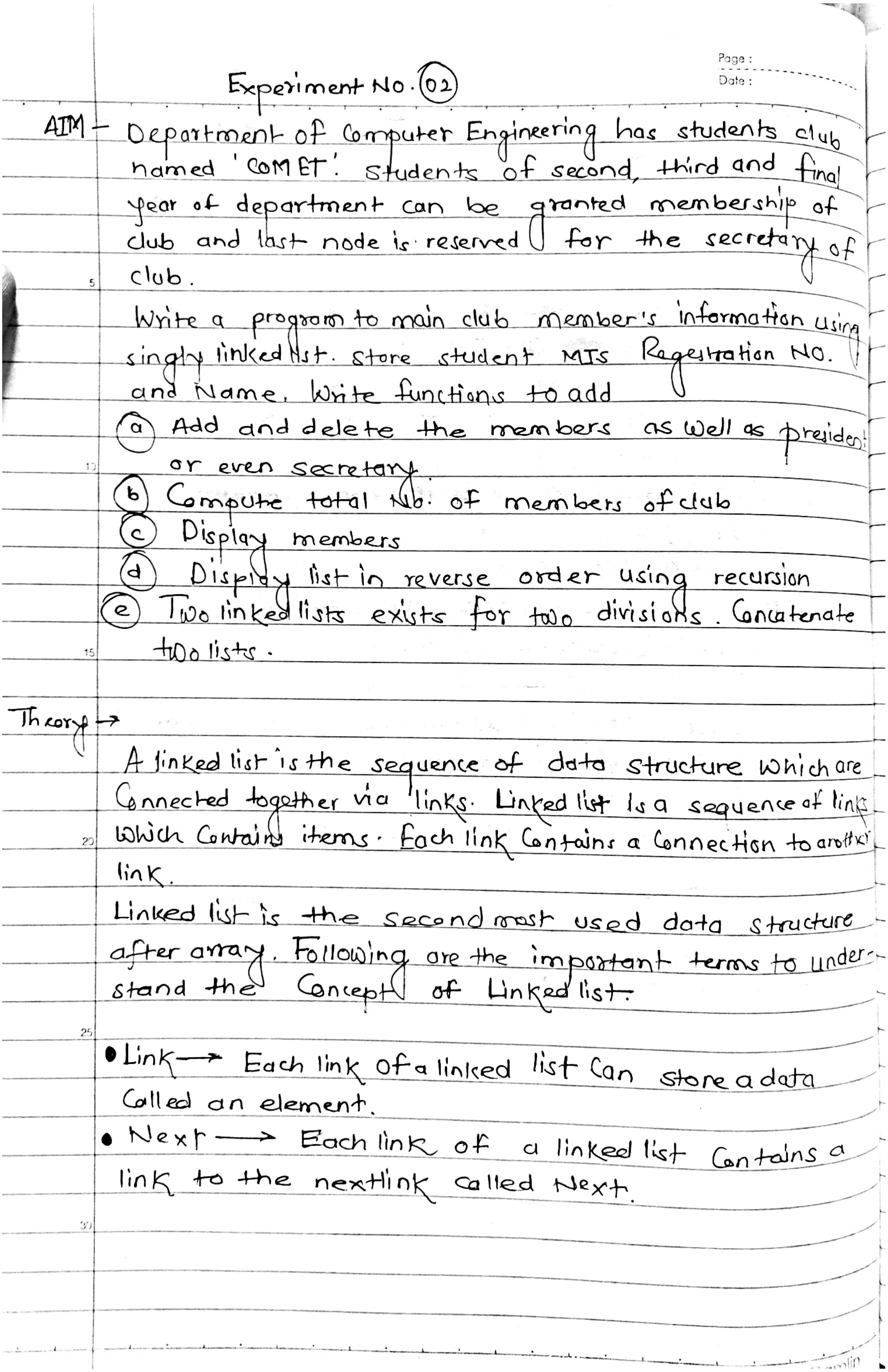
DSA Experiment No.-02



Page :
Date :

Experiment No. 02

AIM – Department of Computer Engineering has students club named 'COMET'. Students of second, third and final year of department can be granted membership of club and last node is reserved for the secretary of club.

Write a program to main club member's information using singly linked list. store student MIS Registration No. and Name. Write functions to add

(a) Add and delete the members as well as president or even secretary.

(b) Compute total No. of members of club

(c) Display members

(d) Display list in reverse order using recursion

(e) Two linked lists exists for two divisions. Concatenate two lists.

Theory →

A linked list is the sequence of data structure which are Connected together via links. Linked list Is a sequence of links which Contains items. Each link Contains a Connection to another link.

