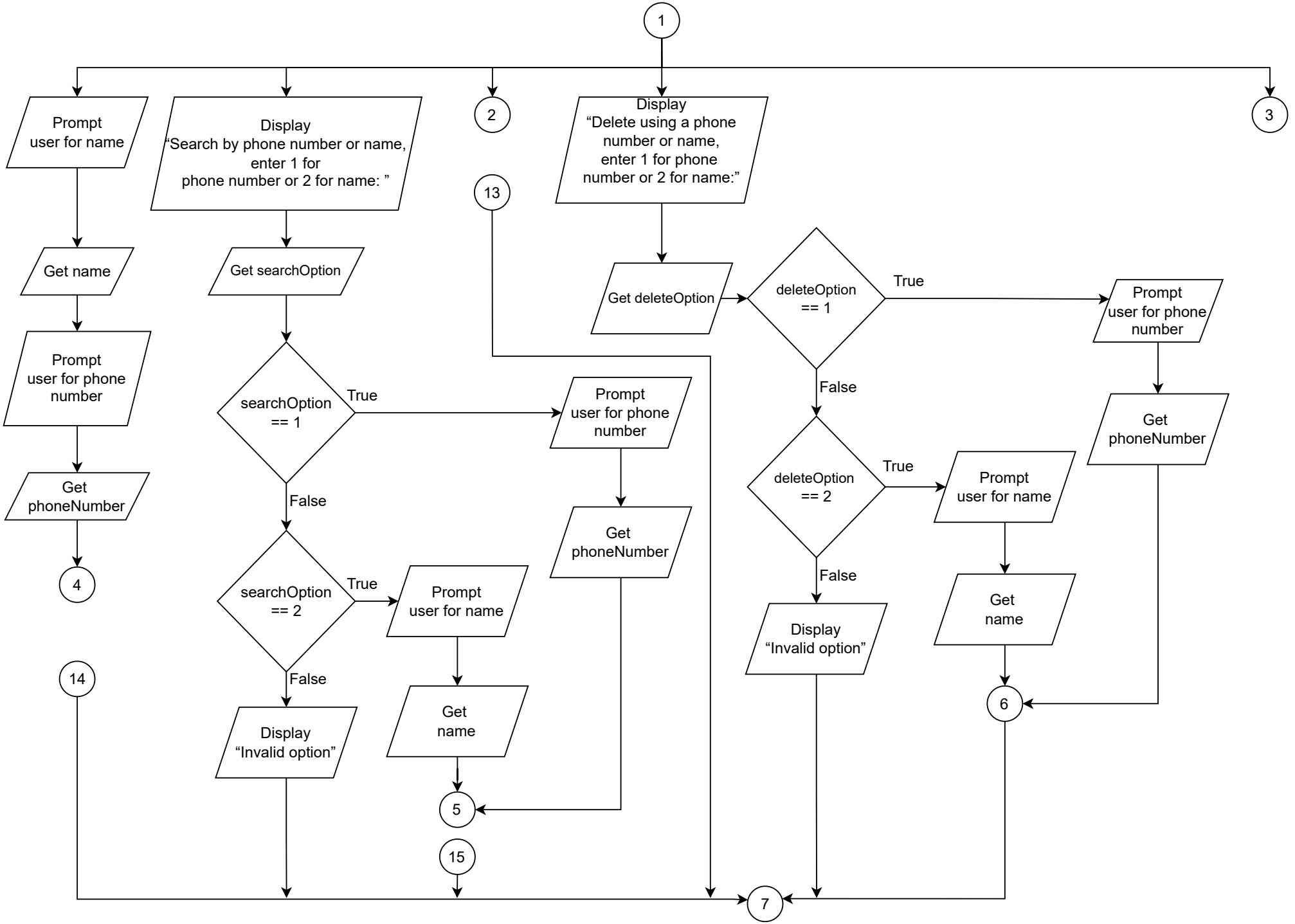
```
                    Display “Enter new phone number, if no changes are to be done enter old phone number”
                    Get newPhoneNumber
                    Display “Enter new name, if no changes are to be done enter old name”
                    Get newName
                    update (name, newName, phonenumber, newPhoneNumber)   //call the function to update contact details
                Else
                    Display “Invalid option”
                Endif
                Endif
            default: Display “Choice not available!”
        endcase
        Display “Do you want to choose another option? (yes/no): ”   //prompt the user if the want to take more actions
        Get userInput
    WHILE (userInput == yes)
End


public void insert (name, phoneNumber) {
    newNode -> name = name   //add data to the new node
    newNode -> phoneNumber = phoneNumber   //add data to the new node
    newNode -> next = null    //make newly created node the last node

    If (head == null)   // check if the list is empty
        head = newNode    //point head pointer to newly created node
    Else
        tail -> next = newNode    //add the new node to the list
    Endif
    tail = newNode   //point rear pointer to newly created node
    Display “Phone number added successfully!”
