BSc (Hons) in Computing
Level C/I/H

INDIVIDUAL ASSIGNMENT

Module Code & Title:
COSE40574
Application Modelling I

Prepared By: Imesh Ranawaka CB007166

Date of Submission: 4th October 2017

Instructor:

Mr. Nipunu Wijesingha

Word Count:
9255

MARKING CRITERIA	%	MARKS OBTAINED
Register customer		
Unregister customer		
Update customer		
View customer, View all customers		
Apply for a loan		
Approve/Reject loan		
View approved, pending, rejected customers		
Documentation		
Presentation		
TOTAL (%)		

Acknowledgement

We would like to express our assignment to Mr. Nipunu Wijesingha who gave us chance to do our best. She gave us guidance and advised how we can improve our assignment better. In addition, we would like to thank the Asia-Pacific Institution of Information Technology for all the facilities of the computer lab and the library facilities. I also want to thank all our senior's students who gave us additional support and knowledge.

Contents

Acknowledgement ..... **- 2 -**

1. Introduction ..... **- 7 -**

2. ANALYSIS ..... **- 8 -**

1. Functional Requirements ..... - 8 -

1.01 Main Menu ..... - 8 -
1.02 Register Customer ..... - 8 -
1.03 Un-Register Customer ..... - 9 -
1.04 Update Customer ..... - 9 -
1.05 View Customer ..... - 9 -
1.06 View All Customer ..... - 10 -
1.07 Apply for a Loan ..... - 10 -
1.08 Approve / Reject Loan ..... - 10 -
1.09 View Approved Customers ..... - 10 -
1.10 View Pending Customers ..... - 11 -
1.11 View General Customers ..... - 11 -

2. Non-Functional Requirements ..... - 12 -
2.01 Response Time ..... - 12 -
2.02 Security ..... - 12 -
2.03 Usability ..... - 12 -
2.04 Capacity ..... - 12 -

3. Software Requirements ..... - 13 -

4. Hardware Requirements ..... - 13 -

3. Design ..... **- 14 -**

1. Main Menu ..... - 14 -
2. Register Customer ..... - 17 -
3. Un-Register Customer ..... - 20 -
4. Update Customer ..... - 22 -
5. View Customer ..... - 24 -
6. View All Customer ..... - 26 -
7. Apply for a Loan ..... - 26 -
8. Approve / Reject Loan ..... - 29 -
9. View Approved Customers ..... - 31 -
10. View Pending Customers ..... - 31 -
11. View General Customers ..... - 31 -
12. Linear NIC Array Search ..... - 32 -
13. Linear Customer Details Search ..... - 32 -
14. Array Sort ..... - 33 -
15. Display Customer Details ..... - 34 -
16. Loan Amount ..... - 36 -

4. Implementation 37

Data Structure 37
Function Prototype 38
Global Variables 40
Main Menu 41
Display Menu 42
Register Customer 43
Un-Register Customer 45
Update Customer 47
View Customer 48
View All Customers 49
Apply for a Loan 50
Approve / Reject Loan 52
View Approve Customers ..... - 53 -
View Pending Customers ..... - 54 -
View General Customers ..... - 54 -
Display Customer Details ..... - 55 -
Linear Array Search ..... - 56 -
Linear Customer Search ..... - 56 -
Loan Amount ..... - 57 -
NIC Array Sort ..... - 57 -
Clear CMD ..... - 57 -
Insert Customer Details ..... - 58 -
Delete Customer Details ..... - 59 -
Is Empty (Customer) ..... - 60 -
Enter Loan Request Details ..... - 61 -
Delete Loan Request Details ..... - 62 -
Is Empty (Loan) ..... - 62 -

5. Test Log ..... - 63 -

6. Conclusion ..... - 68 -

7. Reference ..... - 69 -

Table of Figures

Design

Figure 1 ..... - 16 -
Figure 2 ..... - 19 -
Figure 3 ..... - 22 -
Figure 4 ..... - 23 -
Figure 5 ..... - 25 -
Figure 6 ..... - 26 -
Figure 7 ..... - 28 -
Figure 8 ..... - 30 -
Figure 9 ..... - 31 -
Figure 10 ..... - 31 -
Figure 11 ..... - 31 -
Figure 12 ..... - 32 -
Figure 13 ..... - 33 -
Figure 14 ..... - 34 -
Figure 15 ..... - 35 -
Figure 16 ..... - 36 -

Implementation

Figure 17 37
Figure 18 37
Figure 19 38
Figure 20 38
Figure 21 39
Figure 22 39
Figure 23 39
Figure 24 40
Figure 25 41
Figure 26 42
Figure 27 43
Figure 28 45
Figure 29 47
Figure 30 48
Figure 31 49
Figure 32 50
Figure 33 52
Figure 34 ..... - 53 -
Figure 35 ..... - 54 -
Figure 36 ..... - 54 -
Figure 37 ..... - 55 -
Figure 38 ..... - 56 -
Figure 39 ..... - 56 -

Figure 40 ..... - 57 -
Figure 41 ..... - 57 -
Figure 42 ..... - 57 -
Figure 43 ..... - 58 -
Figure 44 ..... - 59 -
Figure 45 ..... - 60 -
Figure 46 ..... - 61 -
Figure 47 ..... - 62 -
Figure 48 ..... - 62 -

1. Introduction

The System is about handling 'Bank Of Ceylon' customer loaning system. The program is coded with knowledge of C++ Programing Language. The system uses advanced Data Structures, Link List, Queues and Methods to make the program more efficient. The program handles maximum no of 100 customer records. The program allows customers to Register, Unregister, Update, View Customer details and Apply for a loan. From that loans request list bank officer can approve or reject loan and update the loan status. This program is consistent with 11 main functions in it.

1. Register Customer
2. Un-Register Customer
3. Update Customer
4. View Customer
5. View All Customers
6. Apply for a Loan
7. Approve / Reject Loan
8. View Approved Customers
9. View Pending Customers
10. View General Customers

Arrays, Data Structures and Pointers used for the above-mentioned main functions. All Customer Details stored in a Data Structure using Pointers in Memory. Customer NIC is the primary key and it cannot be duplicated again. Also, Customer NIC can be in a range of 5-digit Number. In this Customer Details stored as a Data Structures of Link List and Customer Loan Request Details stored as a Data Structures of Queue in the System. Queue represents "FIFO" Concept.

FIFO – First In First Out

2. ANALYSIS

1. Functional Requirements

1.01 Main Menu

The Bank of Ceylon – Loaning System will start displaying the main menu with the Bank Name and the Welcome message. Then it will display the main 11 functions with numbers, so the user can choose an option using that given number. Options will display as follows:

1. Register Customer
2. Un-Register Customer
3. Update Customer
4. View Customer
5. View All Customers
6. Apply for a Loan
7. Approve / Reject Loan
8. View Approved Customers
9. View Pending Customers
10. View General Customers

If the user enters options that not in the main menu program will prompt a message "Selected Option is Invalid!".

1.02 Register Customer

"Register Customer" will allow the bank officer to register a customer into the System. In this function first, checks whether entered NIC valid by checking the number of digits. This will prevent entering invalid data to the System. Also, it checks entered customer already registered in the System. Only if customer not registered in the System bank officer can register Customer details into the System.

Next function will request other Customer details and proceed to the storing process. Before that System will request to confirm if entered Customer details are correct or not. If bank officer misspelled anything he/she can confirm it.

In this Customer details stored in Data structure of Link list. When storing customer details, it will store in the sorted list of NICs. For storing process function will go through a separate process with sorting system. When successfully stored System will display a message "Registered Successfully!".

1.03 Un-Register Customer

"Un-Register Customer" will allow bank office to un-register a customer from the System. In this function first, checks whether entered NIC valid by checking the number of digits. This will prevent entering invalid data to the System. Also, it checks entered customer already registered in the System. Only if customer registered in the System bank officer can un-register the Customer from the system.

Next function will go through a separate process to do the un-registering. In this process customer, NIC will search through the link list and remove that customer from the System. When customer un-registered Successfully System will display a message "This Customer is Not Registered!".

In this, if customer obtained a Loan or requested a loan System will not allow to un-register the Customer from the System. Any of these conditions System will display a proper error message and notify the Bank officer.

1.04 Update Customer

"Update Customer" will allow the bank officer to update a customer in the System. In this function first, checks whether entered NIC valid by checking the number of digits. This will prevent entering invalid data to the System. Also, it checks entered customer already registered in the System. Only if customer registered in the System bank officer can update the customer in the System.

The system will allow only to update customer address. To update customer address system will go through a process to find the customer details and update the correct customer address with the new address. When customer updated successfully system will display a message "Customer Details Updated Successfully!".

1.05 View Customer

"View Customer" will allow the bank officer to view customer details in the System. In this function first, checks whether entered NIC valid by checking the number of digits. This will prevent entering invalid data to the System. Also, it checks entered customer already registered in the System. Only if customer registered in the System bank officer can view the customer details in the System.

In this function customer details (NIC, Name, Address, Age, Gender, and Type) will display. If customer obtained a Loan then loan details will display in the System.

1.06 View All Customer

"View All Customer" will allow the bank officer to view all the Customers in the System. In this function, it will forward to a separate function to display all the Customers following details.

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

1.07 Apply for a Loan

"Apply for a Loan" will allow the bank officer to place a loan request into the System by giving Customer Loan details. In this function first, checks whether entered NIC valid by checking the number of digits. This will prevent entering invalid data to the System. Also, it checks entered customer already registered in the System. Only if customer registered in the System bank officer can place a loan request in the System.

In this function, if that entered customer already obtained a loan or requested a loan system will not allow placing a loan request and back officer will notify with an error message. To apply for a Loan bank officer, need to enter loan details (Loan Type, Amount, Duration). Then this loan request will be stored in a Queue. For this storing process, there's a separate function used to perform the task. This new loan request will be placed the end of Queue. When loan successfully placed System will display a message "Loan Request Successfully Placed!" and main customer details customer type will change to a Pending customer.

1.08 Approve / Reject Loan

"Approve / Reject Loan" will allow the bank officer to accept or reject requested loans. In this function, the system will go through the Queue and get each loan request and display that loan request details. So, the bank officer checks the loan details and decide to approve or reject the loan request.

When loan approved or rejected System will remove that loan request from the queue. If that request is an approved one then that loan request details will be stored with the main customer details. Also, main customer details customer type will change to Loan customer and System will display a proper message saying loan request approved or rejected.

1.09 View Approved Customers

"View All Customer" will allow the bank officer to view all the Customers in the System. In this function, it will forward to a separate function to display all the Approved Customers following details.

1. Customer NIC
2. Customer Name
3. Loan Amount

Customer Details will display in a sorted order of NICs.

1.10 View Pending Customers

"View All Customer" will allow the bank officer to view all the Customers in the System. In this function, it will forward to a separate function to display all the Pending Customers following details.

1. Customer NIC
2. Customer Name
3. Loan Amount

Customer Details will display in a sorted order of NICs.

1.11 View General Customers

"View All Customer" will allow the bank officer to view all the Customers in the System. In this function, it will forward to a separate function to display all the General Customers following details.

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

2. Non-Functional Requirements

2.01 Response Time

DEFINITION: the length of time taken for a person or system to react to a given stimulus or event.

Users always looking fast response from the System. Customer Register, Un-Register, Update and for other processors, the user needs fast access.

2.02 Security

DEFINITION: the state of being free from danger or threat

The user can mistype some entries to the System. So, the System needs to prevent those threats from the user.

2.03 Usability

DEFINITION: the degree to which something is able or fit to be used.

The system needs to user-friendly for the user. The user will be able to use the System with even the little bit of computer knowledge.

2.04 Capacity

DEFINITION: the maximum amount that something can contain.

The system will be able to store register many customers to System. Each customer storing details needs to manage in proper data types.

3. Software Requirements

Operating System: Windows XP, Windows 7 or higher

CMD (Command Prompt)

Microsoft Visual C++ 2017 Redistributable

4. Hardware Requirements

CPU: Pentium® Dual-Core CPU E5300 @ 2.60GHz

Memory: 2 GB RAM

Hard Disk capacity: 256 MB

VGA: (Not needed to run the program)

3. Design

1. Main Menu

```
BEGIN
    Initialize option to ZERO
    Initialize x to ZERO
    Initialize noOfCust to 100
    Initialize nicMAX to 100000
    FOR x < noOfCust Then
        ::custNIC[x] = nicMAX
        x increment by 1
    ENDFOR

    REPEAT
        Function displayMenu()
        Output "Select an Option :"
        Input option
        CASE option Then
            Case 1:
                Output "Selected Option 1) Register Customer"
                Function registerCustomer()
                Break
            Case 2:
                Output "Selected Option 2) Un-Register Customer"
                Function unregisterCustomer()
                Break
            Case 3:
                Output "Selected Option 3) Update Customer"
                Function updateCustomer()
                Break
            Case 4:
                Output "Selected Option 4) View Customer"
                Function viewCustomer()
                Break
            Case 5:
                Output "Selected Option 5) View All Customers"
                Function viewAllCustomer()
                Break
            Case 6:
                Output "Selected Option 6) Apply for a Loan"
                Function applyLoan()
                Break
            Case 7:
                Output "Selected Option 7) Approve/Reject Loan"
                Function appRejLoan()
                Break
            Case 8:
                Output "Selected Option 8) View Approved Customers"
                Function viewAppCustomer()
                Break
            Case 9:
                Output "Selected Option 9) View Pending Customers"
                Function viewPendingCustomer()
```

```
                Break
            Case 10:
                Output "Selected Option 10) View General Customers"
                Function viewGeneralCustomer()
                Break
            Default:
                Output "Selected Option is Invalid!"
        ENDCASE
    UNTIL option!=0
END
```