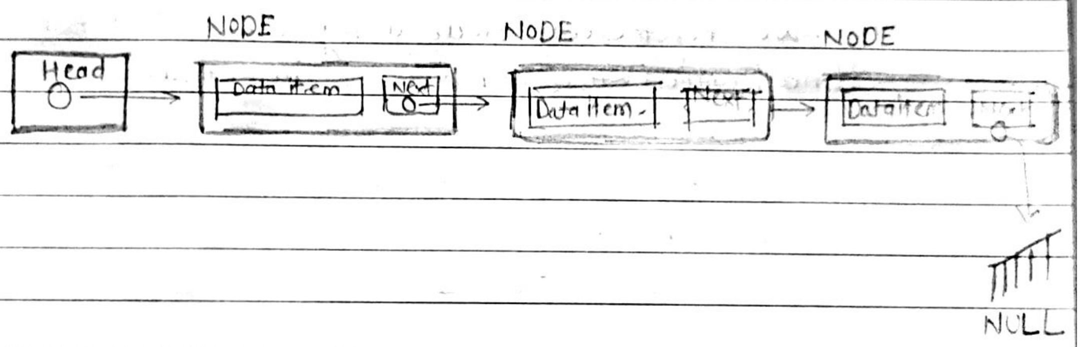
Linked list is the second most used data structure after array. Following are the important terms to understand the Concept of Linked list.

- Link → Each link of a linked list Can store a data Called an element.
- Next → Each link of a linked list Contains a link to the nextlink called Next.

Linked list ⟶ A linked list Contains the connection link to the first link Called first.

Linkedlist Representation ⟶

Linked list Can be visualized as a chain of nodes, Where every node points to the next node.



As a part of above illustration, following are the important points considered, ~~to be~~.
to be .

- Linkedlist Contains a link element Called first.
- Each link Carries a data fields and a link field Called next
- Each link is linked with it's next using the next link.
- Lask link Carries a link as null, mark the end of the list
to

Types of the linked list ⟶

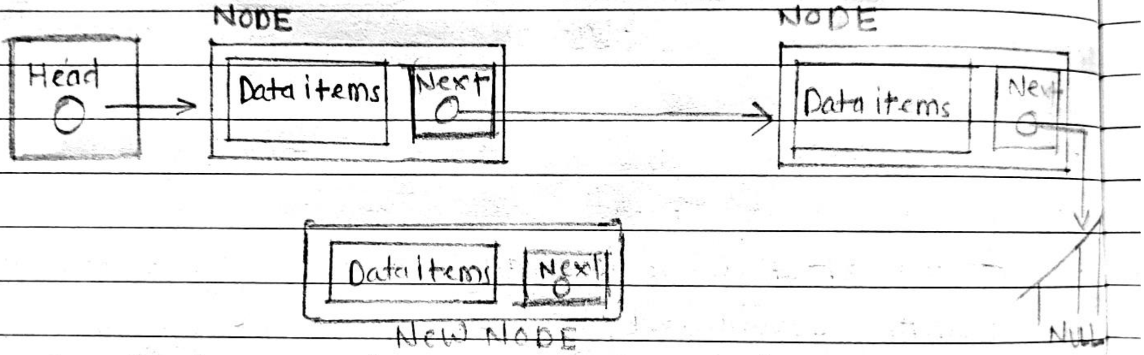
Following are the various types of the linkedlist ⟶

- Simple linkedlist ⟶ Item navigation is forward only.
- Doubly linkedlist ⟶ Items Can be navigated forward and backward.

- Circular linkedlist → Last item Contains link of the first element as next and the first element has a link to the last element as previous.

Basic Operation. →

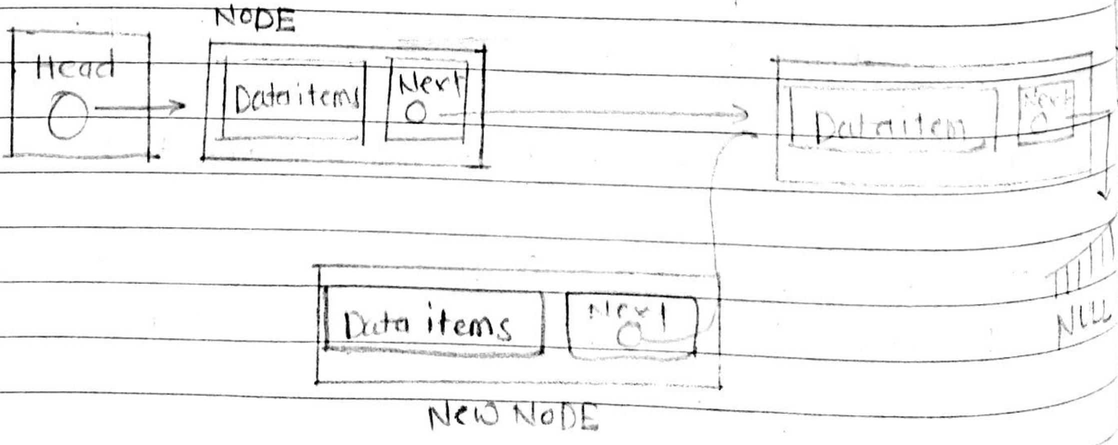
Adding new node to linked list is more than one step activity. We, first create node using same structure and find location where it has to be inserted.