}


public void search (name, phoneNumber) {
```

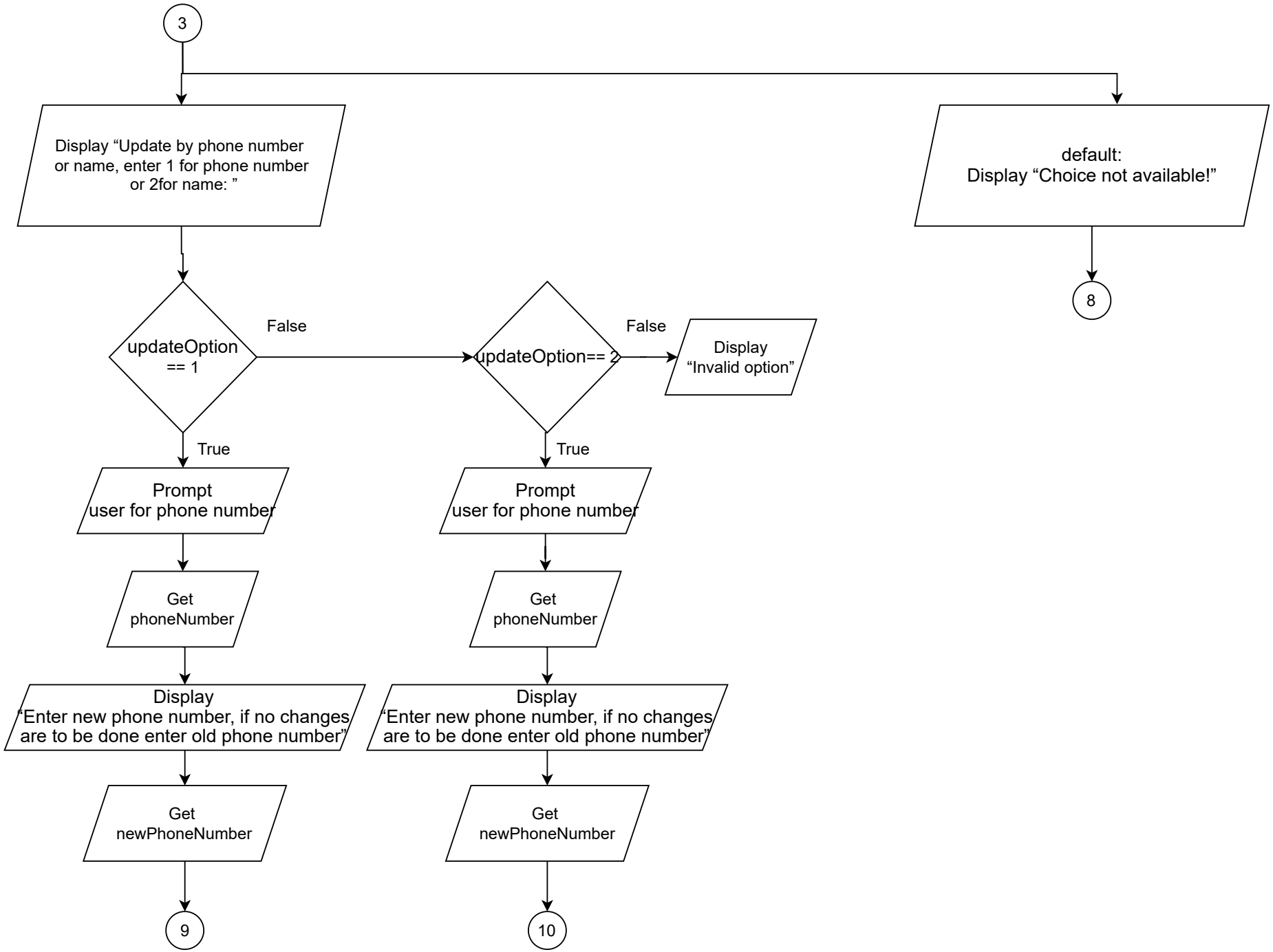
```
    If (head == null)   // check if the list is empty
        Display “List is empty”
        return  //exit the function
    Endif

    temp = head   //initialize temporary pointer to traverse the list
    WHILE (temp != null AND temp -> name != name AND temp -> phoneNumber != phonenumber)  //while loop for list traversal
        temp = temp -> next
    ENDWHILE
    If (temp == null)   //check if the contact has not been found in the list
        Display “Phone number not found!”