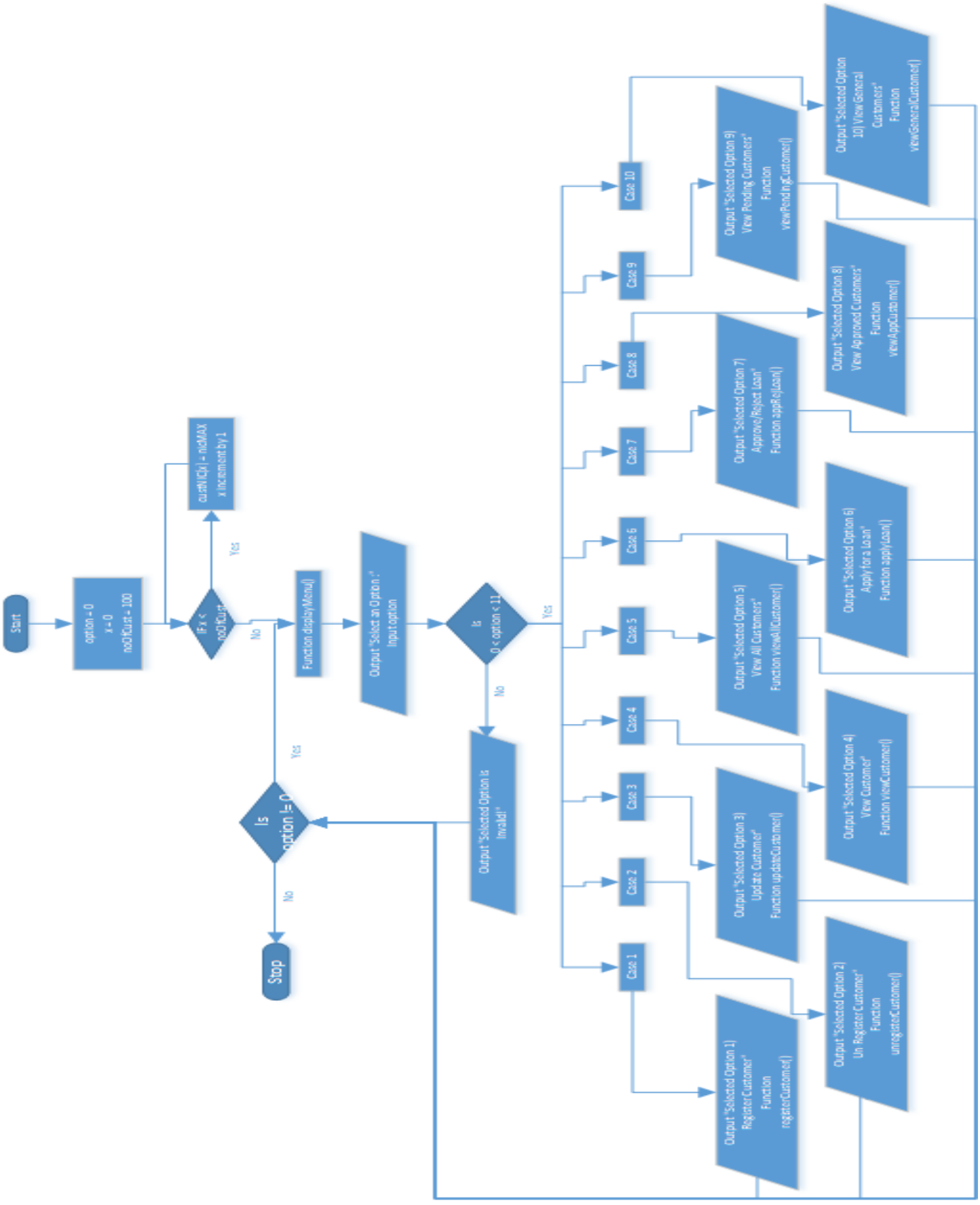
Start
option = 0
x = 0
noOfCust = 100
If x < noOfCust
custN[x] = noMAX
x increment by 1
Yes
No
Function displayMenu()
Output "Select an Option :"
Input option
Is 0 < option < 11
Yes
No
Output "Selected Option is Invalid!"
Is option != 0
Yes
No
Stop
Case 1
Case 2
Case 3
Case 4
Case 5
Case 6
Case 7
Case 8
Case 9
Case 10
Output "Selected Option 1) Register Customer" Function registerCustomer()
Output "Selected Option 2) Un Register Customer" Function unregisterCustomer()
Output "Selected Option 3) Update Customer" Function updateCustomer()
Output "Selected Option 4) View Customer" Function viewCustomer()
Output "Selected Option 5) View All Customers" Function viewAllCustomer()
Output "Selected Option 6) Apply for a Loan" Function applyLoan()
Output "Selected Option 7) Approve/Reject Loan" Function appRejLoan()
Output "Selected Option 8) View Approved Customers" Function viewAppCustomer()
Output "Selected Option 9) View Pending Customers" Function viewPendingCustomer()
Output "Selected Option 10) View General Customers" Function viewGeneralCustomer()

Figure 1

2. Register Customer

```
BEGIN
        Initialize nic to ZERO
        Initialize cName[20] to ""
        Initialize address[50] to ""
        Initialize age to ZERO
        Initialize gender to NULL
        Initialize save to NULL
        Initialize flag to true
        Initialize element to ZERO
        Initialize nicMin to 10000
        Initialize nicMax to 100000

        Output "Enter the NIC Number : "
        Input nic

        IF nic<nicMin OR nic>= nicMax Then
                Function clearCMD()
                Output "Entered NIC is Invalid!"
        ELSE
                element = linearArrSearch(::custNIC, nic)
                IF element!=-1 Then
                        Function clearCMD()
                        Output "Customer already Registered."
                ELSE
                        Output "Enter Customer Name : "
                        Input cName

                        Output "Enter Customer Address : "
                        Input address

                        Output "Enter Customer Age : "
                        Input age

                        Output "Enter Customer Gender (M :- Male / F :- Female): "
                        Input gender

                        IF (gender != 'M') AND (gender != 'm') AND (gender != 'f') AND (gender != 'F') Then
                                Function clearCMD()
                                Output "Selected Gender is Invalid!"
                                Output "Registration Unsuccessful."
                        ELSE
                                Output "Customer Name : " + cName
                                Output "Customer Address : " + address
                                Output "Customer Age : " + age

                                IF (gender == 'M') OR (gender == 'm') Then
                                        Output "Customer Gender : Male"
                                ELSE
                                        Output "Customer Gender : Female"
                                ENDIF

                                REPEAT
```

```
                                        Output "Do you want to save these details (Y/N) : "
                                        Input save
                                        IF (save == 'Y') OR (save == 'y') Then
                                                Function clearCMD()

                                                flag = insertCustDetails(&cust, nic, cName, address, age, gender, 'G')
                                                Function Sort(::custNIC)
                                                IF flag Then
                                                        Output "Registered Successfully!"
                                                ELSE
                                                        Output "Registration Unsuccessful."
                                                ENDIF
                                        ELSEIF (save == 'N') OR (save == 'n') Then

                                                Function clearCMD()
                                                Output "Registration Unsuccessful."
                                        ELSE
                                                Output "Invalid Entry!"
                                        ENDIF

                                UNTIL (save != 'N') AND (save != 'n') AND (save != 'Y') AND (save != 'y')
                        ENDIF
                ENDIF
        ENDIF
END
```

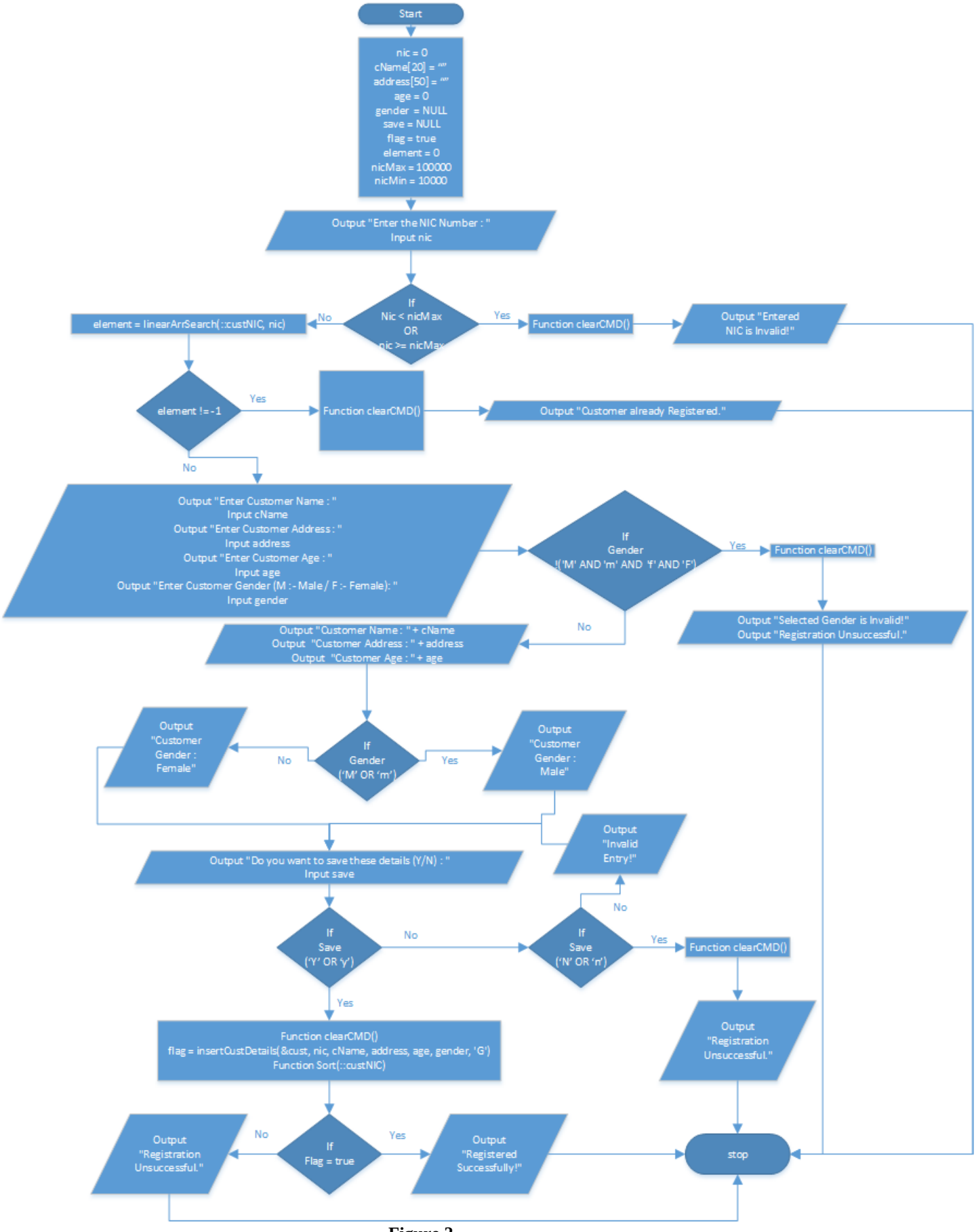
Start
nic = 0
cName[20] = ""
address[50] = ""
age = 0
gender = NULL
save = NULL
flag = true
element = 0
nicMax = 100000
nicMin = 10000
Output "Enter the NIC Number : "
Input nic
If
Nic < nicMax
OR
nic >= nicMax
Yes
Function clearCMD()
Output "Entered NIC is Invalid!"
No
element = linearArrSearch(::custNIC, nic)
element != -1
Yes
Function clearCMD()
Output "Customer already Registered."
No
Output "Enter Customer Name : "
Input cName
Output "Enter Customer Address : "
Input address
Output "Enter Customer Age : "
Input age
Output "Enter Customer Gender (M :- Male / F :- Female): "
Input gender
If
Gender
!('M' AND 'm' AND 'f' AND 'F')
Yes
Function clearCMD()
Output "Selected Gender is Invalid!"
Output "Registration Unsuccessful."
No
Output "Customer Name : " + cName
Output "Customer Address : " + address
Output "Customer Age : " + age
If
Gender
('M' OR 'm')
No
Output "Customer Gender : Female"
Yes
Output "Customer Gender : Male"
Output "Invalid Entry!"
Output "Do you want to save these details (Y/N) : "
Input save
If
Save
('Y' OR 'y')
No
If
Save
('N' OR 'n')
Yes
Function clearCMD()
No
Yes
Function clearCMD()
flag = insertCustDetails(&cust, nic, cName, address, age, gender, 'G')
Function Sort(::custNIC)
Output "Registration Unsuccessful."
If
Flag = true
No
Output "Registration Unsuccessful."
Yes
Output "Registered Successfully!"
stop

Figure 2

3. Un-Register Customer

```
BEGIN
        Initialize nic to ZERO
        Initialize cName[20] to ""
        Initialize address[50] to ""
        Initialize age to ZERO
        Initialize gender to NULL
        Initialize save to NULL
        Initialize flag to true
        Initialize element to ZERO
        Initialize nicMin to 10000
        Initialize nicMax to 100000

        Output "Enter the NIC Number : "
        Input nic

        IF nic<nicMin OR nic>= nicMax Then
                Function clearCMD()
                Output "Entered NIC is Invalid!"
        ELSE
                element = linearArrSearch(::custNIC, nic)
                IF element!=-1 Then
                        Function clearCMD()
                        Output "Customer already Registered."
                ELSE
                        Output "Enter Customer Name : "
                        Input cName

                        Output "Enter Customer Address : "
                        Input address

                        Output "Enter Customer Age : "
                        Input age

                        Output "Enter Customer Gender (M :- Male / F :- Female): "
                        Input gender

                        IF (gender != 'M') AND (gender != 'm') AND (gender != 'f') AND (gender != 'F') Then
                                Function clearCMD()
                                Output "Selected Gender is Invalid!"
                                Output "Registration Unsuccessful."
                        ELSE
                                Output "Customer Name : " + cName
                                Output "Customer Address : " + address
                                Output "Customer Age : " + age

                                IF (gender == 'M') OR (gender == 'm') Then
                                        Output "Customer Gender : Male"
                                ELSE
                                        Output "Customer Gender : Female"
                                ENDIF

                                REPEAT
```

```
                                        Output "Do you want to save these details (Y/N) : "
                                        Input save
                                        IF (save == 'Y') OR (save == 'y') Then
                                                Function clearCMD()
                                                flag = insertCustDetails(&cust, nic, cName, address, age, gender, 'G')
                                                Function Sort(::custNIC)
                                                IF flag Then
                                                        Output "Registered Successfully!"
                                                ELSE
                                                        Output "Registration Unsuccessful."
                                                ENDIF
                                        ELSEIF (save == 'N') OR (save == 'n') Then
                                                Function clearCMD()
                                                Output "Registration Unsuccessful."
                                        ELSE
                                                Output "Invalid Entry!"
                                        ENDIF

                                UNTIL (save != 'N') AND (save != 'n') AND (save != 'Y') AND (save != 'y')
                        ENDIF
                ENDIF
        ENDIF
END
```

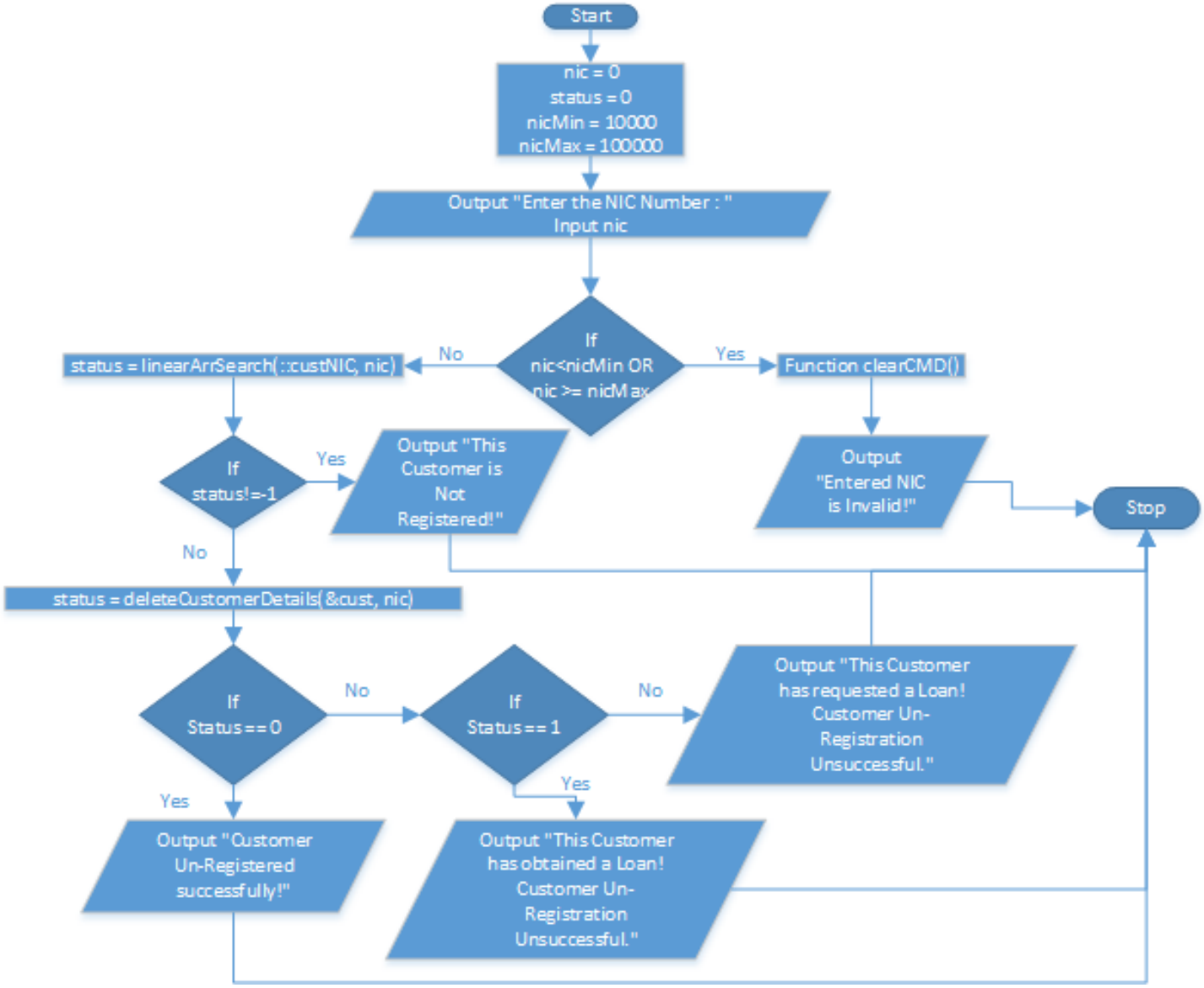

Figure 3

4. Update Customer

```
BEGIN
        Initialize nic to ZERO
        Initialize address[40] to NULL
        Initialize element to ZERO
        Initialize currentPtr to NULL
        Initialize nicMin to 10000
        Initialize nicMax to 100000

        Output "Enter the NIC Number : "
        Input nic
        IF nic<nicMin OR nic >= nicMax Then
                Function clearCMD()
                Output "Entered NIC is Invalid!"
```

```
        ELSE
                element = linearArrSearch(::custNIC, nic)
                Function clearCMD()
                IF element != -1 Then
                        Output "Enter New Address : "
                        Input address
                        currentPtr = linearCustSearch(cust, nic)
                        strcpy_s(currentPtr->customerAddress, address)
                        Output "Customer Details Updated Sucessfully!"
                ELSE
                        Output "This Customer is Not Registered!"
                ENDIF
        ENDIF
END
```