Imagine that we are inserting a node B (Newnode) between A(Leftnode) and C (Rightnode). Then point B.next to C -

NewNode.next → RightNode;

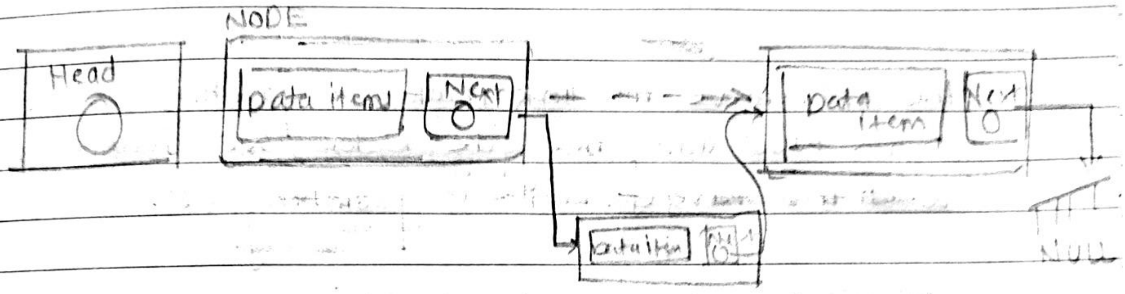
It should look like this →



Page :
Date :

Now, the next node at the left should point to the new node.

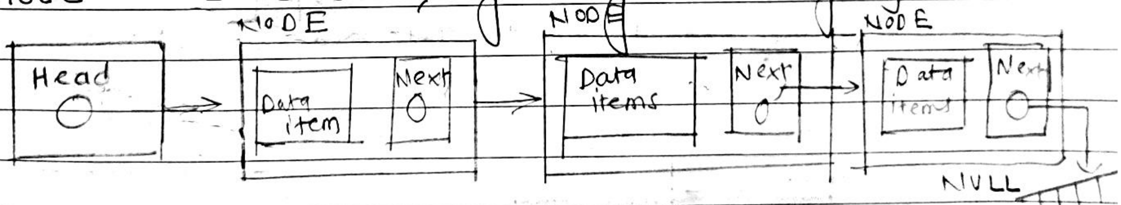
LeftNode.next ⟶ NewNode;



This will put the new node in the middle of two. The new list should look like this ⟶

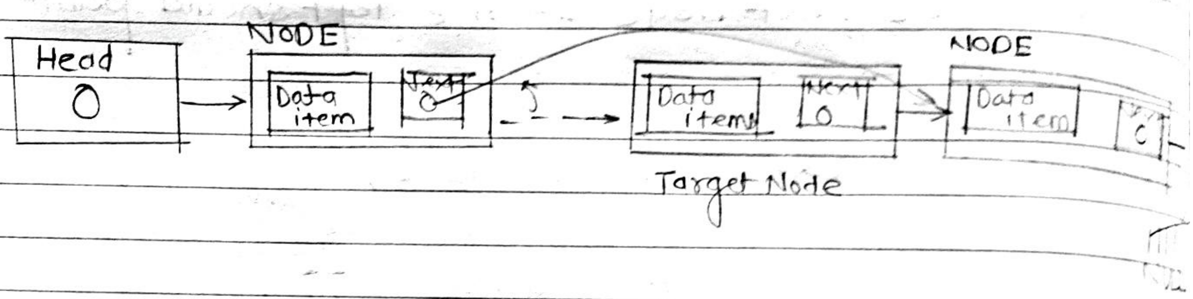
Deletion operation ⟶

Deletion is also a more than one step process. We shall learn with pictorial representation. First locate the target node to be removed, by using searching algorithms.



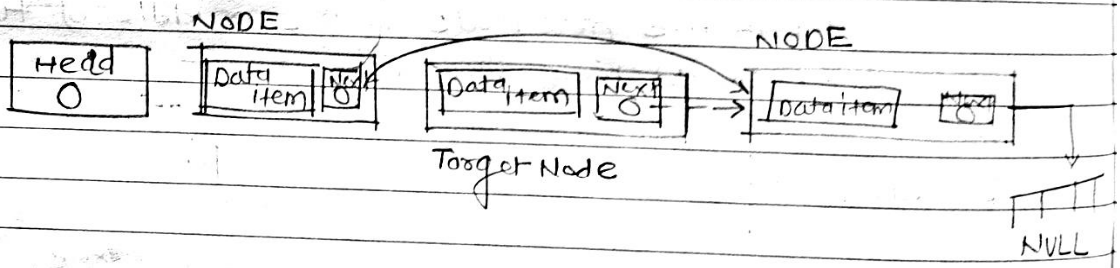
The left (previous) node of the target node now should point to the next node of the target node.

LeftNode.next ⟶ TargetNode.next;

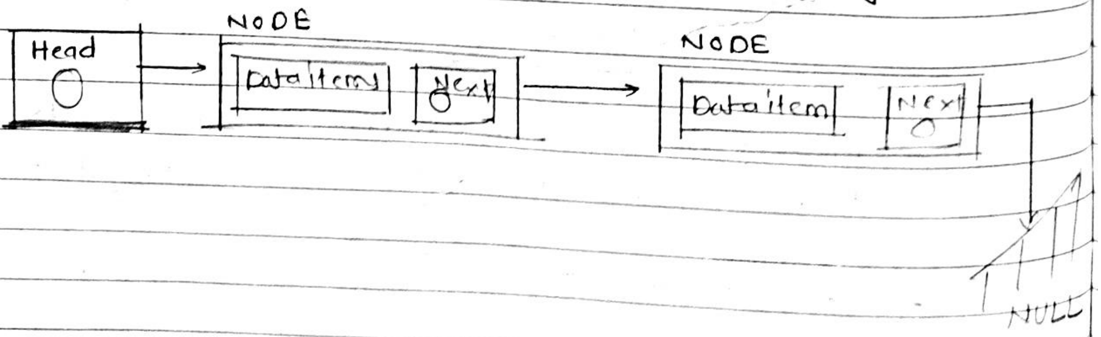


This will remove the link that was pointing to the target node. Now using the code below, we can remove what the target. node is pointing at.

```
TargetNode.next ——> NULL;
```