    Else
        Display “Phone number found: ” + temp -> name + “ ” + temp -> phoneNumber   //display found contact
    Endif
}

public void display () {
    If (head == null)    // check if the list is empty
        Display “List is empty”
    Else
        temp = head   //initialize temporary pointer to traverse the list
        WHILE (temp != null)   //while loop for list traversal and displaying contacts
            Display temp -> name + temp -> phoneNumber
            temp = temp -> next
        ENDWHILE
    Endif
}

public void delete (name, number) {
    if(head == null )   // check if the list is empty
        Display “List is empty”
```

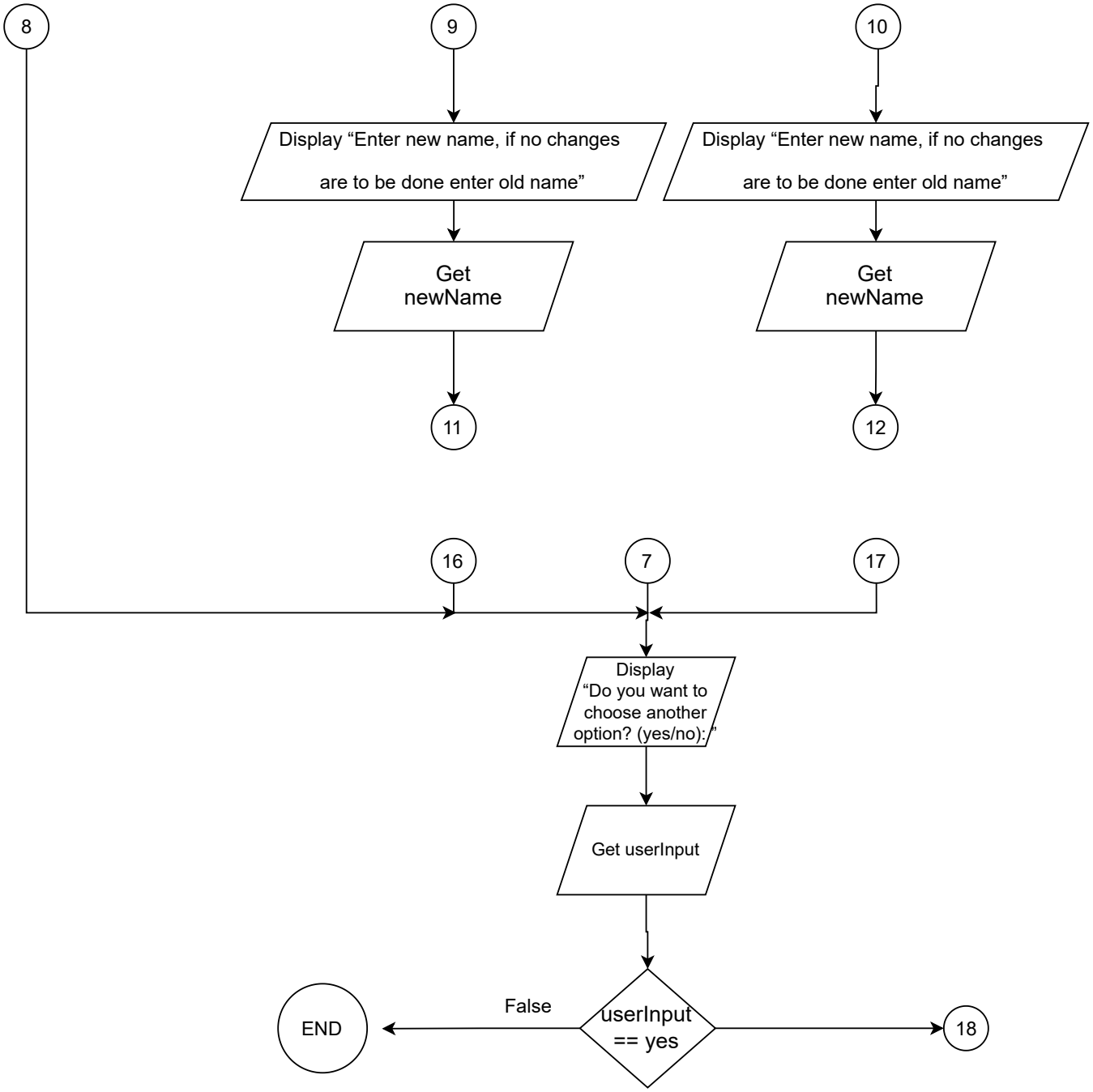
```
        return   //exit the function
    Endif

    temp = head   //initialize temporary pointer to traverse the list
    prev = null   //initialize temporary pointer point at the temp node

    WHILE (temp != null AND temp -> name != name AND temp -> phoneNumber != phonenumber)   //while loop for list traversal
        prev = temp
        temp = temp.next
    ENDWHILE

    If (temp == null)   //check if the contact has not been found in the list