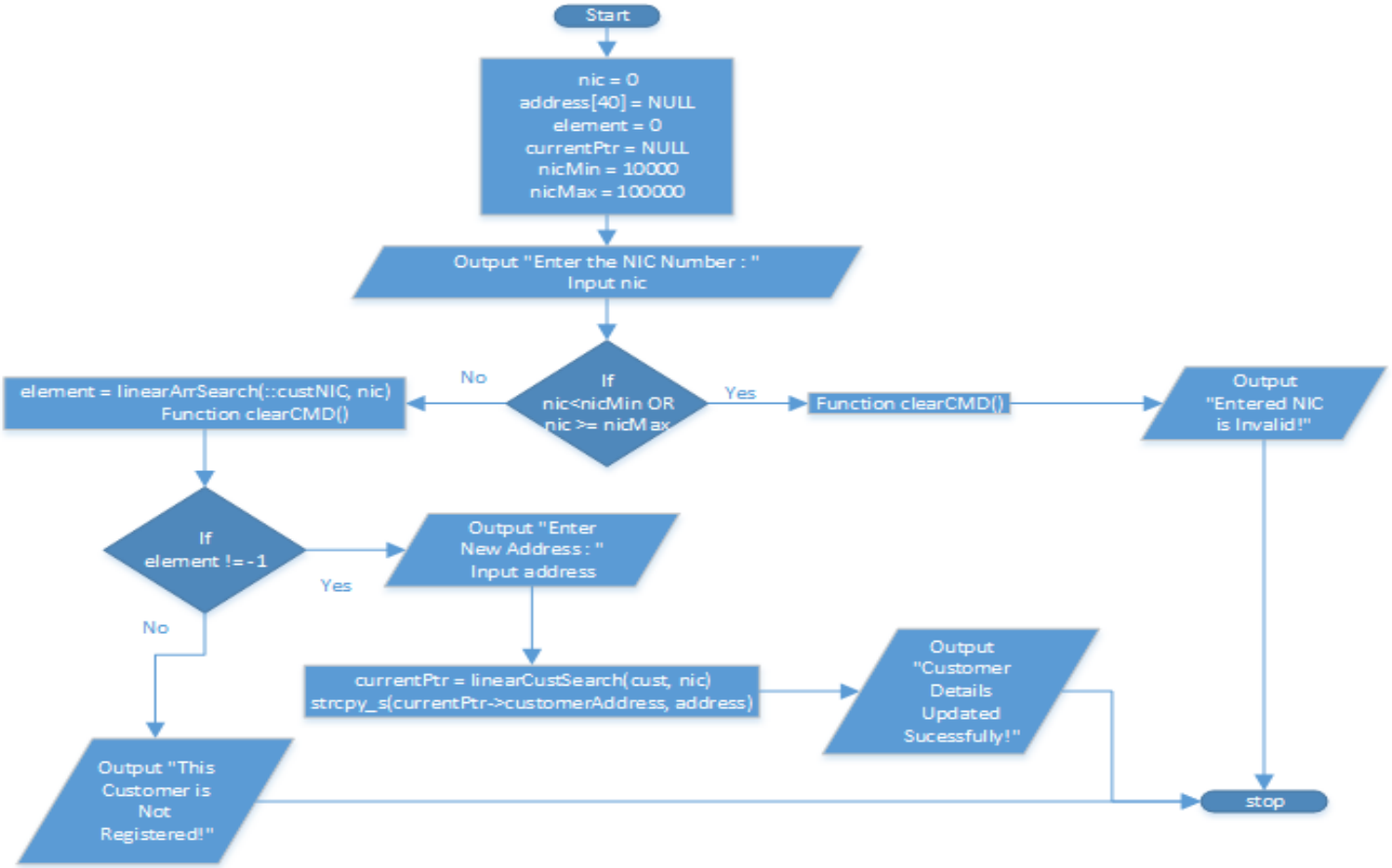

Figure 4

5. View Customer

```
BEGIN
    Initialize nic to ZERO
    Initialize gender to NULL
    Initialize type to NULL
    Initialize lType to NULL
    Initialize element to ZERO
    Initialize currentPtr to NULL
    Initialize nicMin to 10000
    Initialize nicMax to 100000
    Output "Enter the NIC Number : "
    Input nic
    IF nic<nicMin OR nic >= nicMax Then
        Function clearCMD()
        Output "Entered NIC is Invalid!"
    ELSE
        element = linearArrSearch(::custNIC, nic)
        Function clearCMD()
        IF element != -1 Then
            currentPtr = linearCustSearch(cust, nic)
            Output "Customer Name : " + currentPtr->customerName
            Output "Customer Address : " + currentPtr->customerAddress
            Output "Customer Age : " + currentPtr->customerAge

            gender = currentPtr->customerGender
            IF gender=='M' OR gender=='m' Then
                Output "Customer Gender : Male"
            ELSE
                Output "Customer Gender : Female"
            ENDIF
            type = currentPtr->customerType
            IF type=='G' Then
                Output "Customer Type : General Customer"
            ELSEIF type=='P' Then
                Output "Customer Type : Pending Customer"
            ELSE
                Output "Customer Type : Loan Customer"
                lType = currentPtr->loanType
                IF lType == 'H' OR lType == 'h' Then
                    Output "Loan Type : Home Loan"
                ELSEIF lType == 'P' OR lType == 'p' Then
                    Output "Loan Type : Personal Loan"
                ELSEIF lType == 'G' OR lType == 'G' Then
                    Output "Loan Type : Gold Loan"
                ELSE
                    Output "Loan Type : Leasing"
                ENDIF
                Output "Loan Amount : " + currentPtr->loanAmount
                Output "Loan Duration : " + currentPtr->loanDuration
            ENDIF
        ELSE
            Output "This Customer is Not Registered!"
        ENDIF
```

ENDIF

END

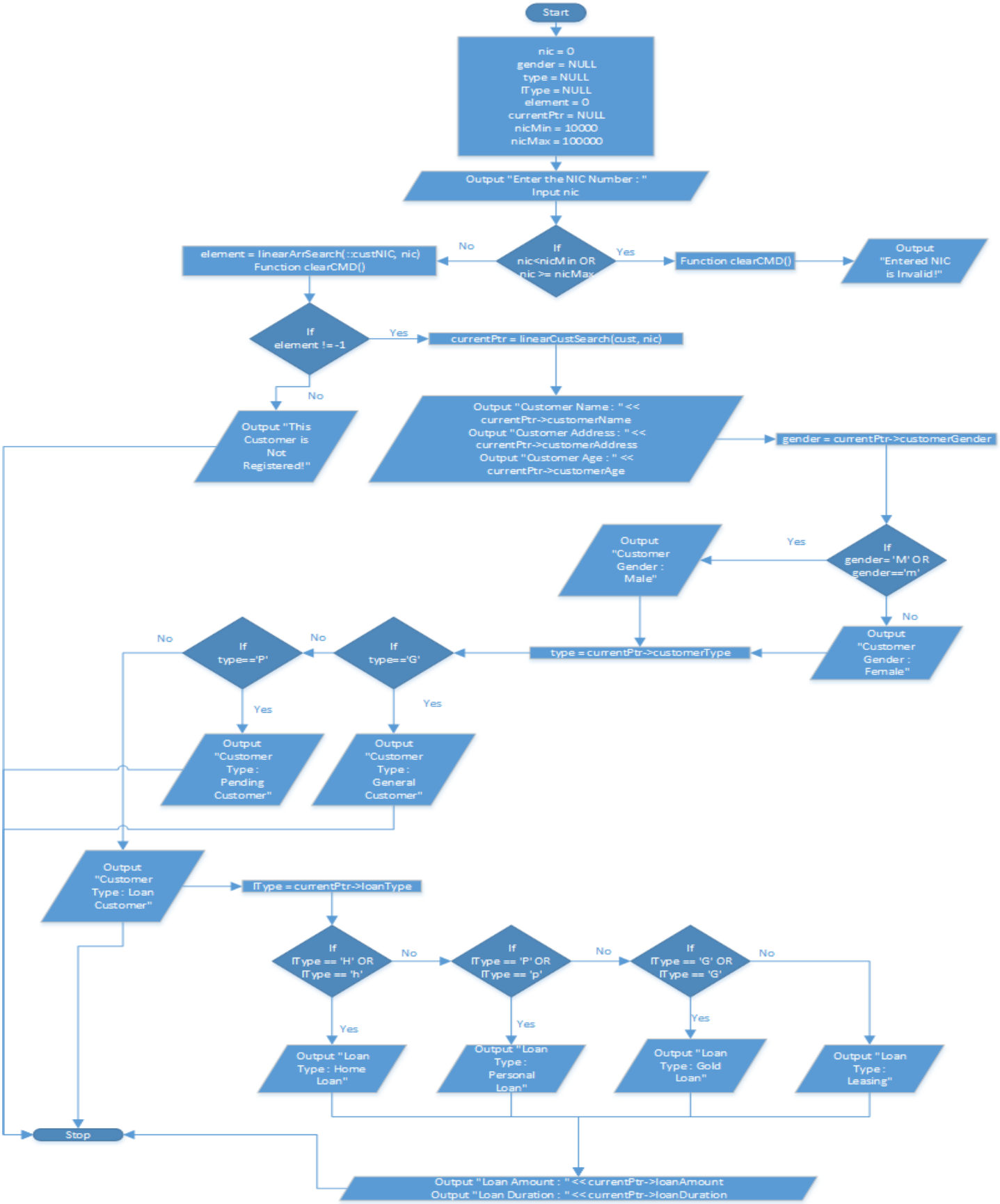

Figure 5

6. View All Customer

```
BEGIN
        Function displayCustomerDetails('A')
END
```

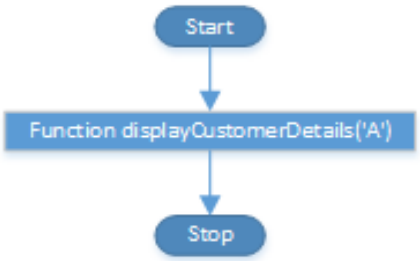

Figure 6

7. Apply for a Loan

```
BEGIN
        Initialize nic to ZERO
        Initialize cType to NULL
        Initialize lType to NULL
        Initialize lAmount to ZERO
        Initialize lDuration to ZERO
        Initialize save to NULL
        Initialize element to ZERO
        Initialize currentPtr to NULL
        Initialize nicMin to 10000
        Initialize nicMax to 100000

        Output "Enter the NIC Number : "
        Input nic
        IF nic<nicMin OR nic >= nicMax Then
                Function clearCMD()
                Output "Entered NIC is Invalid!"
        ELSE
                element = linearArrSearch(::custNIC, nic)
                IF element!=-1 Then
                        currentPtr = linearCustSearch(cust, nic)

                        cType = currentPtr->customerType

                        IF cType=='G' Then
                                Output "Enter Loan Type (H:- Home Loans / P:- Personal Loans / G:- Gold Loans / L:-
                                                                                                Leasing) : "
                                Input lType
                                IF (lType!='H') AND (lType != 'P') AND (lType != 'G') AND
                                (lType != 'L') AND (lType != 'h') AND (lType != 'p') AND
                                (lType != 'g') AND (lType != 'l') Then
                                        Function clearCMD()
                                        Output "Selected Loan Type is Invalid! Loan Request Unsuccessful!"
                                ELSE
                                        Output "Enter Loan Amount : "
```

```
                                Input lAmount
                                Output "Enter Loan Duration : "
                                Input lDuration

                                IF lType=='H' OR lType=='h' Then
                                        Output "Loan Type : Home Loan"
                                ELSEIF lType == 'P' OR lType == 'p' Then
                                        Output "Loan Type : Personal Loan"
                                ELSEIF lType == 'G' OR lType == 'G' Then
                                        Output "Loan Type : Gold Loan"
                                ELSE
                                        Output "Loan Type : Leasing"
                                ENDIF

                                Output "Loan Amount : " + lAmount
                                Output "Loan Duration : " + lDuration

                                REPEAT
                                    Output "Do you want to save these details (Y/N) : "
                                      Input save
                                      IF save == 'Y' OR save == 'y' Then
                                              Function enterLoanDetails(&headLoanPtr, &tailLoanPtr, nic,
                                              lType, lAmount, lDuration)
                                              currentPtr->customerType = 'P'
                                              Function clearCMD()
                                              Output "Loan Request Successfully
                                                                      Placed!"
                                      ELSEIF save == 'N' OR save == 'n' Then
                                              Function clearCMD()
                                              Output "Loan Request Unsuccessful!"
                                      ELSE
                                              Output "Invalid Entry!"
                                      ENDIF
                                UNTIL save != 'Y' AND save != 'y' AND save != 'N'
                                                                AND save != 'n'
                        ENDIF
                ELSEIF cType=='L' Then
                        Function clearCMD()
                        Output "Already there's a Obtained Loan under this Customer.
                                Loan Request Unsuccessful!"
                ELSE
                        Function clearCMD()
                        Output "Already there's a Pending Loan under this Customer.
                                Loan Request Unsuccessful!"
                ENDIF
        ELSE
                Function clearCMD()
                Output "This Customer is Not Registered!"
        ENDIF
    ENDIF
END
```