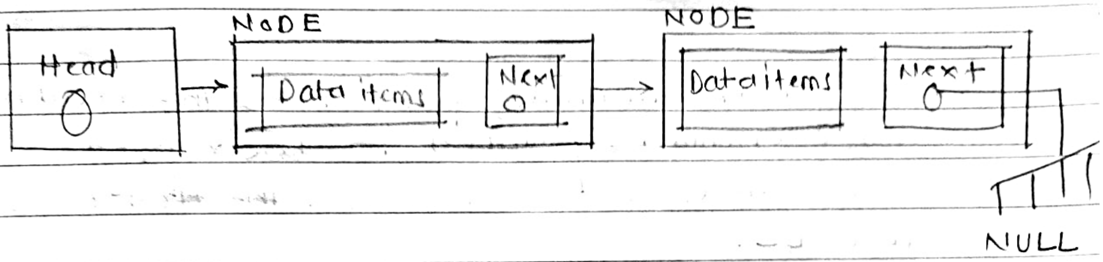


We need to use the deleted node. We con keep that in memory otherwise, we can simply deallocate memory and Wipe off the target node Completely.

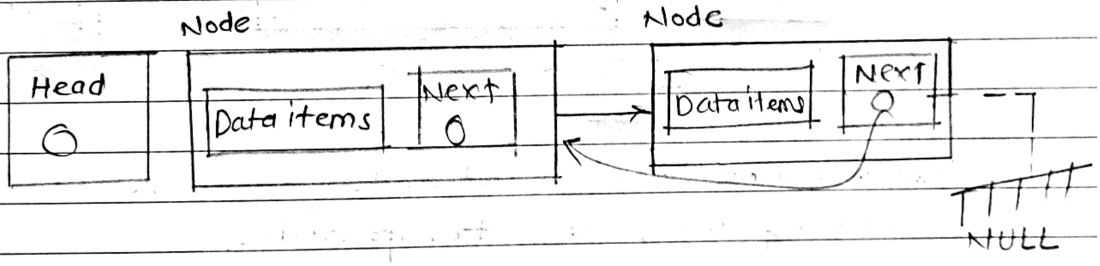


Reverse Operation →

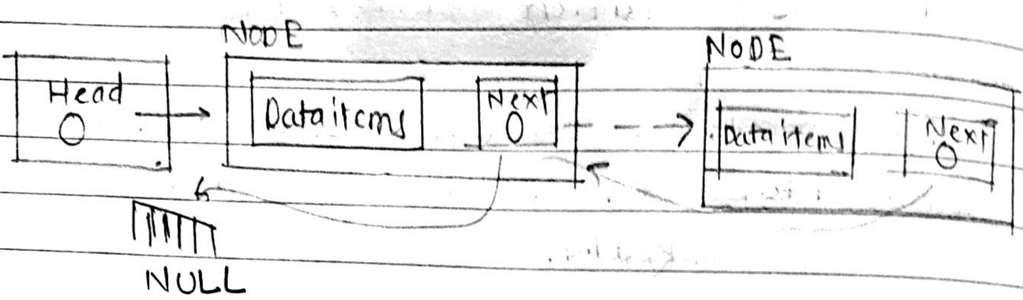
This operation is through one. We need to make the last node to be pointed by the head node and reverse the whole linked list.



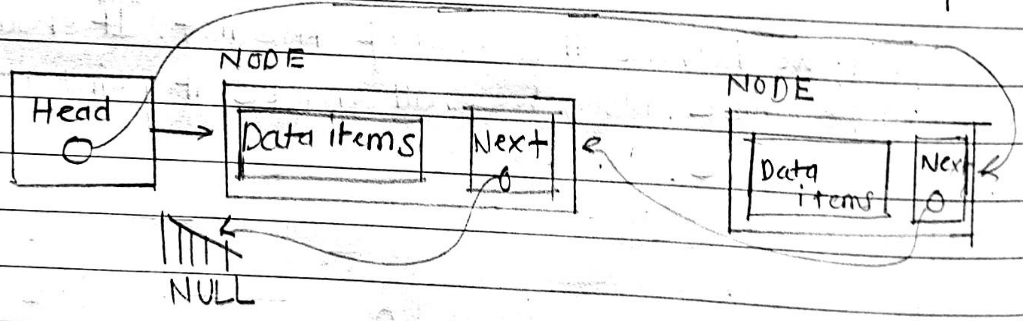
First, We traverse the end of the list. It should be pointing to NULL. Now, we shall make it point to the previous node.