        Display "Phone number not found"
        return //exit the function
    Elseif (temp == head)   //check if the contact is first in the list
        head = head -> next  //change head pointer to point to the second contact in the list
    Else
        prev.next = temp.next   //change the node before the node being deleted to point to the node after the node to be deleted
    Endif
    Endif

    If (temp == tail) //check if the contact is last in the list
        tail = prev   //change tail pointer to point to the second last node
    Endif

    Display temp -> name + temp -> phoneNumber + "deleted!"
}

public void update (name, newName, phonenumber, newPhoneNumber) {
    If (head == null )    // check if the list is empty
        Display "List is empty!"
        return   //exit the function
```

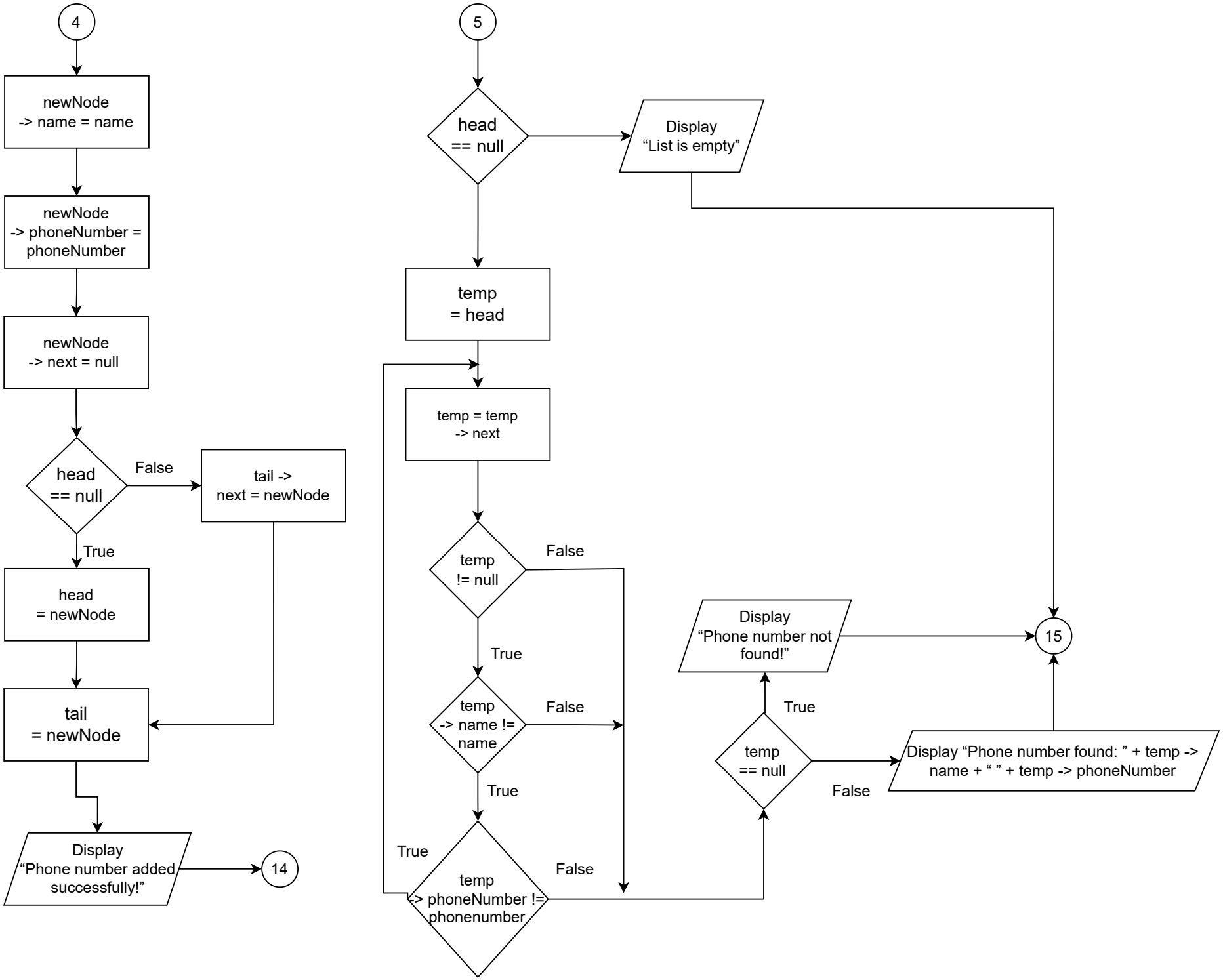
```
    Endif

    Temp = head   //initialize temporary pointer to traverse the list

    WHILE (temp != null AND temp -> name != name AND temp -> phoneNumber != phonenumber)   //while loop for list traversal
        temp = temp.next
    ENDWHILE

    If (temp == null)   //check if the contact has not been found in the list