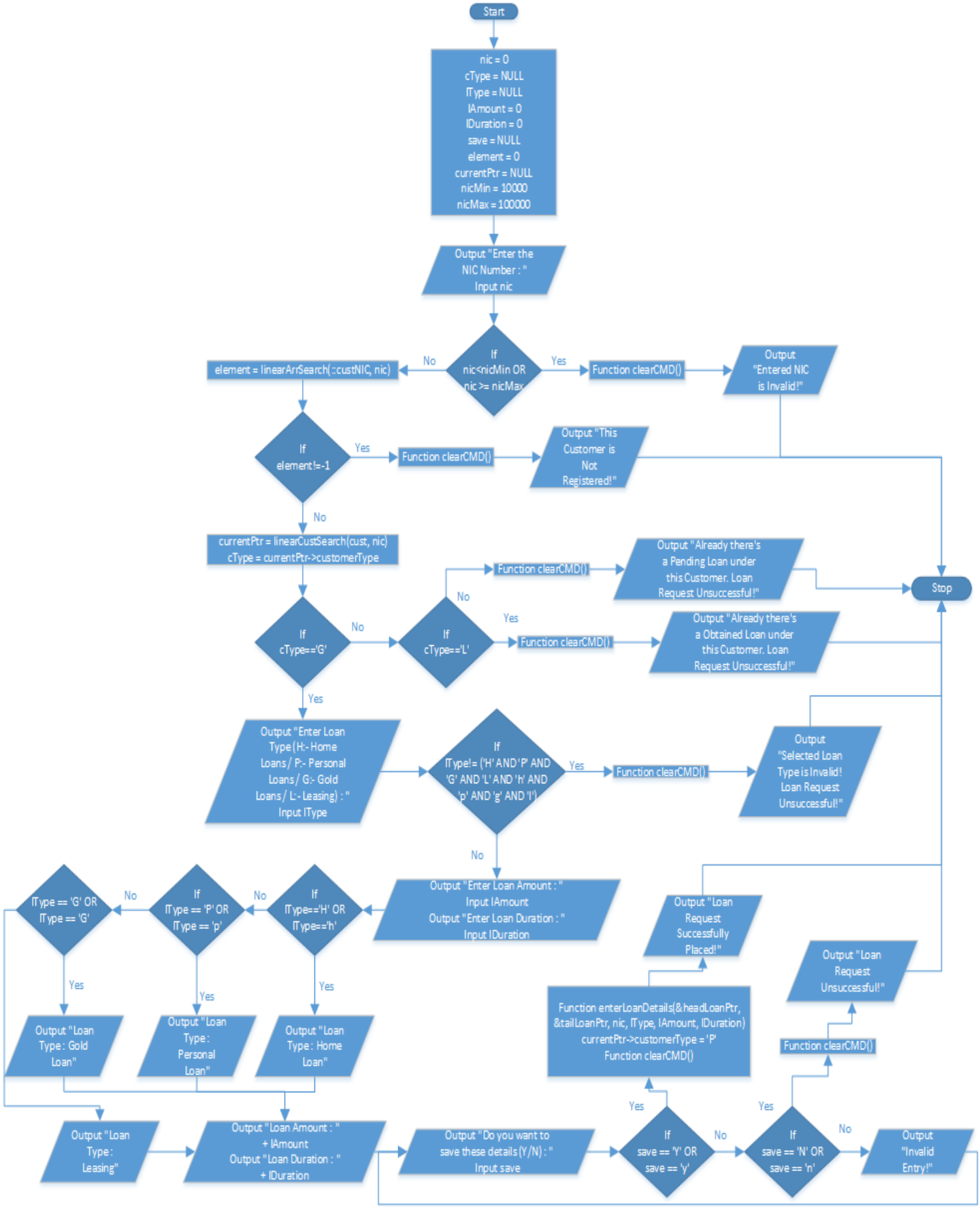
Start
nic = 0
cType = NULL
lType = NULL
lAmount = 0
lDuration = 0
save = NULL
element = 0
currentPtr = NULL
nicMin = 10000
nicMax = 100000
Output "Enter the NIC Number : "
Input nic
If nic<nicMin OR nic >= nicMax
Yes
Function clearCMD()
Output "Entered NIC is Invalid!"
No
element = linearArrSearch(□custNIC, nic)
If element!=-1
Yes
Function clearCMD()
Output "This Customer is Not Registered!"
No
currentPtr = linearCustSearch(cust, nic)
cType = currentPtr->customerType
If cType=='G'
No
If cType=='L'
Yes
Function clearCMD()
Output "Already there's a Obtained Loan under this Customer. Loan Request Unsuccessful!"
Function clearCMD()
Output "Already there's a Pending Loan under this Customer. Loan Request Unsuccessful!"
No
Stop
Yes
Output "Enter Loan Type (H:- Home Loans / P:- Personal Loans / G:- Gold Loans / L:- Leasing) : "
Input lType
If lType!= ('H' AND 'P' AND 'G' AND 'L' AND 'h' AND 'p' AND 'g' AND 'l')
Yes
Function clearCMD()
Output "Selected Loan Type is Invalid! Loan Request Unsuccessful!"
No
Output "Enter Loan Amount : "
Input lAmount
Output "Enter Loan Duration : "
Input lDuration
If lType == 'H' OR lType=='h'
If lType == 'P' OR lType == 'p'
If lType == 'G' OR lType == 'G'
No
Yes
Output "Loan Type : Home Loan"
Output "Loan Type : Personal Loan"
Output "Loan Type : Gold Loan"
Output "Loan Type : Leasing"
Output "Loan Amount : " + lAmount
Output "Loan Duration : " + lDuration
Output "Do you want to save these details (Y/N) : "
Input save
If save == 'Y' OR save == 'y'
If save == 'N' OR save == 'n'
Yes
No
Function enterLoanDetails(&headLoanPtr, &tailLoanPtr, nic, lType, lAmount, lDuration)
currentPtr->customerType = 'P'
Function clearCMD()
Output "Loan Request Successfully Placed!"
Function clearCMD()
Output "Loan Request Unsuccessful!"
Output "Invalid Entry!"

Figure 7

8. Approve / Reject Loan

```
BEGIN
    Initialize nic to ZERO
    Initialize save to NULL
    Initialize lType to NULL
    Initialize element to NULL
    Initialize currentPtr to headLoanPtr
    Initialize flag to false

    IF Function isEmpty(currentPtr) Then
        Output "There are no loan requests in the queue!"
    ELSE
        REPEAT
                nic = currentPtr->NIC
                lType = currentPtr->loanType

                Output "Customer NIC : " << nic
                IF lType == 'H' OR lType == 'h' Then
                    Output "Loan Type : Home Loan"
                ELSEIF lType == 'P' OR lType == 'p' Then
                    Output "Loan Type : Personal Loan"
                ELSEIF lType == 'G' OR lType == 'G' Then
                    Output "Loan Type : Gold Loan"
                ELSE
                    Output "Loan Type : Leasing"
                ENDIF

                Output "Loan Amount : " << currentPtr->loanAmount
                Output "Loan Duration : " << currentPtr->loanDuration

                REPEAT
                    Output "Do you want to Approve the Loan (Y / N / Q :- Exit from queue) : "
                    Input save
                    IF save == 'Y' OR save == 'y' OR save == 'N' OR save == 'n' Then
                        element = linearCustSearch(cust, nic)
                        IF save == 'Y' OR save == 'y' Then
                            element->loanType = currentPtr->loanType
                            element->loanAmount = currentPtr->loanAmount
                            element->loanDuration = currentPtr->loanDuration
                            element->customerType = 'L'
                            Output "Loan Approved."
                        ELSE
                            element->customerType = 'G'
                            Output "Loan Rejected."
                        ENDIF
                        Function deleteLoanDetails(&headLoanPtr, &tailLoanPtr)
                    ELSEIF save == 'Q' OR save == 'q' Then
                        Function clearCMD()
                        flag = true
                        Output "Exiting from loan queue...."
                    ELSE
                        Output "Invalid Entry!"
                    ENDIF
```

```
                    UNTIL save != 'Q' AND save != 'q' AND save == 'Y' AND save == 'y' AND save == 'N'
                                                                        AND save == 'n'
                    currentPtr = headLoanPtr
            UNTIL Function !isEmpty(currentPtr) AND (save != 'Q' AND save != 'q')
        ENDIF
END
```

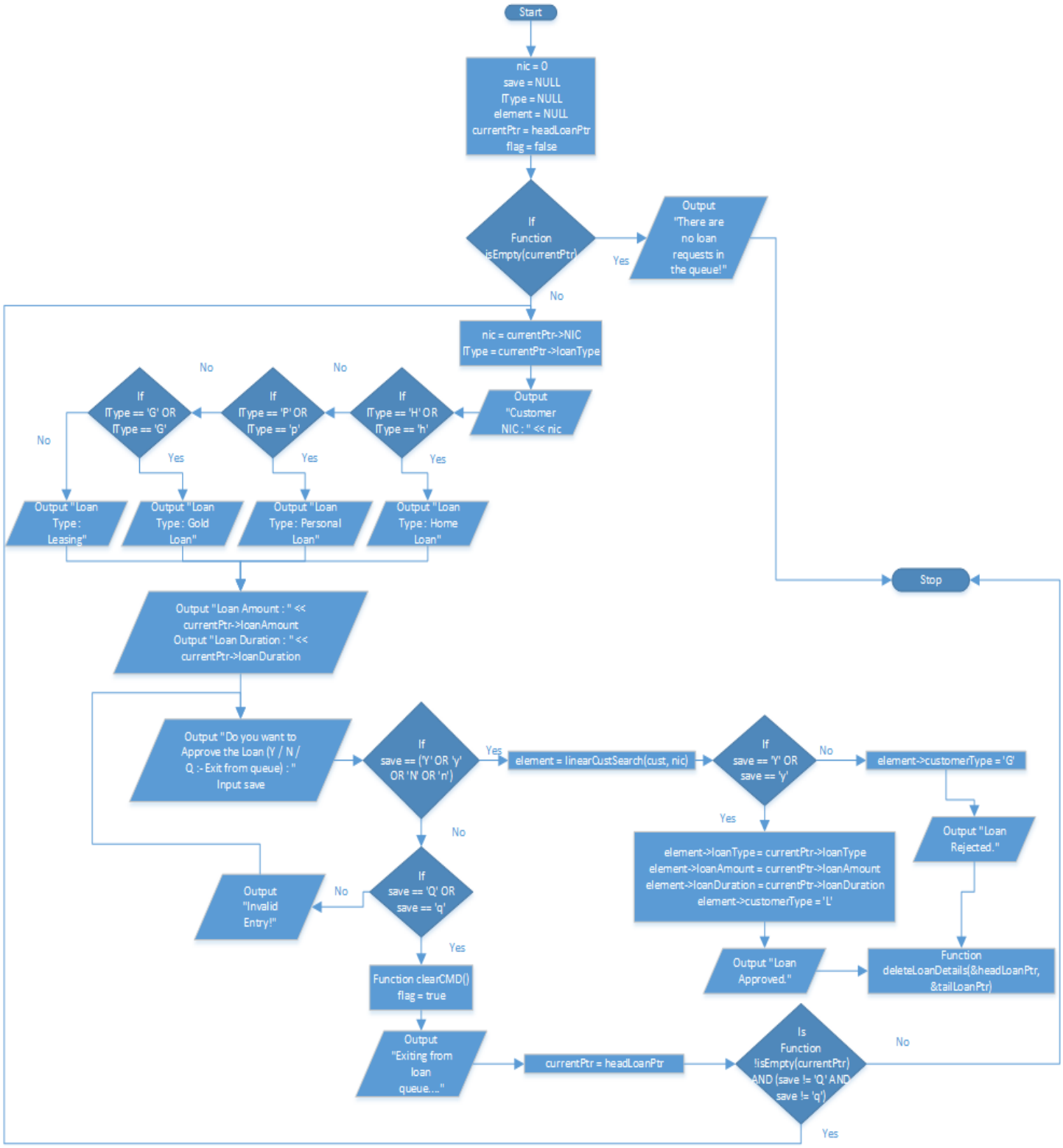

Figure 8

9. View Approved Customers

BEGIN
Function displayCustomerDetails('L')
END

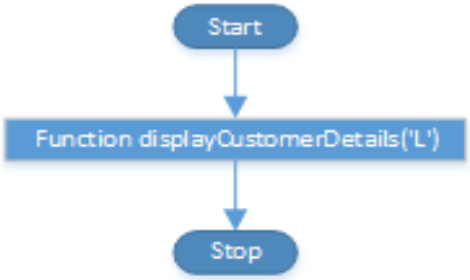

Figure 9

10. View Pending Customers

BEGIN
Function displayCustomerDetails('P')
END

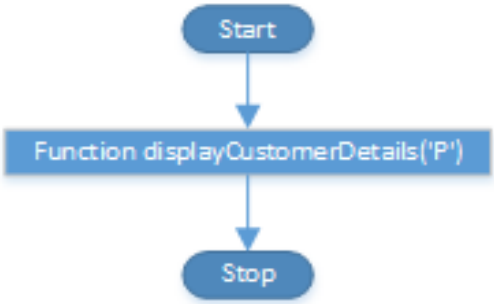

Figure 10

11. View General Customers

BEGIN
Function displayCustomerDetails('G')
END

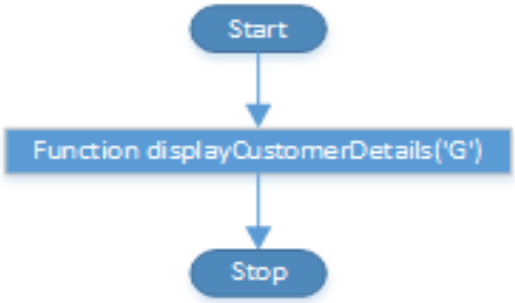

Figure 11

12. Linear NIC Array Search

```
BEGIN
        Initialize x to ZERO
        Initialize noOfCust to 100
        FOR x < noOfCust Then
                IF NIC[x] == searchKey Then
                        return x
                ENDIF
                x increment by 1
        ENDFOR
        return -1
END
```

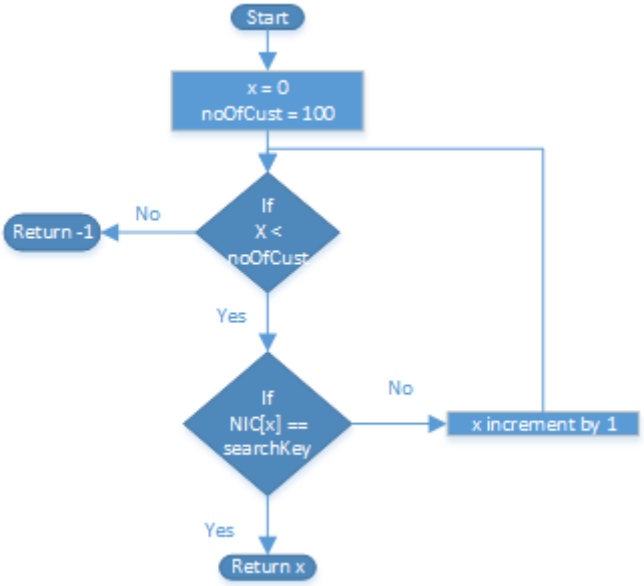

Figure 12

13. Linear Customer Details Search

```
BEGIN
        Initialize currentPtr to sPtr
        WHILE Function !isEmpty(currentPtr) Then
                IF currentPtr->NIC == nic Then
                        return currentPtr
                ENDIF
                currentPtr = currentPtr->nextCustPtr
        ENDWHILE
        return NULL
END
```