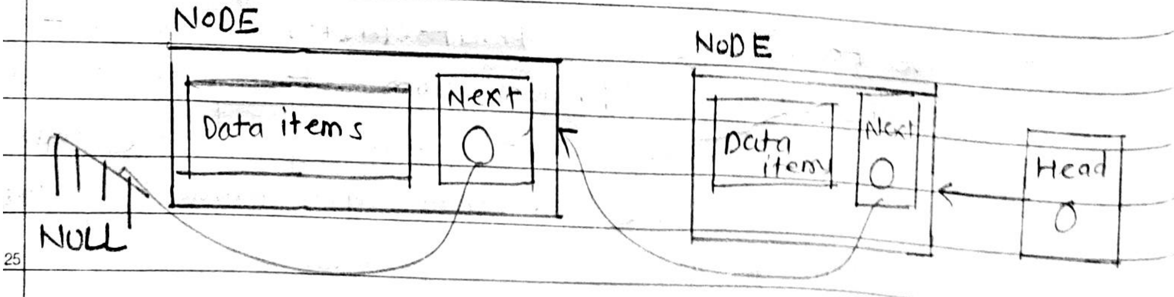
We have to make sure that last node is not the last node. So we'll have some temp node, which looks like the head node pointing to the last node. Now, we shall make all left side Nodes point to their previous nodes one by one.



Except the (first node) pointed by the head node, all nodes should be pointed to their predecessor making them their new successor. The first node will point to NULL.



We'll make the head node point to the new first node by using the temp node.



Thus, the linked list is now reversed.

Program code:

```
#include <iostream>
#include <string>
using namespace std;

class list;

class node
{
    int MIS;
    string name;
    node *next;

public:
    node(int x, string nm)
    {
        MIS = x;
        next = NULL;
        name = nm;
    }
    friend class list;
};
class list
{
    node *start;

public:
    list()
    {
        start = NULL;
    }
    void create();
    void display();
    void insertAtBeginning();
    void insertAtEnd();
    void insertAfter();
    void deleteAtFirst();
    void deleteByValue();
    void deleteAtEnd();
    int computeTotal();
    void sortList();
    void concatList(list &q1);
    void displayRev(node *t);
    bool reverseDisplay() //function is only for passing start as
argument to recursive function
    {
        if (start == NULL)
            return false;
        node *temp = start;
        displayRev(temp);
        //cout<<"(President)";
        return true;
    }
};
void list::displayRev(node *t)
{
    if (t == NULL)
        return;
    else
    {
        displayRev(t->next);
        cout << "\nMIS NO:" << t->MIS << " Name: " << t->name;
    }
```

```
}
void list::create()
{
    int no;
    string nam;
    if (start == NULL)
    {
        cout << "Enter MIS number: ";
        cin >> no;
        cout << "Enter name: ";
        cin >> nam;
        cout << nam;
        start = new node(no, nam);
        cout << "\n====== List Created =====";
    }
    else
    {
        cout << "\nList is already created.";
    }
}
void list::display()
```

```
{
    node *t;
    t = start;
    if (start == NULL)
        cout << "\nList is Empty";
    else
    {
        cout << "\n====== List: ======\n";
        while (t != NULL)
        {
            cout << t->MIS << "  " << t->name << " \n";
            t = t->next;
        }
        //cout<<t->MIS<<"  "<<t->name<<" \n";
    }
}
void list::insertAtBeginning()
{
    int no;
    string nam;
    node *temp;
    if (start == NULL)
    {
        create();
    }
    else
    {
        cout << "\nEnter MIS number: ";
        cin >> no;
        cout << "Enter name: ";
        cin >> nam;
        //cout<<nam;
        temp = new node(no, nam);
        temp->next = start;
        start = temp;
        ;
        cout << "Inserted  " << temp->name << " at the
beginning.";
    }
}
void list::insertAtEnd()
{
    int no;
    string nam;
    node *t;
    if (start == NULL)
        create();
    else
    {
        cout << "\nEnter MIS number: ";
        cin >> no;
        cout << "Enter name: ";
        cin >> nam;
        t = start;
        while (t->next != NULL)
            t = t->next;

        node *p = new node(no, nam);
        t->next = p;
    }
}
void list::insertAfter()
{
    int prev_no;
```

```
cout << "\nENter MIS No. after do you want insert:";
cin >> prev_no;
node *t;
t = start;
string nam;
int flag = 0, no;
while (t != NULL)
{
    if (t->MIS == prev_no)
    {
        flag = 1;
        break;
    }
    t = t->next;
}
if (flag == 1)
{
    node *p;
    cout << "\nEnter MIS number: ";
    cin >> no;
    cout << "Enter name: ";
```

```
        cin >> nam;
        p = new node(no, nam);
        p->next = t->next;
        t->next = p;
    }
    else
    {
        cout << "\n"
             << prev_no << " is not in list.";
    }
}

void list::deleteAtFirst()
{
    node *t;
    if (start == NULL)
        cout << "\nClub is Empty..";
    else
    {
        t = start;
        start = start->next;
    t->next = NULL; //Not necessary
    delete t;
    cout << "\nPresident deleted..";
  }
}

void list::deleteByValue()
{
  int no, flag = 0;
  node *t, *prev;
  if (start == NULL)
    cout << "\nList/Club is empty;";
  else
  {
    cout << "\nEnter MIS no. of member to be deleted: ";
    cin >> no;
    t = start->next; //t=start if we have to delete precident also.. start->next is first member
    while (t->next != NULL)
    {
      if (t->MIS == no)
      {
```

```
                flag = 1;
                break;
            }
            prev = t;
            t = t->next;
        }
        if (flag == 1)
        {
            prev->next = t->next;
            t->next = NULL;
            delete t;
            cout << "\nMember with MIS no: " << no << " is deleted.";
        }
        else
            cout << "\nMember not found in List./president or secretary cannot be deleted.";
    }
}
void list::deleteAtEnd()
{
    node *t, *prev;
    t = start;
    if (start == NULL)
        cout << "\nClub is Empty..";
    else
    {
        while (t->next != NULL)
        {
            prev = t;
            t = t->next;
        }
        prev->next = NULL;
        delete t;
        cout << "\nSecretary Deleted.";
    }
}
int list::computeTotal()
{
    node *t;
    int count = 0;
    t = start;
    if (start == NULL)
    {
        cout << "\nList is empty.";
        return 0;
    }
    while (t != NULL)
    {
        count++;
        t = t->next;
    }

    return count;
}

void list::sortList()
{
    node *i, *j, *last = NULL;
    int tMIS;
    string tname;

    if (start == NULL)
    {
```

```
    cout << "\nList is empty.";
    return;
}
for (i = start; i->next != NULL; i = i->next)
{
    for (j = start; j->next != last; j = j->next)
    {
        if ((j->MIS) > (j->next->MIS))
        {
            tMIS = j->MIS;
            tname = j->name;
            j->MIS = j->next->MIS;
            j->name = j->next->name;

            j->next->MIS = tMIS;
            j->next->name = tname;
        }
    }
}
cout << "\n List is sorted.";
display();
```