        Display "Phone number not found"
        return   //exit the function
    Else
        temp -> name = newname  //update name of node
        temp -> phoneNumber = newPhoneNumber   //update phone number of node
    Endif
    Display "Phone number successfully updated"
}
```

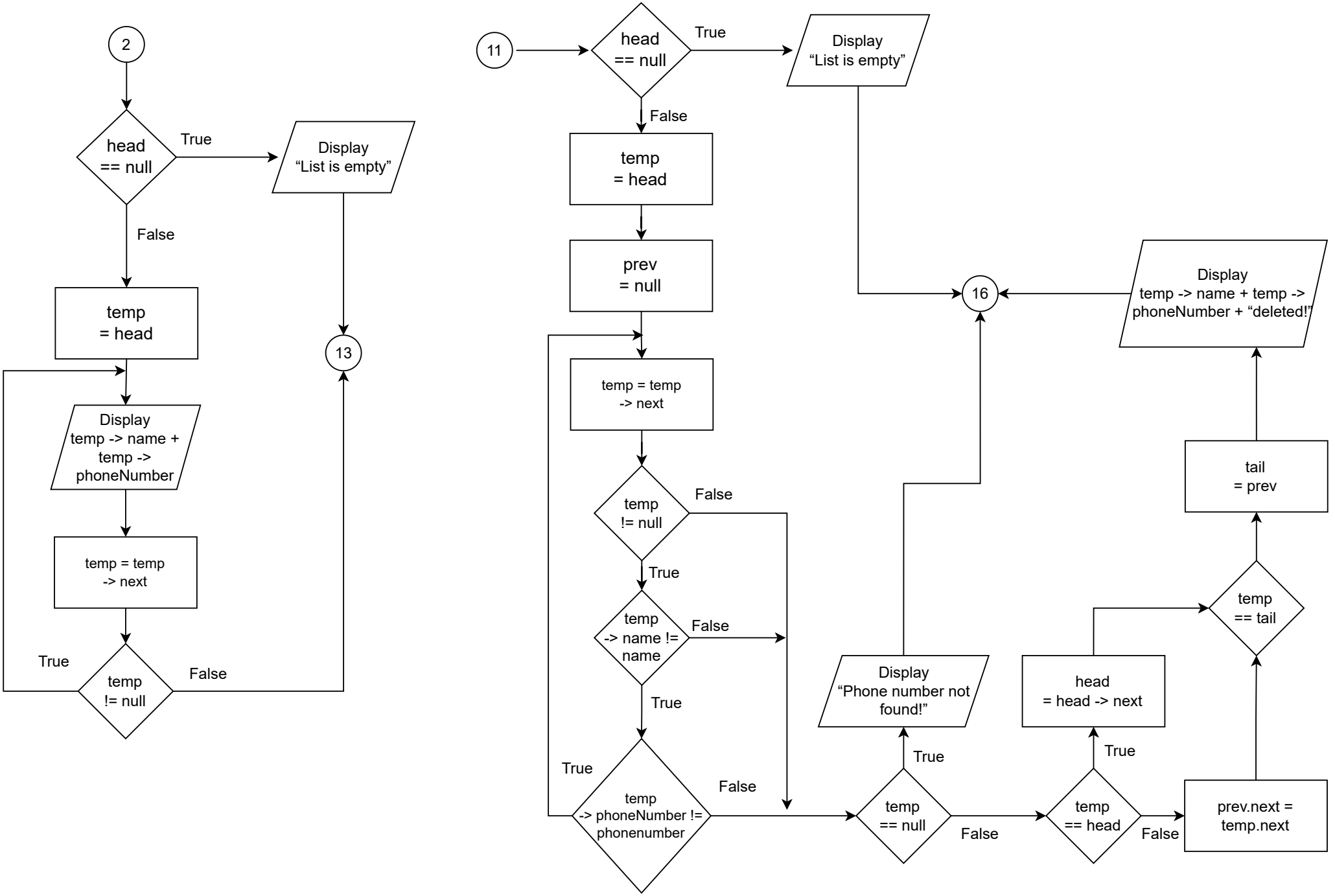

START
head = null
tail = null
userInput = null
18
Choose among the options
below using the option numbers:
1. Insert contact
2. Search contact
3. Display all contacts
4. Delete contact
5. Update contact”
Get option
option == ?
1

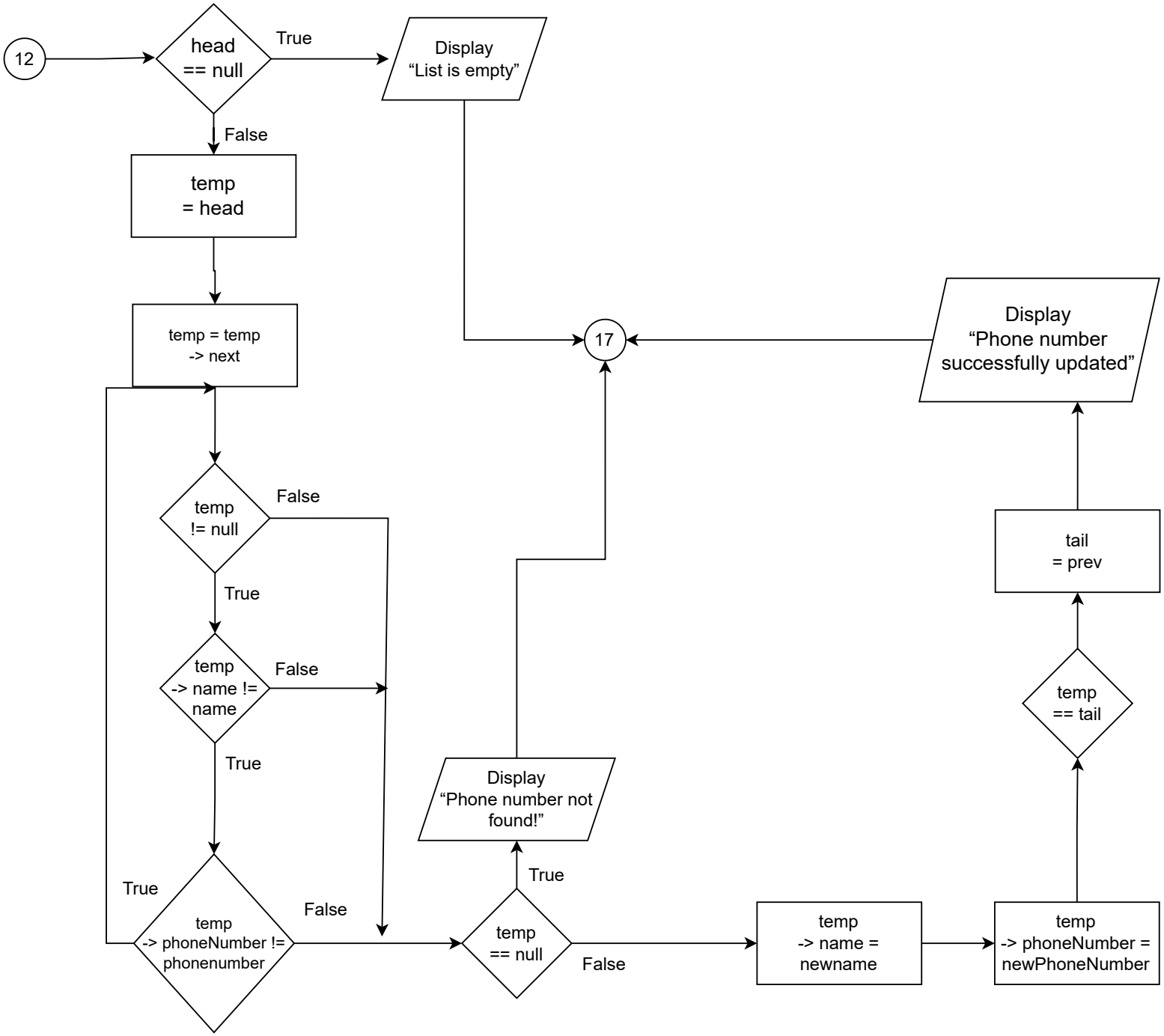

1
Prompt user for name
Display "Search by phone number or name, enter 1 for phone number or 2 for name: "
2
Display "Delete using a phone number or name, enter 1 for phone number or 2 for name:"
3
Get name
Get searchOption
13
Get deleteOption
deleteOption == 1
True
Prompt user for phone number
Prompt user for phone number
searchOption == 1
True
Get phoneNumber
False
deleteOption == 2
True
Prompt user for name
Get phoneNumber
Get phoneNumber
False
searchOption == 2
True
Prompt user for name
Get name
Get name
False
Display "Invalid option"