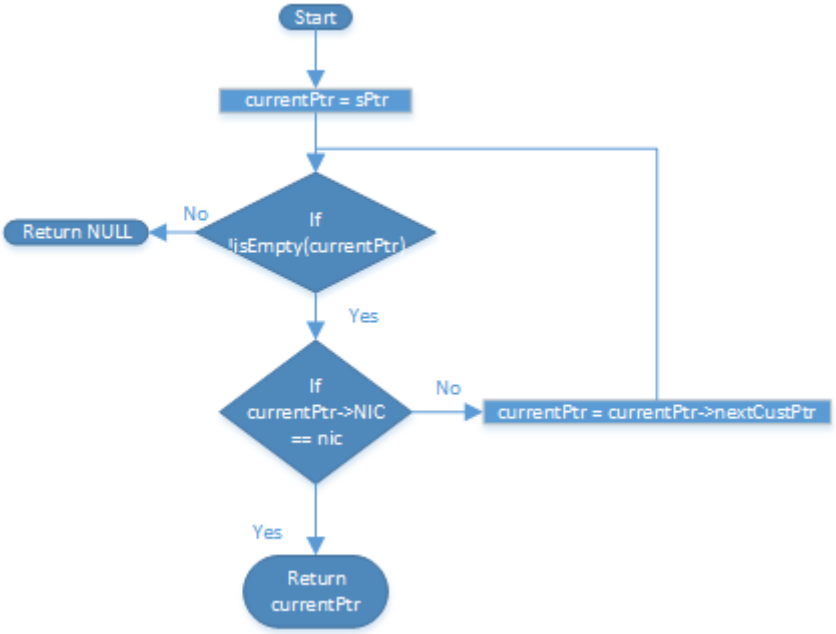

Figure 13

14. Array Sort

```
BEGIN
        Initialize temp to ZERO
        Initialize x to ONE
        Initialize y to ZERO
        Initialize noOfCust to 100
        FOR  x < noOfCust Then
                FOR  y < noOfCust-1 Then
                        IF NIC[y]>NIC[y+1] Then
                                temp = NIC[y]
                                NIC[y] = NIC[y + 1]
                                NIC[y + 1] = temp
                        ENDIF
                        y increment by 1
                ENDFOR
                x increment by 1
        ENDFOR
END
```

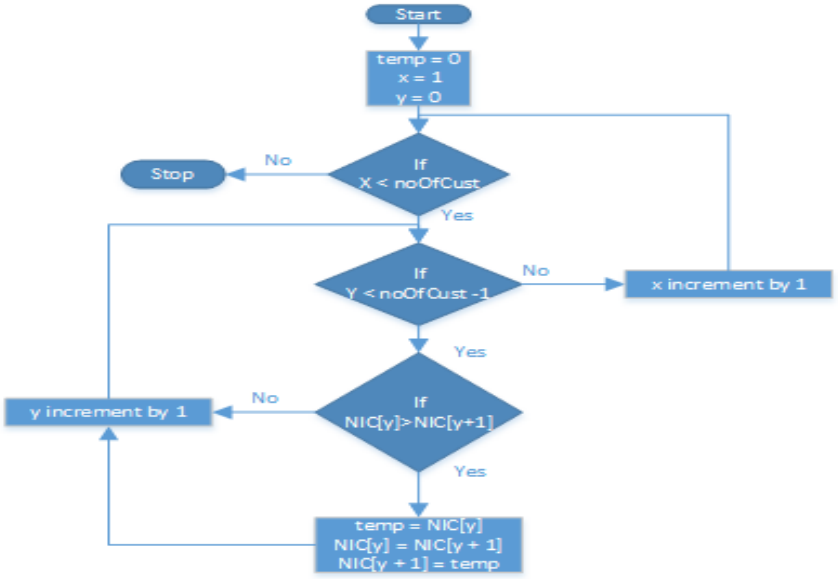

Figure 14

15. Display Customer Details

```
BEGIN
        Function clearCMD()
        Initialize flag to true
        Initialize currentPtr to cust
        IF type=='G' OR type == 'A' Then
                Output "Customer NIC" + “       “ + "Customer Name"
        ELSE
                Output "Customer NIC" + “       “ + "Customer Name" + “       “ + "Loan Amount"
        ENDIF
        WHILE Function !isEmpty(currentPtr) Then
                IF type == 'A' OR (currentPtr->customerType == 'G' AND type == 'G') Then
                        flag = false
                        Output "      " + currentPtr->NIC + "       " + currentPtr->customerName
                ELSEIF currentPtr->customerType == 'P' AND type == 'P' Then
                        flag = false
                        Output "      " + currentPtr->NIC + "       " + currentPtr->customerName
                        + "       " + loanAmount(headLoanPtr, currentPtr->NIC)
                ELSEIF currentPtr->customerType == 'L' AND type == 'L' Then
                        flag = false
                        Output "      " + currentPtr->NIC + "       " + currentPtr->customerName
                        + "       " + currentPtr->loanAmount
                ENDIF
                currentPtr = currentPtr->nextCustPtr
        ENDWHILE
        IF flag == true Then
                Output "No Record Found!"
        ENDIF
END
```

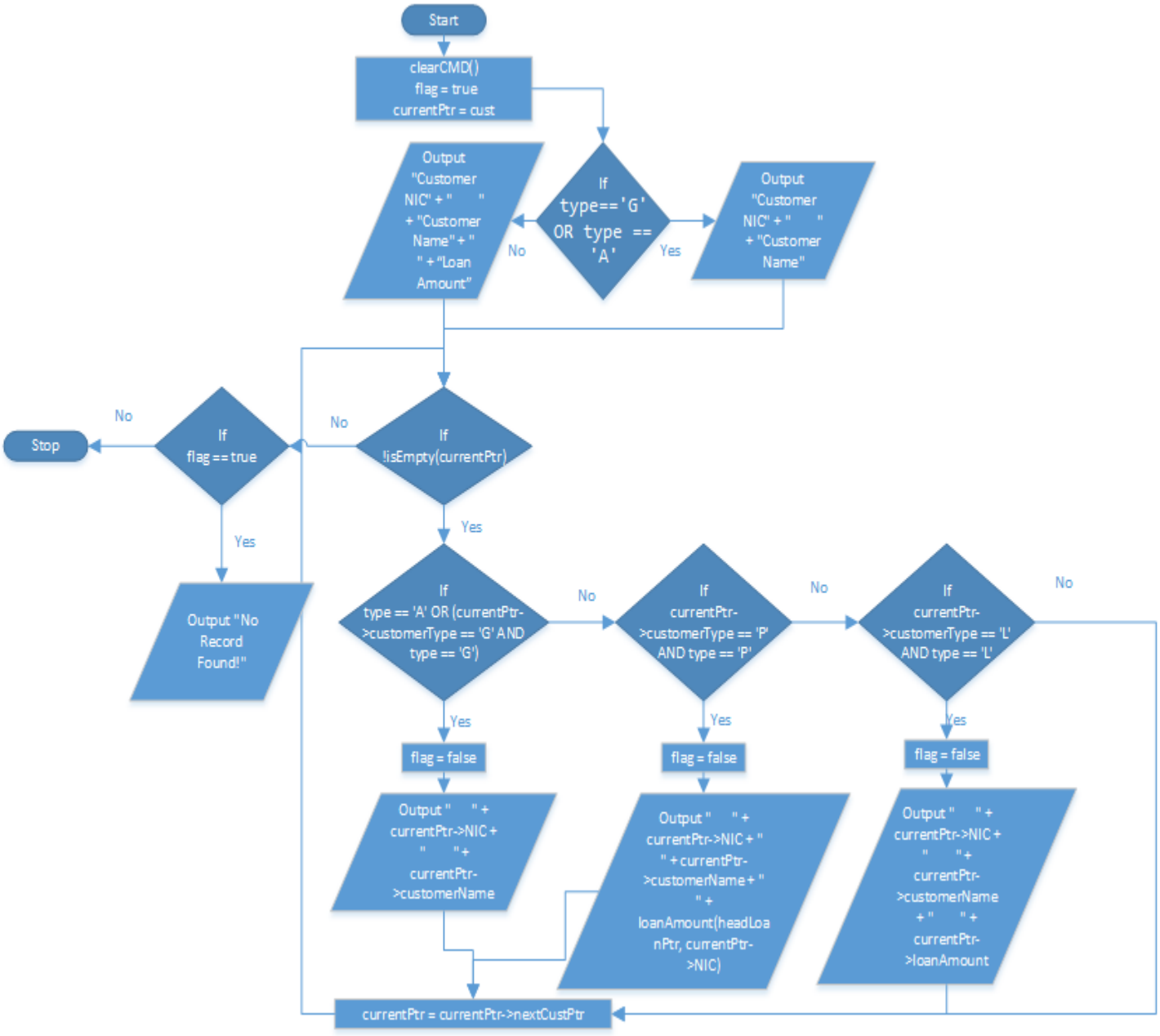
Start
clearCMD()
flag = true
currentPtr = cust
If type=='G' OR type == 'A'
Yes
Output "Customer NIC" + " " + "Customer Name"
No
Output "Customer NIC" + " " + "Customer Name" + " " + "Loan Amount"
If !isEmpty(currentPtr)
No
If flag == true
No
Stop
Yes
Output "No Record Found!"
Yes
If type == 'A' OR (currentPtr->customerType == 'G' AND type == 'G')
No
If currentPtr->customerType == 'P' AND type == 'P'
No
If currentPtr->customerType == 'L' AND type == 'L'
No
Yes
flag = false
Output " " + currentPtr->NIC + " " + currentPtr->customerName
Yes
flag = false
Output " " + currentPtr->NIC + " " + currentPtr->customerName + " " + loanAmount(headLoanPtr, currentPtr->NIC)
Yes
flag = false
Output " " + currentPtr->NIC + " " + currentPtr->customerName + " " + currentPtr->loanAmount
currentPtr = currentPtr->nextCustPtr

Figure 15

16. Loan Amount

```
BEGIN
        Initialize current to headLoanPtr
        WHILE current!=NULL Then
                IF current->NIC == nic Then
                        return current->loanAmount
                ENDIF
                current = current->nextLoanPtr
        ENDWHILE
        return NULL
END
```

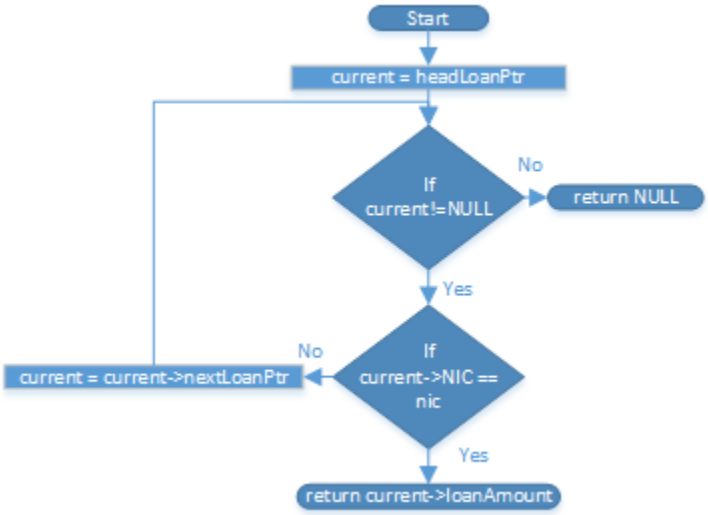

Figure 16

4. Implementation

This section explains all the functions of the program. This section explains the code section by section with its execution process.

```
1   #include <iostream>
2   #include <iomanip>
3   #define nicMax 100000 // NIC maximum number
4   #define nicMin 10000 // NIC minimum number
5
6   using namespace std;
```

Figure 17

1. #include <iostream> = instructs the preprocessor to include a section of standard C++ code, this allows performing standard input and output operations
2. #include <iomanip> = contains formatting manipulators, this allows performing white spaces between words
3. #define nicMax 100000
4. #define nicMin 10000

} This preprocessor directive creates symbolic constants.

Data Structure

```
8   //Data Structures and Pointers
9   struct customer {
10      int NIC;
11      char customerName[20];
12      char customerAddress[50];
13      int customerAge;
14      char customerGender;
15      char customerType;
16      char loanType;
17      double loanAmount;
18      int loanDuration;
19      struct customer *nextCustPtr;
20  };
21
22  typedef struct customer CustomerNode;
23  typedef CustomerNode *customerPtr;
```

Figure 18

This data structure stores the Customer Main details (NIC, Name, Address, Age, Gender, Customer Type). Also, when the Customer requested Loan approved that loan details will be stored in this Main Customer data structure.

In this data structure, we're using Data Structure of Link list. So, when we're going to access to next Customer details we're using a pointer inside the Data Structure to access to the next Customer details (Line 19).

Line 22 and 23 shows the customer Data Structure Node and Its pointer.

```
25  struct loanQueue {
26      int NIC;
27      char loanType;
28      double loanAmount;
29      int loanDuration;
30      struct loanQueue *nextLoanPtr;
31  };
32
33  typedef struct loanQueue LoanQueueNode;
34  typedef LoanQueueNode *loanQueuePtr;
```

Figure 19

This data structure stores the Loan details (NIC, Loan Type, Amount, Duration). When Customer request a Loan that loan details will store in this data structure.

In this data structure, we're using Data Structure of Queue. So, when we're going to access to next Loan details we're using a pointer inside the Data Structure to access to the next Loan details (Line 30).

Line 33 and 34 show the Loan Data Structure Node and Its pointer.

Function Prototype

```
36  //Function Prototypes
37  void displayMenu();
38  void registerCustomer();
39  void unregisterCustomer();
40  void updateCustomer();
41  void viewCustomer();
42  void viewAllCustomer();
43  void applyLoan();
44  void appRejLoan();
45  void viewAppCustomer();
46  void viewPendingCustomer();
47  void viewGeneralCustomer();
```

Figure 20

These are the main function prototypes in the program.

```
49      //Linear Array Search
50      int linearArrSearch(int[],int);
51      //Linear Customer Search
52      customerPtr linearCustSearch(customerPtr, int);
53      //Sorting
54      void Sort(int[]);
55      //Command prompt clear command
56      void clearCMD();
57      //Customer Details display
58      void displayCustomerDetails(char);
59      //Getting Loan Amount from Queue
60      double loanAmount(loanQueuePtr , int);
```

Figure 21

These functions perform the Array Linear Search, Customer Linear Search, Array Sorting, Display details clearing function (clearCMD) and All the Customer details displaying function.

```
62      //Link List
63      bool insertCustDetails(customerPtr *, int, char[20], char[50], int, char, char);
64      int deleteCustomerDetails(customerPtr *, int);
65      int isEmpty(customerPtr);
```

Figure 22

These functions perform the Data Structure of Link list.

```
67      //Queue
68      bool enterLoanDetails(loanQueuePtr *, loanQueuePtr *, int, char, double , int );
69      void deleteLoanDetails(loanQueuePtr *,loanQueuePtr *);
70      int isEmpty(loanQueuePtr);
```

Figure 23

These functions perform the Data Structure of Queue.
A function prototype is just another name for a declaration of a function.