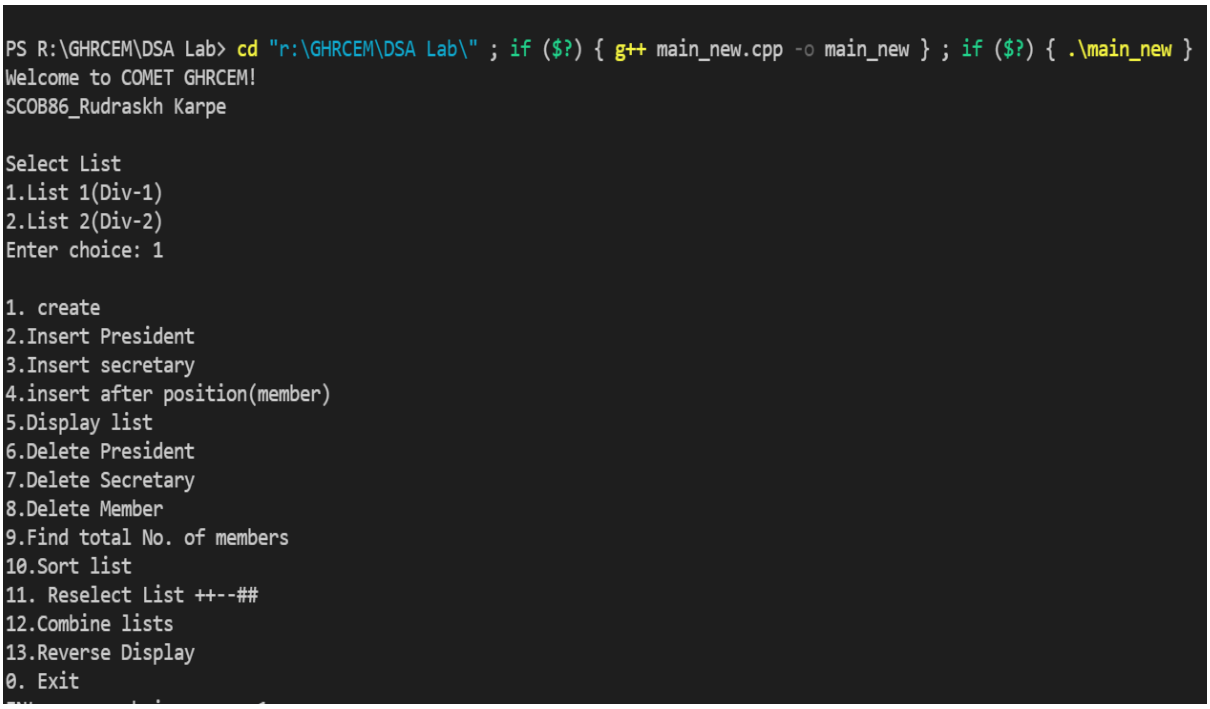
```
}
void list::concatList(list &q1)
{
    node *t, *p;
    t = q1.start;
    if (t == NULL)
    {
        cout << "\nList 2 is empty";
        return;
    }
    p = start; //first list
    while (p->next != NULL)
    {
        p = p->next;
    }
    p->next = t;
    q1.start = NULL; //second list is set to  null
    cout << "\nAfter concatenationlist";
    display();
}
int main()
```

```
{
    list *l;
    int choice, selectList;
    list l1, l2;
    l = &l1;
X:
    cout << "Welcome to COMET GHRCEM!" << endl;
    cout << "SCOB86_Rudraskh Karpe" << endl;
    cout << "\nSelect List\n1.List 1(Div-1)\n2.List 2(Div-2)\nEnter choice: ";
    cin >> selectList;

    if (selectList == 1)
    {
        l = &l1;
    }
    else if (selectList == 2)
    {
        l = &l2;
    }
    else
    {
        cout << "\nWrong list Number.";
        goto X;
    }
    do
    {
        cout << "\n1. create\n2.Insert President\n3.Insert
secretary\n4.insert after position(member)\n5.Display list"
            << "\n6.Delete President\n7.Delete
Secretary\n8.Delete Member\n9.Find total No. of
members\n10.Sort list\n11. Reselect List ++--##"
            << "\n12.Combine lists\n13.Reverse Display\n0.
Exit\nENter your choice:\t";
        cin >> choice;

        switch (choice)
        {
        case 1:
            l->create();
            break;
        case 2:
            l->insertAtBeginning();
            break;
        case 3:
            l->insertAtEnd();
            break;
        case 4:
            l->insertAfter();
            break;
        case 5:
            l->display();
            break;
        case 6:
            l->deleteAtFirst();
            break;
        case 7:
            l->deleteAtEnd();
            break;
        case 8:
            l->deleteByValue();
            break;
        case 9:
            cout << "\nTotal members(including President &
Secretary): " << l->computeTotal();
            break;
```

```
        case 10:
            l->sortList();
            break;
        case 11:
            goto X;
            break;
        case 12:
            l1.concatList(l2);
            break;
        case 13:
            l->reverseDisplay();
            break;
        deafult:
            cout << "Wrong choice";
        }
    } while (choice != 0);
    cout << "\n========== GOOD BYE
====================\n";

    return 0;
}
```