Display "Invalid option"
4
14
6
5
15
7


3
Display “Update by phone number or name, enter 1 for phone number or 2for name: ”
default:
Display “Choice not available!”
8
updateOption == 1
False
updateOption== 2
False
Display “Invalid option”
True
True
Prompt user for phone number
Prompt user for phone number
Get phoneNumber
Get phoneNumber
Display “Enter new phone number, if no changes are to be done enter old phone number”
Display “Enter new phone number, if no changes are to be done enter old phone number”
Get newPhoneNumber
Get newPhoneNumber
9
10


8
9
10
Display “Enter new name, if no changes are to be done enter old name”
Display “Enter new name, if no changes are to be done enter old name”
Get newName
Get newName
11
12
16
7
17
Display “Do you want to choose another option? (yes/no): ”
Get userInput
END
False
userInput == yes
18


4
newNode
-> name = name
newNode
-> phoneNumber =
phoneNumber
newNode
-> next = null
head
== null
False
tail ->
next = newNode
True
head
= newNode
tail
= newNode
Display
“Phone number added
successfully!”
14
5
head
== null
Display
“List is empty”
temp
= head
temp = temp
-> next
temp
!= null
False
True
temp
-> name !=
name
False
True
temp
-> phoneNumber !=
phonenumber
True
False
Display
“Phone number not
found!”
15
True
temp
== null
False
Display “Phone number found: ” + temp ->
name + “ ” + temp -> phoneNumber


2
head
== null
True
Display
“List is empty”
False
temp
= head
Display
temp -> name +
temp ->
phoneNumber
temp = temp
-> next
True
temp
!= null
False
13
11
head
== null
True
Display
“List is empty”
False
temp
= head
prev
= null
temp = temp
-> next
temp
!= null
False
True
temp
-> name !=
name
False
True
True
temp
-> phoneNumber !=
phonenumber
False
Display
“Phone number not
found!”
True
temp
== null
False
head
= head -> next
True
temp
== head
False
temp
== tail
tail
= prev
prev.next =
temp.next
16
Display
temp -> name + temp ->
phoneNumber + “deleted!”


12
head
== null
True
Display
“List is empty”
False
temp
= head
temp = temp
-> next
17
Display
“Phone number
successfully updated”
temp
!= null
False
True
tail
= prev
temp
-> name !=
name
False
True
temp
== tail
Display
“Phone number not
found!”
True
True
temp
-> phoneNumber !=
phonenumber
False
temp
== null
False
temp
-> name =
newname
temp
-> phoneNumber =
newPhoneNumber