Global Variables

```
72      //Global Variables
73      loanQueuePtr headLoanPtr = NULL;
74      loanQueuePtr tailLoanPtr = NULL;
75      customerPtr cust = NULL;
76      const int noOfCust = 100; // No of customers
77      int custNIC[noOfCust];
```

Figure 24

1. headLoanPtr = This is the Data Structure of Queue start pointer
2. tailLoanPtr = This is the Data Structure of Queue end pointer
3. cust = This is the Data Structure of Link list start pointer
4. noOfCust = This is a constant variable defines the size of the NIC Array
5. custNIC = This is the NIC Array used in this program

Main Menu

```
79  int main(){
80      int option = 0;
81
82      for (int x = 0; x < noOfCust;x++) {
83          ::custNIC[x] = nicMax;
84      }
85
86      do{
87          //Display Menu
88          displayMenu();
89          cout << "Select an Option : ";
90
91          cin >> option;
92          cout << endl;
93
94          switch (option){
95              case 1:
96                  //Register Customer
97                  cout << "Selected Option 1) Register Customer" << endl << endl;
98                  registerCustomer();
99                  break;
100             case 2:
101                 //Unregister Customer
102                 cout << "Selected Option 2) Un-Register Customer" << endl << endl;
103                 unregisterCustomer();
104                 break;
105             case 3:
106                 //Update Customer
107                 cout << "Selected Option 3) Update Customer" << endl << endl;
108                 updateCustomer();
109                 break;
110             case 4:
111                 //View Customer
112                 cout << "Selected Option 4) View Customer" << endl << endl;
113                 viewCustomer();
114                 break;
115             case 5:
116                 //View All Customers
117                 cout << "Selected Option 5) View All Customers" << endl << endl;
118                 viewAllCustomer();
119                 break;
120             case 6:
121                 //Apply for a loan
122                 cout << "Selected Option 6) Apply for a Loan" << endl << endl;
123                 applyLoan();
124                 break;
125             case 7:
126                 //Approve/Reject Loan
127                 cout << "Selected Option 7) Approve/Reject Loan" << endl << endl;
128                 appRejLoan();
129                 break;
130             case 8:
131                 //View approved Customers
132                 cout << "Selected Option 8) View Approved Customers" << endl << endl;
133                 viewAppCustomer();
134                 break;
135             case 9:
136                 //View Pending Customers
137                 cout << "Selected Option 9) View Pending Customers" << endl << endl;
138                 viewPendingCustomer();
139                 break;
140             case 10:
141                 //View General Customers1
142                 cout << "Selected Option 10) View General Customers" << endl << endl;
143                 viewGeneralCustomer();
144                 break;
145             default:
146                 cout << "Selected Option is Invalid!" << endl << endl;
147         }
148         cout << endl << endl;
149
150     } while (option!=0);
151
152     //getchar();
153     return 0;
154 }
```

Figure 25

Main Menu will first initialize Array values to a default value using a variable name called “nicMax”. Then it will display the Welcome Menu with available options by calling the function name called “displayMenu”.

When the user enters a number between 1 to 10 program will go through a switch case statement and proceed to display a message with some function. If the user enters number zero then the program will terminate. If the user enters a number not between 0 to 10 program will display an error message and request to re-enter a number again.

Display Menu

```
156  void displayMenu(){
157      //Display Menu Options
158      cout << "================================================" << endl << endl;
159      cout << setw(27) << "Bank Of Ceylon" << endl << endl;
160      cout << setw(23) << "Welcome" << endl << endl;
161      cout << setw(23) << "1) Register Customer" << endl;
162      cout << setw(26) << "2) Un-Register Customer" << endl;
163      cout << setw(21) << "3) Update Customer" << endl;
164      cout << setw(19) << "4) View Customer" << endl;
165      cout << setw(24) << "5) View All Customers" << endl;
166      cout << setw(22) << "6) Apply for a Loan" << endl;
167      cout << setw(25) << "7) Approve/Reject Loan" << endl;
168      cout << setw(29) << "8) View Approved Customers" << endl;
169      cout << setw(28) << "9) View Pending Customers" << endl;
170      cout << setw(29) << "10) View General Customers" << endl;
171      cout << setw(10) << "0) Exit" << endl << endl;
172  }
```

Figure 26

Display Menu function will display the Welcome Message and the available option.

Register Customer

```
174  void registerCustomer(){
175      int nic = 0; //Customer NIC
176      char cName[20] = ""; // Customer Name
177      char address[50] = ""; // Customer Address
178      int age = 0; // Customer age
179      char gender = NULL; //Customer gender
180      char save = NULL;
181      bool flag = true;
182      int element = 0;
183
184      cout << "Enter the NIC Number : ";
185      cin >> nic;
186      cin.ignore(); //This use to ignore the input stream charactSers manually
187
188      if (nic<nicMin || nic>= nicMax){ //Checking NIC is a 5 digit number
189          clearCMD();
190          cout << "Entered NIC is Invalid!" << endl;
191      }
192      else{
193          //Checking Customer already registered
194          element = linearArrSearch(::custNIC, nic);
195          if (element!=-1){
196              clearCMD();
197              cout << "Customer already Registered." << endl;
198          }
199          else{
200              //Program request user to enter Customer Details
201              cout << "Enter Customer Name : ";
202              cin.get(cName, 20);
203              cin.ignore();
204
205              cout << "Enter Customer Address : ";
206              cin.get(address, 40);
207
208              cout << "Enter Customer Age : ";
209              cin >> age;
210
211              cout << "Enter Customer Gender (M :- Male / F :- Female): ";
212              cin >> gender;
213
214              //Checking Enter Gender Type Correct
215              if((gender != 'M') && (gender != 'm') && (gender != 'f') && (gender != 'F')){
216                  clearCMD();
217                  cout << "Selected Gender is Invalid!" << endl;
218                  cout << "Registration Unsuccessful." << endl;
219              }else{
220                      //Displaying entered customer details
221                      cout << endl;
222                      cout << "Customer Name : " << cName << endl;
223                      cout << "Customer Address : " << address << endl;
224                      cout << "Customer Age : " << age << endl;
225
226                      if ((gender == 'M') || (gender == 'm')) {
227                          cout << "Customer Gender : Male" << endl;
228                      }
229                      else {
230                          cout << "Customer Gender : Female" << endl;
231                      }
232
233                      do{
234                          //Requesting user to confirm entered Customer details correct
235                          cout << "Do you want to save these details (Y/N) : ";
236                          cin >> save;
237                          if ((save == 'Y') || (save == 'y')) {
238                              clearCMD();
239                              //Storing Customer Details in the System
240                              flag = insertCustDetails(&cust, nic, cName, address, age, gender, 'G');
241                              //Sorting customer NIC array
242                              Sort(::custNIC);
243                              if (flag) {
244                                  cout << "Registered Successfully!" << endl;
245                              }
246                              else {
247                                  cout << "Registration Unsuccessful." << endl;
248                              }
249                          }
250                          else if ((save == 'N') || (save == 'n')) {
251                              //User Decline to save the Customer Details
252                              clearCMD();
253                              cout << "Registration Unsuccessful." << endl;
254                          }
255                          else {
256                              cout << "Invalid Entry!" << endl;
257                          }
258                          //Until User Enter Y/N character it will go through a loop
259                      } while ((save != 'N') && (save != 'n') && (save != 'Y') && (save != 'y'));
260              }
261          }
262      }
263  }
264
```

Figure 27

“Register Customer” is a function where Customer Register into the “Bank of Ceylon” System. The first program will set local variables to a default value. Then it requests the user to enter Customer NIC number, so the program first checks if that entered NIC is in the range of 5-digit number. If NIC, not a 5-digit Number program will prompt a message “Entered NIC is Invalid!”. If NIC is a 5-digit number it will proceed to the next step.

The program checks if that customer is already registered in the System. If Customer Already Registered, System will Prompt a message "Customer already Registered.". In this Program Checks Customer available in the System by doing an “Array Linear Search”. To do this process there’s separate method called “LinearArrSearch”.

If the Customer Not Registered in the System, the program will request the user to enter Customer Details. The user needs to enter following details.

1. Customer Name
2. Customer Address
3. Customer Age
4. Customer Gender

In this Customer Name can be maximum 20 Characters and Customer Address can be maximum 50 Characters. In Customer Gender program request user to enter’ or ‘F’ Character. So, the program will identify’ as Male and ‘F’ as Female. If the user enters some other Character, the program will prompt a message "Selected Gender is Invalid! And Registration Unsuccessful.". So, the Customer Registration process will terminate by the System.

When the Customer Details registering initially Customer Type will be ‘G’ – General Customer. The user will not enter it manually, the program will automatically set it to ‘G’ when Customer Registering.

If the user enters Customer Details correctly program will prompt Customer Name, Address, Age, Gender, and Type. After that program requests users to confirm that entered Customer Details are correct. The user needs to enter ‘Y’ or ‘N’ (Yes or No) Character to proceed that step. If the user enters Character except ‘Y’ and ‘N’ program will prompt a message "Invalid Entry!" and the program will go through a loop until the user enters one of the valid Character.

If the user enters ‘N’ program will prompt a message "Registration Unsuccessful." and Customer Details will not be stored in the System. If the user enters ‘Y’ program will prompt a message "Registered Successfully!" and Customer details will be stored in sorted order of NICs. This storing will is done by a method called “insertCustDetails”.

After completing Register Customer program go back the Main Menu. Before that, it will clear the previous prompted details by calling to the function called “clearCMD”.

Un-Register Customer

```
265 void unregisterCustomer(){
266     int nic = 0; // customer nic
267     int status = 0;
268 
269     cout << "Enter the NIC Number : ";
270     cin >> nic;
271     if (nic<nicMin || nic >= nicMax){ //Checking NIC is a 5 digit number
272         clearCMD();
273         cout << "Entered NIC is Invalid!" << endl;
274     }
275     else{
276         //Checking customer already registered
277         status = linearArrSearch(::custNIC, nic);
278         if (status!=-1) {
279             //Finding customer details and remove
280             status = deleteCustomerDetails(&cust, nic);
281             if (status == 0) {
282                 cout << "Customer Un-Registered successfully!" << endl;
283                 Sort(::custNIC);
284             }
285             else if(status == 1) {
286                 cout << "This Customer has obtained a Loan!\nCustomer Un-Registration Unsuccessful." << endl;
287             }
288             else {
289                 cout << "This Customer has requested a Loan!\nCustomer Un-Registration Unsuccessful." << endl;
290             }
291         }
292         else {
293             cout << "This Customer is Not Registered!" << endl;
294         }
295     }
296 }
297 
```

Figure 28

“Un-Register Customer” is a function where Customer Un-Register from the System. First program request user to enter Customer NIC number, so the program first checks if that entered NIC is in the range of 5-digit number. If NIC, not a 5-digit Number program will prompt a message “Entered NIC is Invalid!”. If NIC is a 5-digit number it will proceed to the next step.

Then program checks if that customer is registered in the System. If Customer not Registered System will Prompt a message "This Customer is Not Registered!". If Customer registered then the program will process a function called “deleteCustomerDetails” and return a status code. Status codes are following:

- Status 0: - Customer registered in the system and that customer unregistered successfully
- Status 1: - Customer registered in the system but Customer has obtained a loan. Because of that customer cannot unregister from the system
- Status 2: - Customer registered in the system but Customer has requested a loan. Because of that customer cannot unregister from the system

If the Status 0 program will prompt a message “Customer Un-Registered successfully!”. If the Status 1, program will prompt a message “This Customer has obtained a Loan! And Customer Un-Registration Unsuccessful.”. If the Status 2, program will prompt a message “This Customer has requested a Loan! And Customer Un-Registration Unsuccessful.”

After completing Register Customer program go back the main menu. Before that, it will clear the previous prompted details.

Update Customer

```
298  void updateCustomer(){
299      int nic = 0 ; // customer nic
300      char address[40] = "";
301      int element = 0;
302      customerPtr currentPtr = NULL;
303
304      cout << "Enter the NIC Number : ";
305      cin >> nic;
306      cin.ignore();
307
308      if (nic<nicMin || nic >= nicMax){ //Checking NIC is a 5 digit number
309          clearCMD();
310          cout << "Entered NIC is Invalid!" << endl;
311      }
312      else{
313          //Checking customer registered
314          element = linearArrSearch(::custNIC, nic);
315          clearCMD();
316          if (element != -1){
317              //Request to enter new customer address
318              cout << "Enter New Address : ";
319              cin.get(address, 40);
320
321              currentPtr = linearCustSearch(cust, nic);
322
323              //Updating Customer Address
324              strcpy_s(currentPtr->customerAddress, address);
325              cout << "Customer Details Updated Sucessfully!" << endl;
326          }
327          else{
328              cout << "This Customer is Not Registered!" << endl;
329          }
330      }
331  }
332
```

Figure 29

"Update Customer" is a function where Customer Address update from the System. First program request user to enter Customer NIC number, so the program first checks if that entered NIC is in the range of 5-digit number. If NIC, not a 5-digit Number program will prompt a message "Entered NIC is Invalid!". If NIC is a 5-digit number it will proceed to the next step.

The program checks if that customer is registered in the System. If Customer Not Registered System will Prompt a message "This Customer is Not Registered!". If Customer registered then program request enters new Customer Address. Then that Customer Address will be replaced with the new one and prompt a message "Customer Details Updated Successfully!".

View Customer

```
333  void viewCustomer(){
334      int nic = 0; // customer nic
335      char gender = NULL;
336      char type = NULL;
337      char lType = NULL;
338      int element = 0;
339      customerPtr currentPtr = NULL;
340
341      cout << "Enter the NIC Number : ";
342      cin >> nic;
343
344      if (nic<nicMin || nic >= nicMax){ //Checking NIC is a 5 digit number
345          clearCMD();
346          cout << "Entered NIC is Invalid!" << endl;
347      }
348      else{
349          //Checking customer registered
350          element = linearArrSearch(::custNIC, nic);
351          clearCMD();
352          if (element != -1){
353              currentPtr = linearCustSearch(cust, nic);
354              //Display customer Details
355              cout << endl;
356              cout << "Customer Name : " << currentPtr->customerName << endl;
357              cout << "Customer Address : " << currentPtr->customerAddress << endl;
358              cout << "Customer Age : " << currentPtr->customerAge << endl;
359
360              gender = currentPtr->customerGender;
361              if ((gender=='M') || (gender=='m')) {
362                  cout << "Customer Gender : Male" << endl;
363              }
364              else {
365                  cout << "Customer Gender : Female" << endl;
366              }
367
368              type = currentPtr->customerType;
369              if (type=='G') {
370                  cout << "Customer Type : General Customer" << endl;
371              }
372              else if (type=='P') {
373                  cout << "Customer Type : Pending Customer" << endl;
374              }
375              else {
376                  cout << "Customer Type : Loan Customer" << endl;
377                  lType = currentPtr->loanType;
378                  if (lType == 'H' || lType == 'h') {
379                      cout << "Loan Type : Home Loan" << endl;
380                  }
381                  else if (lType == 'P' || lType == 'p') {
382                      cout << "Loan Type : Personal Loan" << endl;
383                  }
384                  else if (lType == 'G' || lType == 'G') {
385                      cout << "Loan Type : Gold Loan" << endl;
386                  }
387                  else {
388                      cout << "Loan Type : Leasing" << endl;
389                  }
390                  cout << "Loan Amount : " << currentPtr->loanAmount << endl;
391                  cout << "Loan Duration : " << currentPtr->loanDuration << endl;
392              }
393              cout << endl;
394          }
395          else{
396              cout << "This Customer is Not Registered!" << endl;
397          }
398      }
399  }
400
```

Figure 30

"View Customer" is a function where Customer Details can be viewed. First program request user to enter Customer NIC number, so the program first checks if that entered NIC is in the range of 5-digit number. If NIC, not a 5-digit Number program will prompt a message "Entered NIC is Invalid!". If NIC is a 5-digit number it will proceed to the next step.

The program checks if that customer is registered in the System. If Customer Not Registered System will Prompt a message "This Customer is Not Registered!". If Customer registered then the program will prompt Customer Name, Address, Age, Gender, and Type.

If Customer type is 'P' or 'L' (Pending Customer or Loan Customer) system will prompt loan details.

View All Customers

```
401    void viewAllCustomer(){
402        displayCustomerDetails('A');
403    }
```

Figure 31

"View All Customer" is a function where All the Registered Customers Details can be viewed. In this function it will forward to the "displayCustomerDetails" function with parameter value 'A', So the program will prompt all the Customers Registered in the System. The System will display following Customer details:

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

Apply for a Loan

```
405  void applyLoan(){
406      int nic = 0; // customer nic
407      char cType = NULL; //Customer type
408      char lType = NULL; // Loan type
409      double lAmount = 0; // Loan amount
410      int lDuration = 0; //Loan Duration
411      char save = NULL;
412      int element = 0;
413      customerPtr currentPtr = NULL;
414
415      cout << "Enter the NIC Number : ";
416      cin >> nic;
417      if (nic<nicMin || nic >= nicMax) { //Checking NIC is a 5 digit number
418          clearCMD();
419          cout << "Entered NIC is Invalid!" << endl;
420      }
421      else {
422          element = linearArrSearch(::custNIC, nic);
423          if (element!=-1) {
424              currentPtr = linearCustSearch(cust, nic);
425
426              cType = currentPtr->customerType;
427              //Checking Customer is a General Customer. Only General Customer can request for a loan
428              if (cType=='G') {
429                  cout << "Enter Loan Type (H:- Home Loans / P:- Personal Loans / G:- Gold Loans / L:- Leasing) : ";
430                  cin >> lType;
431                  if ((lType!='H') && (lType != 'P') && (lType != 'G') && (lType != 'L') &&
432                      (lType != 'h') && (lType != 'p') && (lType != 'g') && (lType != 'l')) {
433                      clearCMD();
434                      cout << "Selected Loan Type is Invalid!\nLoan Request Unsuccessful!" << endl;
435                  }
436                  else {
437                      cout << "Enter Loan Amount : ";
438                      cin >> lAmount;
439                      cout << "Enter Loan Duration : ";
440                      cin >> lDuration;
441
442                      cout << endl;
443                      //Display entered loan details (Type, Amount and Duration)
444                      if (lType=='H' || lType=='h') {
445                          cout << "Loan Type : Home Loan" << endl;
446                      }
447                      else if(lType == 'P' || lType == 'p'){
448                          cout << "Loan Type : Personal Loan" << endl;
449                      }
450                      else if (lType == 'G' || lType == 'G') {
451                          cout << "Loan Type : Gold Loan" << endl;
452                      }
453                      else {
454                          cout << "Loan Type : Leasing" << endl;
455                      }
456                      cout << "Loan Amount : " << lAmount << endl;
457                      cout << "Loan Duration : " << lDuration << endl;
458
459                      do {
460                          //Entered Loan details save request
461                          cout << "Do you want to save these details (Y/N) : ";
462                          cin >> save;
463                          if (save == 'Y' || save == 'y') {
464                              //Storing Loan Request Details in a Queue
465                              enterLoanDetails(&headLoanPtr, &tailLoanPtr, nic, lType, lAmount, lDuration);
466                              currentPtr->customerType = 'P'; //Changing Customer Type
467                              clearCMD();
468                              cout << "Loan Request Successfully Placed!" << endl;
469                          }
470                          else if (save == 'N' || save == 'n') {
471                              clearCMD();
472                              cout << "Loan Request Unsuccessful!" << endl;
473                          }
474                          else {
475                              cout << "Invalid Entry!" << endl;
476                          }
477                      } while (save != 'Y' && save != 'y' && save != 'N' && save != 'n');
478                  }
479              }
480              else if(cType=='L'){
481                  clearCMD();
482                  cout << "Already there's a Obtained Loan under this Customer.\nLoan Request Unsuccessful!" << endl;
483              }
484              else {
485                  clearCMD();
486                  cout << "Already there's a Pending Loan under this Customer.\nLoan Request Unsuccessful!" << endl;
487              }
488          }
489          else {
490              clearCMD();
491              cout << "This Customer is Not Registered!" << endl;
492          }
493      }
494  }
495
```

Figure 32

“Apply for a Loan” is a function where Customer can request for a Loan. First program request user to enter Customer NIC number, so the program first checks if that entered NIC is in the range of 5-digit number. If NIC, not a 5-digit Number program will prompt a message “Entered NIC is Invalid!”. If NIC is a 5-digit number it will proceed to the next step.

Then program checks if that customer is registered in the System. If Customer not Registered System will Prompt a message "This Customer is Not Registered!". If Customer registered then the program will check the Customer type, so the program can identify is this Customer has already obtained a loan or requested a loan. If the Customer already obtained a loan, System will prompt a message "Already there's a Loan under this Customer and Loan Request Unsuccessful!". If the Customer already has a pending loan request, System will prompt a message “Already there's a Pending-Loan under this Customer. And Loan Request Unsuccessful!”.

If this Customer is a General Customer (No loan obtained or request under the Customer) then the program will request to enter following details.

1. Loan Type
2. Loan Amount
3. Loan Duration

There are 4 loan types used in the System.

- H: - Home Loans
- P: - Personal Loans
- G: - Gold Loans
- L: - Leasing

Program request to enter one of above Loan type character. If the user enters some other character program will prompt a message “Selected Loan Type is Invalid! and Loan Request Unsuccessful!”.

If the user enters valid Loan type program request to enter Loan Amount and Loan Duration. To check entered the Loan details correct program will prompt Loan Type, Amount, and Duration. After that program requests the user to confirm that entered Loan Details are correct. The user needs to enter ‘Y’ or ‘N’ (Yes or No) Character to proceed that step.

If the user enters Character except ‘Y’ and ‘N’ program will prompt a message "Invalid Entry!" and the program will go through a loop until the user enters one of the Characters. If the user enters ‘N’ program will prompt a message "Loan Request Unsuccessful!" and Loan Details will not be stored in the System. If the user enters ‘Y’ program will prompt a message "Loan Request Successfully Placed!" and Loan details will be stored in a Queue. This storing will is done by a method called “enterLoanDetails”.

After placing the Loan request program go back to the main menu. Before that, it will clear the previous prompted details.

Approve / Reject Loan

```
496  void appRejLoan(){
497      int nic = 0;
498      char save = NULL;
499      char lType = NULL;
500      customerPtr element = NULL;
501      loanQueuePtr currentPtr = headLoanPtr;
502      bool flag = false; // Use to end the queue
503
504      if (isEmpty(currentPtr)) {
505          cout << "There are no loan requests in the queue!" << endl;
506      }
507      else {
508              do{
509                  nic = currentPtr->NIC;
510                  lType = currentPtr->loanType;
511
512                  cout << "Customer NIC : " << nic << endl;
513
514                  if (lType == 'H' || lType == 'h') {
515                      cout << "Loan Type : Home Loan" << endl;
516                  }
517                  else if (lType == 'P' || lType == 'p') {
518                      cout << "Loan Type : Personal Loan" << endl;
519                  }
520                  else if (lType == 'G' || lType == 'G') {
521                      cout << "Loan Type : Gold Loan" << endl;
522                  }
523                  else {
524                      cout << "Loan Type : Leasing" << endl;
525                  }
526                  cout << "Loan Amount : " << currentPtr->loanAmount << endl;
527                  cout << "Loan Duration : " << currentPtr->loanDuration << endl;
528                  cout << endl;
529
530                  do {
531                      //Entered Loan details save request
532                      cout << "Do you want to Approve the Loan (Y / N / Q :- Exit from queue) : ";
533                      cin >> save;
534                      if (save == 'Y' || save == 'y' || save == 'N' || save == 'n') {
535                          element = linearCustSearch(cust, nic);
536
537                          //Updating customer details
538                          if (save == 'Y' || save == 'y') {
539                              element->loanType = currentPtr->loanType;
540                              element->loanAmount = currentPtr->loanAmount;
541                              element->loanDuration = currentPtr->loanDuration;
542                              element->customerType = 'L';
543                              cout << "Loan Approved." << endl;
544                          }
545                          else {
546                              element->customerType = 'G';
547                              cout << "Loan Rejected." << endl;
548                          }
549                          deleteLoanDetails(&headLoanPtr, &tailLoanPtr);
550                      }
551                      else if (save == 'Q' || save == 'q') {
552                          clearCMD();
553                          flag = true;
554                          cout << "Exiting from loan queue...." << endl;
555                      }
556                      else {
557                          cout << "Invalid Entry!" << endl;
558                      }
559                  } while (save != 'Q' && save != 'q' && save == 'Y' && save == 'y' && save == 'N' && save == 'n');
560
561                  currentPtr = headLoanPtr;
562              } while (currentPtr!=NULL && (save != 'Q' && save != 'q'));
563      }
564  }
565
```

Figure 33

"Approve / Reject Loan" is a function where Back office approves or reject the Loan requests from the system. First System checks if there're any requests in the Queue. If there's no requests program will prompt a message "There're no loan requests in the queue!".

If there're requests program will go through the queue and prompt Customer NIC, Requested Loan Type, Loan Amount, Loan Duration. Program request user to enter 'Y' or 'N' or 'Q' (Yes or No or Exit from Queue). If the user enters Character except 'Y' or 'N' or 'Q' program will prompt a message "Invalid Entry!".

If the user enters 'Y' program will save the Loan details (Loan Type, Amount, Duration) in the Main Customer record and update Customer Type as 'L' (Loan Customer). The system will prompt a message "Loan Approved." And if there's another Loan request in the queue System will display that Loan request details.

If the user enters 'N' program will not save the Loan details and Customer Type will set to 'G' (General Customer). The system will prompt a message "Loan Rejected." And if there's another Loan request in the queue System will display that Loan request details.

If the user enters 'Q' program will exit from the queue.

View Approve Customers

```
566    void viewAppCustomer(){
567        displayCustomerDetails('L');
568    }
```

Figure 34

"View Approved Customers" is a function where All the Loan Obtained Customers Details can be viewed. In this function it will forward to the "displayCustomerDetails" function with parameter value 'L', So the program will prompt all the Loan Obtained Customers in the System. The System will display following Customer details:

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

View Pending Customers

```
570    void viewPendingCustomer(){
571        displayCustomerDetails('P');
572    }
```

Figure 35

"View Pending Customers" is a function where All the Loan Requested Customers Details can be viewed. In this function it will forward to the "displayCustomerDetails" function with parameter value 'P', So the program will prompt all the Loan Requested Customers in the System. The System will display following Customer details:

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

View General Customers

```
574    void viewGeneralCustomer(){
575        displayCustomerDetails('G');
576    }
```

Figure 36

"View General Customers" is a function where All the Customers except Loan and Pending Customer Details can be viewed. In this function it will forward to the "displayCustomerDetails" function with parameter value 'G', So the program will prompt all the Customers who haven't request or obtained a Loan in the System. The System will display following Customer details:

1. Customer NIC
2. Customer Name

Customer Details will display in a sorted order of NICs.

Display Customer Details

```
578  void displayCustomerDetails(char type) {
579      clearCMD();
580      bool flag = true; //This used to check whether if there's any registered records
581      customerPtr currentPtr = cust;
582      if (type=='G' || type == 'A') {
583          cout << "Customer NIC" << setw(20) << "Customer Name" << endl;
584      }
585      else {
586          cout << "Customer NIC" << setw(20) << "Customer Name" << setw(20) << "Loan Amount" << endl;
587      }
588
589      while (!isEmpty(currentPtr)) {
590          if (type == 'A' || (currentPtr->customerType == 'G' && type == 'G')) {
591              flag = false;
592              cout << setw(12) << currentPtr->NIC << "       " << currentPtr->customerName << endl;
593          }
594          else if (currentPtr->customerType == 'P' && type == 'P') {
595              flag = false;
596              cout << setw(12) << currentPtr->NIC << "       " << currentPtr->customerName << "       " << loanAmount(headLoanPtr, currentPtr->NIC) << endl;
597          }else if(currentPtr->customerType == 'L' && type == 'L'){
598              flag = false;
599              cout << setw(12) << currentPtr->NIC << "       " << currentPtr->customerName << "       " << currentPtr->loanAmount << endl;
600          }
601          currentPtr = currentPtr->nextCustPtr;
602      }
603
604      if (flag) {
605          cout << endl << "No Record Found!" << endl;
606      }
607      cout << endl;
608  }
609
```

Figure 37

"Display Customer Details" is a function where Customer NIC and Name will display using the given Customer Type or All the Registered Customers.

In this function, it will get a char parameter from View All Customers ('A'), View Approve Customers ('L'), View Pending Customers ('P') and View General Customers ('G') functions. Using this char parameter program will display the specific Customers NIC and Name.

Then it will display a header "Customer NIC" and "Customer Name" (Line 583 or 586). If given Customer type is 'L' or 'P' additionally "Loan Amount" will displayin the System. It will go through a list of Customers and display only selected Customer Details (NIC and Name). Line 594 shows the Iteration part.

In this function, there's a variable name called "flag" and its default value is TRUE (Line 580). It used to identify if there's any Customer record displayed. If any Customer record displayed program will set "flag" to FALSE (Line 587 or 591). If no Customer record displayed end of the function it will go through an if statement and display a message "No Record Found!" (Line 598).

Linear Array Search

```
605  int linearArrSearch(int NIC[] ,int searchKey){
606      for (int x = 0; x < noOfCust;x++) {
607          if (NIC[x] == searchKey) {
608              return x;
609          }
610      }
611      return -1;
612  }
```

Figure 38

"Linear Array Search" is a function where search Customer is Stored in the NIC Array (Customer Registered or not). This function will get two parameters. Array and the Search Key (NIC). In this Array go through a loop and check each value with the given search key. If it matches function will return the NIC or else return -1.

Linear Customer Search

```
614  customerPtr linearCustSearch(customerPtr sPtr, int nic) {
615      customerPtr currentPtr = sPtr;
616      while (!isEmpty(currentPtr)) {
617          if (currentPtr->NIC == nic) {
618              return currentPtr;
619          }
620          currentPtr = currentPtr->nextCustPtr;
621      }
622      return NULL;
623  }
```

Figure 39

"Linear Customer Search" is a function where search Customer is Stored in the Data Structure of Link list (Customer Registered or not). This function will get two parameters. Link list starting pointer and NIC. In this Link List go through a loop and check each NIC with the given NIC. If it matches function will return the pointer to that Node or else return NULL.

Loan Amount

```
625  double loanAmount(loanQueuePtr headLoanPtr,int nic) {
626      loanQueuePtr current = headLoanPtr;
627      while (current!=NULL) {
628          if (current->NIC == nic) {
629              return current->loanAmount;
630          }
631          current = current->nextLoanPtr;
632      }
633      return NULL;
634  }
```

Figure 40

"Loan Amount" is a function where the system will get the loan amount from the Data structure of Queue. This function will get two parameters. Queue starting pointer and NIC. The system will go through the Queue and get the specific customer loan request amount.

NIC Array Sort

```
636  void Sort(int NIC[]) {
637      int temp = 0;
638      for (int x = 1; x < noOfCust;x++) {
639          for (int y = 0; y < noOfCust-1;y++) {
640              if (NIC[y]>NIC[y+1]) {
641                  temp = NIC[y];
642                  NIC[y] = NIC[y + 1];
643                  NIC[y + 1] = temp;
644              }
645          }
646      }
647  }
```

Figure 41

"NIC Array Sort" is a function where NIC Array will be sorted in ascending order. In this function, NIC Array comes as a parameter.

Clear CMD

```
649  void clearCMD() {
650      system("cls");
651  }
```

Figure 42

"Clear CMD" is a function where previously display details will clear from the display windows.

Insert Customer Details

```
653  bool insertCustDetails(customerPtr *sPtr,int nic,char name[20],char address[50],int age,char gender,char type) {
654      customerPtr newPtr = NULL;
655      customerPtr currentPtr = NULL;
656      customerPtr previousPtr = NULL;
657
658      //Creating a new CustomerNode
659      newPtr = new CustomerNode;
660
661      if (!isEmpty(newPtr)) {
662          //Stroing values to newly created node
663          newPtr->NIC = nic;
664          strcpy_s(newPtr->customerName, name);
665          strcpy_s(newPtr->customerAddress, address);
666          newPtr->customerAge = age;
667          newPtr->customerGender = gender;
668          newPtr->customerType = type;
669
670          newPtr->nextCustPtr = NULL;
671          currentPtr = *sPtr;
672
673          //Finding where to place the new node
674          while (!isEmpty(currentPtr) && currentPtr->NIC<nic) {
675              previousPtr = currentPtr;
676              currentPtr = currentPtr->nextCustPtr;
677          }
678
679          //Linking newly created node
680          if (isEmpty(previousPtr)) {
681              newPtr->nextCustPtr = *sPtr;
682              *sPtr = newPtr;
683          }
684          else {
685              previousPtr->nextCustPtr = newPtr;
686              newPtr->nextCustPtr = currentPtr;
687          }
688
689          //Enter customer to array custNIC
690          for (int x = 0; x < noOfCust;x++) {
691              if (::custNIC[x]==nicMax) {
692                  ::custNIC[x] = nic;
693                  break;
694              }
695          }
696          return true;
697      }
698      else {
699          cout << "No memory!" << endl;
700          return false;
701      }
702  }
703
```

Figure 43

"Insert Customer Details" is a function where the Customer details storing process happens. In this function, 7 parameters will get from "Register Customer" function. They are Link List Starting pointer with reference and Main Customer Details (NIC, Name, Address, Age, Gender, Type).

All these New Customer details stored to a new Node and then It will place in the specific place by checking the NIC order in the Link list (If there's no Link list then this will become the Link list starting pointer). Also, this new Customer NIC will be stored in the NIC Array and return TRUE.

While creating new Node if the new Node not created successfully it will display an error message "No memory!" and return FALSE.

Delete Customer Details

```
704  int deleteCustomerDetails(customerPtr *sPtr, int nic) {
705      /*
706      1) Return value 0 represents Customer registered in the system and that customer unregistered successfully
707      2) Return value 1 represents Customer registered in the system but Customer has obtained a loan. Because of that customer
708          cannot unregister from the system
709      3) Return value 2 represents Customer registered in the system but Customer has requested a loan. Because of that customer
710          cannot unregister from the system
711      */
712      customerPtr tempPtr = NULL;
713      customerPtr previousPtr = NULL;
714      customerPtr currentPtr = NULL;
715
716      if (nic == (*sPtr)->NIC) {
717          if ((*sPtr)->customerType == 'G') {
718              tempPtr = *sPtr;
719              *sPtr = (*sPtr)->nextCustPtr;
720              delete(tempPtr);
721          }
722          else if ((*sPtr)->customerType == 'L') {
723              return 1;
724          }
725          else {
726              return 2;
727          }
728      }
729      else {
730          previousPtr = *sPtr;
731          currentPtr = (*sPtr)->nextCustPtr;
732
733          while (!isEmpty(currentPtr) && currentPtr->NIC != nic) {
734              previousPtr = currentPtr;
735              currentPtr = currentPtr->nextCustPtr;
736          }
737
738          if (!isEmpty(currentPtr)) {
739              if (currentPtr->customerType == 'G') {
740                  tempPtr = currentPtr;
741                  previousPtr->nextCustPtr = currentPtr->nextCustPtr;
742                  delete(tempPtr);
743              }
744              else if(currentPtr->customerType == 'L'){
745                  return 1;
746              }
747              else {
748                  return 2;
749              }
750          }
751      }
752      for (int x = 0; x < noOfCust; x++) {
753          if (::custNIC[x] == nic) {
754              ::custNIC[x] = nicMax;
755          }
756      }
757      return 0;
758  }
759
```

Figure 44

"Delete Customer Details" is a function where the Customer details removing process happens. In this function, 2 parameters will get from "Un-Register Customer" function. They are Link List Starting pointer with reference and NIC.

Function checks starting pointer Customer NIC with the given NIC. If that's the Customer details we want to remove then it will delete from the Link list and the NIC Array will set to default value. If not the starting Customer then the function will go through a loop and get the specific Customer details and remove from the System.

If that Customer is obtained a Loan then function will return 1. If that Customer is requested a Loan the function will return 2. If the Customer details removed successfully then the function will return 0.

Is Empty (Customer)

```
760    int isEmpty(customerPtr sPtr) {
761        return sPtr == NULL;
762    }
```

Figure 45

"Is Empty" is a function where check the given Customer Data Structure pointer is NULL or Not (Return TRUE or FALSE).

Enter Loan Request Details

```
764  bool enterLoanDetails(loanQueuePtr *headPtr, loanQueuePtr *tailPtr, int nic, char type, double amount, int duration) {
765      loanQueuePtr newPtr = new LoanQueueNode;
766
767      if (!isEmpty(newPtr)) {
768          newPtr->NIC = nic;
769          newPtr->loanType = type;
770          newPtr->loanAmount = amount;
771          newPtr->loanDuration = duration;
772          newPtr->nextLoanPtr = NULL;
773
774          if (isEmpty(*headPtr)) {
775              *headPtr = newPtr;
776          }
777          else {
778              (*tailPtr)->nextLoanPtr = newPtr;
779          }
780          *tailPtr = newPtr;
781          return true;
782      }
783      else {
784          cout << "not inserted. No Memory Available." << endl;
785          return false;
786      }
787  }
788
```

Figure 46

"Enter Loan Request Details" is a function where Customer Loan Details storing process happens. In this function, 6 parameters will get from "Apply for a Loan" function. They are Queue starting pointer and ending pointer with reference and Customer Loan Details (NIC, Loan Type, Amount, Duration).

All these New Customer Loan details stored to a new Node and then It will place at the end of the Queue (If there's no Queue this new Node becomes the start and the end of the Queue) and return TRUE.

While creating new Node if the new Node not created successfully it will display an error message "Not Inserted. No memory Available" and return FALSE.

Delete Loan Request Details

```
789  void deleteLoanDetails(loanQueuePtr *headPtr,loanQueuePtr *tailPtr) {
790      loanQueuePtr tempPtr = NULL;
791
792      tempPtr = *headPtr;
793      *headPtr = (*headPtr)->nextLoanPtr;
794
795      if (isEmpty(*headPtr)) {
796          *tailPtr = NULL;
797      }
798
799      delete(tempPtr);
800  }
```

Figure 47

"Delete Loan Request Details" is a function where Loan request removing process happens. In this function, 2 parameters get from the "Approve / Reject Loan". They are Queue starting and ending pointer with reference.

This function removes the Queue starting pointer and set next pointer as the starting pointer (Remove queue first Node).

Is Empty (Loan)

```
802  int isEmpty(loanQueuePtr currentPtr) {
803      return currentPtr == NULL;
804  }
```

Figure 48

"Is Empty" is a function where check the given Customer Loan Data Structure pointer is NULL or Not (Return TRUE or FALSE).

5. Test Log

Test Case #	Description	Expected Result	Actual Result	Status
1	Execute Program	Program will display a message "Select an Option: "	Program will display a message "Select an Option: "	Pass
2	Main Menu enter Option between 1 to 10	Program display a message and call to a function	Program display a message and call to a function	Pass
3	Main Menu enter Option 0	Program will terminate	Program will terminate	Pass
4	Main Menu enter Option not between 0 to 10	Program display a message and call to a function	Program will display an error message “Selected Option is Invalid!”	Pass
5	Register Customer with NIC (Not a 5 digit number)	Program will check Customer registered in the system	Program will display an error message "Entered NIC is Invalid!"	Pass
6	Register Customer with NIC (5 digit number)	Program will check Customer registered in the system	Program will check Customer registered in the system	Pass
7	Registered Customer with NIC (Customer Already Registered)	Program will request to enter Customer details	Program will display an error message "Customer already Registered."	Pass
8	Registered Customer with NIC (Not Registered Customer)	Program will request to enter Customer details	Program will request to enter Customer details	Pass
9	Register Customer enter Customer Name (More than 20 characters)	Program will request to enter customer address	Program will skip entering customer address	Fail
10	Register Customer Enter Customer Type (Not M or F)	Program will display the entered Customer details	Program will display an error message “Selected Gender is Invalid!” and “Registration Unsuccessful.”	Pass

11	Register Customer Enter Customer Type (M or F)	Program will display the entered Customer details	Program will display the entered Customer details	Pass
12	Register Customer request to save details (Not Y or N)	Program will store customer details and display a message	Program will display an error message “Invalid Entry!”	Pass
13	Register Customer request to save details (entering Y character)	Program will store customer details and display a message “Registered Successfully!”	Program will store customer details and display a message “Registered Successfully!”	Pass
14	Register Customer request to save details (entering N character)	Program will display a message “Registration Unsuccessful.”	Program will display a message “Registration Unsuccessful.”	Pass
15	Un-Register Customer with NIC (Not a 5 digit number)	Program will check Customer registered in the system	Program will display an error message "Entered NIC is Invalid!"	Pass
16	Un-Register Customer with NIC (5 digit number)	Program will check Customer registered in the system	Program will check Customer registered in the system	Pass
17	Un-Register Customer return delete status as 0	Program will display a message "Customer Un-Registered successfully!"	Program will display a message "Customer Un-Registered successfully!"	Pass
18	Un-Register Customer return delete status as 1	Program will display an error message "This Customer has obtained a Loan! Customer Un-Registration Unsuccessful."	Program will display an error message "This Customer has obtained a Loan! Customer Un-Registration Unsuccessful."	Pass
19	Un-Register Customer return delete status as 2	Program will display an error message "This Customer has requested a Loan! Customer Un-Registration Unsuccessful."	Program will display an error message "This Customer has requested a Loan! Customer Un-Registration Unsuccessful."	Pass
20	Update Customer storing new address	Program will display a message "Customer Details Updated Successfully!"	Program will display a message "Customer Details Updated Successfully!"	Pass
21	View Customer if customer is Loan Customer display loan details	Program will display Loan Type, Amount, Duration	Program will display Loan Type, Amount, Duration	Pass

22	Apply Loan requesting Customer is a Loan Customer	Program will display a message "Already there's a Obtained Loan under this Customer. Loan Request Unsuccessful!"	Program will display a message "Already there's a Obtained Loan under this Customer. Loan Request Unsuccessful!"	Pass
23	Apply Loan requesting Customer is a Pending Customer	Program will display a message "Already there's a Pending Loan under this Customer. Loan Request Unsuccessful!"	Program will display a message "Already there's a Pending Loan under this Customer. Loan Request Unsuccessful!"	Pass
24	Apply Loan enter Loan Type (not H, P, G or L)	Program will proceed to enter Loan Amount and Duration	Program will display an error message "Selected Loan Type is Invalid! Loan Request Unsuccessful!"	Pass
25	Apply Loan enter Loan Type (H, P, G or L)	Program will proceed to enter Loan Amount and Duration	Program will proceed to enter Loan Amount and Duration	Pass
26	Apply Loan save loan request (not Y or N)	Program will store the loan request details and change customer type	Program will display an error message "Invalid Entry!"	Pass
27	Apply Loan save loan request (Enter Y)	Program will store the loan request details and change customer type	Program will store the loan request details and change customer type	Pass
28	Apply Loan save loan request (Enter N)	Program will store the loan request details and change customer type	Program will display an error message "Loan Request Unsuccessful!"	Pass
29	Approve / Reject Loan (No loan requests)	Program will display loan requests from the Queue	Program will display a message "There are no loan requests in the queue!"	Pass
30	Approve / Reject Loan (Enter not Y/N or Q)	Program will approve the loan request details	Program will display an error message “Invalid Entry!”	Pass
31	Approve / Reject Loan (Enter Y)	Program will approve the loan request details	Program will approve the loan request details	Pass
32	Approve / Reject Loan (Enter N)	Program will reject the loan request and display a message "Loan Rejected."	Program will reject the loan request and display a message "Loan Rejected."	Pass
33	Approve / Reject Loan (Enter Q)	Program will exit from approve / reject loan and display a message "Exiting from loan queue...."	Program will exit from approve / reject loan and display a message "Exiting from loan queue...."	Pass
34	Display Customer Details If no record displayed	Program will display a message "No Record Found!"	Program will display a message "No Record Found!"	Pass

35	Linear Array Search (Registered Customer)	Program will return the Array Index	Program will return the Array Index	Pass
36	Linear Array Search (Customer Not Registered)	Program will return -1	Program will return -1	Pass
37	Linear Customer Search (Registered Customer)	Program will return the Customer Node pointer	Program will return the Customer Node pointer	Pass
38	Linear Customer Search (Customer Not Registered)	Program will return NULL	Program will return NULL	Pass
39	Loan Amount (Loan request under given NIC number)	Program will return the loan amount	Program will return the loan amount	Pass
40	Loan Amount (No Loan request under given NIC number)	Program will return NULL	Program will return NULL	Pass
41	Insert Customer Details (New node not Initialized)	Program will store the Customer details to the new Node and return TRUE	Program will display an error message “"No memory!" and return FALSE	Pass
42	Insert Customer Details (New node Initialized)	Program will store the Customer details to the new Node and return TRUE	Program will store the Customer details to the new Node and return TRUE	Pass
43	Delete Customer Details (Customer is a Loan Customer)	Program will remove the Customer details from the System and return 0	Program will not remove the Customer details and return 1	Pass
44	Delete Customer Details (Customer is a Pending Customer)	Program will remove the Customer details from the System and return 0	Program will not remove the Customer details and return 2	Pass
45	Delete Customer Details (Customer is a General Customer)	Program will remove the Customer details from the System and return 0	Program will remove the Customer details from the System and return 0	Pass
46	Is Empty (Customer Pointer)	Program will return TRUE if pointer equals to NULL or else return FALSE	Program will return TRUE if pointer equals to NULL or else return FALSE	Pass
47	Enter Loan Details (New Node Not Initialized)	Program will store the Customer details to the new Node and return TRUE	Program will display an error message "not inserted. No Memory Available." and return FALSE	Pass

48	Enter Loan Details (New Node Initialized)	Program will store the Customer details to the new Node and return TRUE	Program will store the Customer details to the new Node and return TRUE	Pass
49	Is Empty (Loan Pointer)	Program will return TRUE if pointer equals to NULL or else return FALSE	Program will return TRUE if pointer equals to NULL or else return FALSE	Pass
50	Main Menu enter option (Not an integer)	Program display a message and call to a function	Program will crash and user needs to terminate the program manually	Fail

6. Conclusion

This System covers Bank of Ceylon Loaning System all the functions according to what they have requested. In this System Customer Register, Un-Register, Update, View Customer Details, Loan request and Loan approve/reject processors will handle according to the scenario. The user may mistype some input so the system will handle those error and system will inform them with an error message.

Whoever going to interact with the system will easily understand what exactly he/she can do in the System with basic computer knowledge. From developers' side program commented so whoever going to do the changes to the system can easily understand what kind of process each function will perform. Also, some processors used many times in the system so program separated into sub-functions to handle those processors.

In this System, we don't handle exceptions. Because of that some mistype words can crash the program. Also, in this System, we're not storing these customer details permanently in the system. So the user cannot close the program and re-run it again.

7. Reference

1. Cplusplus.com. (2017). *Preprocessor directives - C++ Tutorials*. [online] Available at: http://www.cplusplus.com/doc/tutorial/preprocessor/ [Accessed 3 Oct. 2017].

2. Cplusplus.com. (2017). *istream::ignore - C++ Reference*. [online] Available at: http://www.cplusplus.com/reference/istream/istream/ignore/ [Accessed 3 Oct. 2017].

3. Cplusplus.com. (2017). *istream::get - C++ Reference*. [online] Available at: http://www.cplusplus.com/reference/istream/istream/get/ [Accessed 3 Oct. 2017].

4. Cplusplus.com. (2017). *Data structures - C++ Tutorials*. [online] Available at: http://www.cplusplus.com/doc/tutorial/structures/ [Accessed 3 Oct. 2017].

5. Cplusplus.com. (2017). *system - C++ Reference*. [online] Available at: http://www.cplusplus.com/reference/cstdlib/system/ [Accessed 3 Oct. 2017].