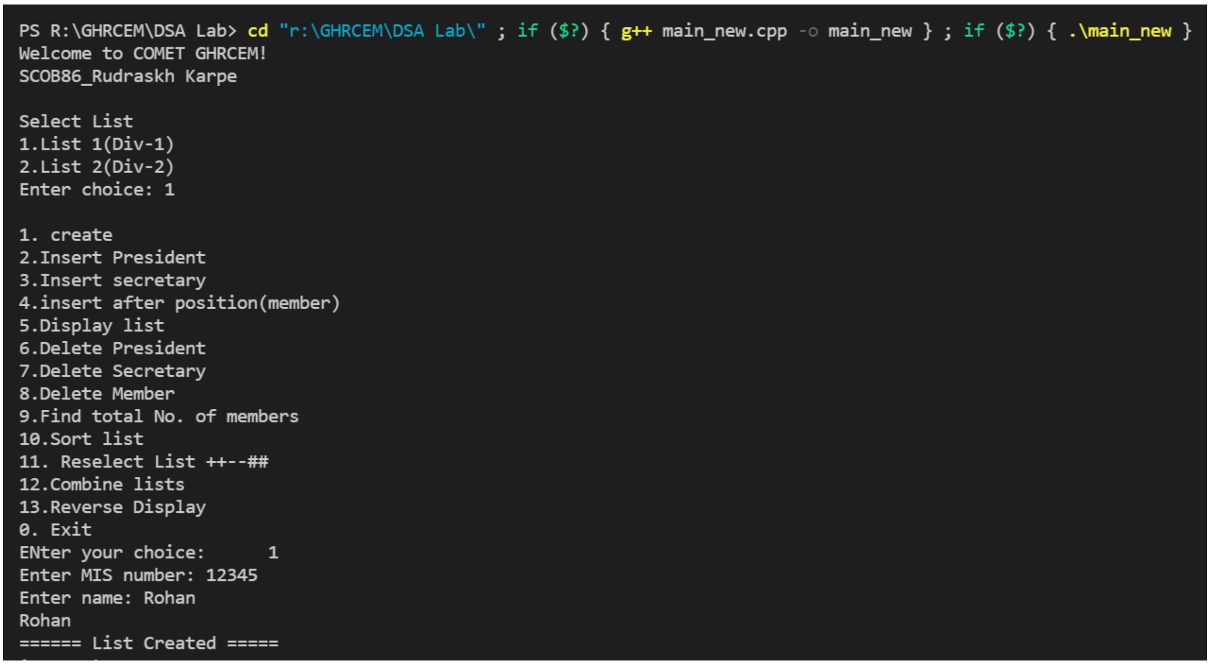
Output:

Interface of the program



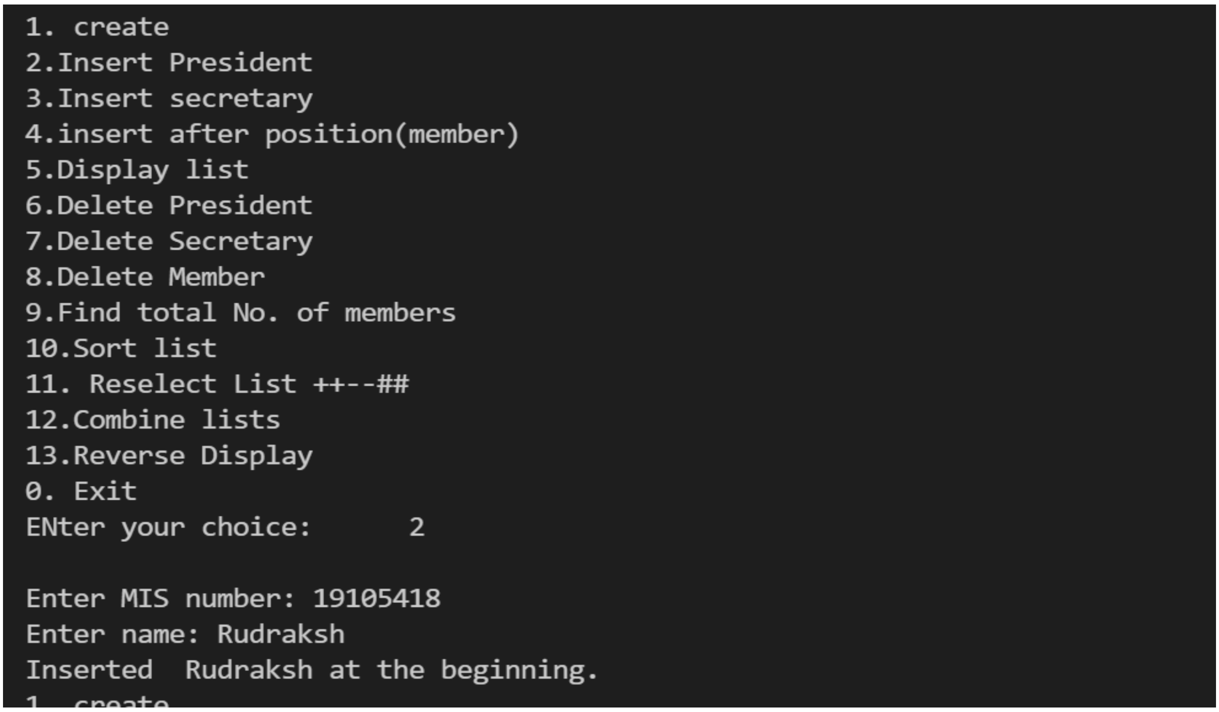
```
PS R:\GHRCEM\DSA Lab> cd "r:\GHRCEM\DSA Lab\" ; if ($?) { g++ main_new.cpp -o main_new } ; if ($?) { .\main_new }
Welcome to COMET GHRCEM!
SCOB86_Rudraskh Karpe

Select List
1.List 1(Div-1)
2.List 2(Div-2)
Enter choice: 1

1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
```

Adding a member



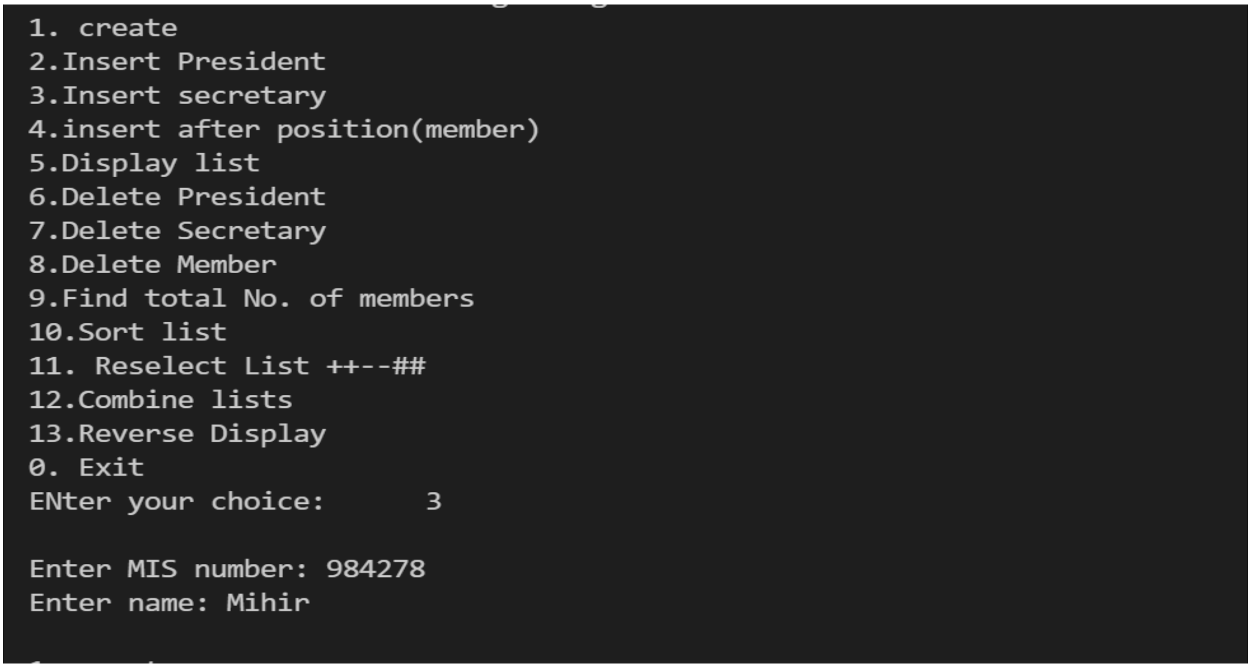
```
PS R:\GHRCEM\DSA Lab> cd "r:\GHRCEM\DSA Lab\" ; if ($?) { g++ main_new.cpp -o main_new } ; if ($?) { .\main_new }
Welcome to COMET GHRCEM!
SCOB86_Rudraskh Karpe

Select List
1.List 1(Div-1)
2.List 2(Div-2)
Enter choice: 1

1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      1
Enter MIS number: 12345
Enter name: Rohan
Rohan
====== List Created =====
```

Adding President



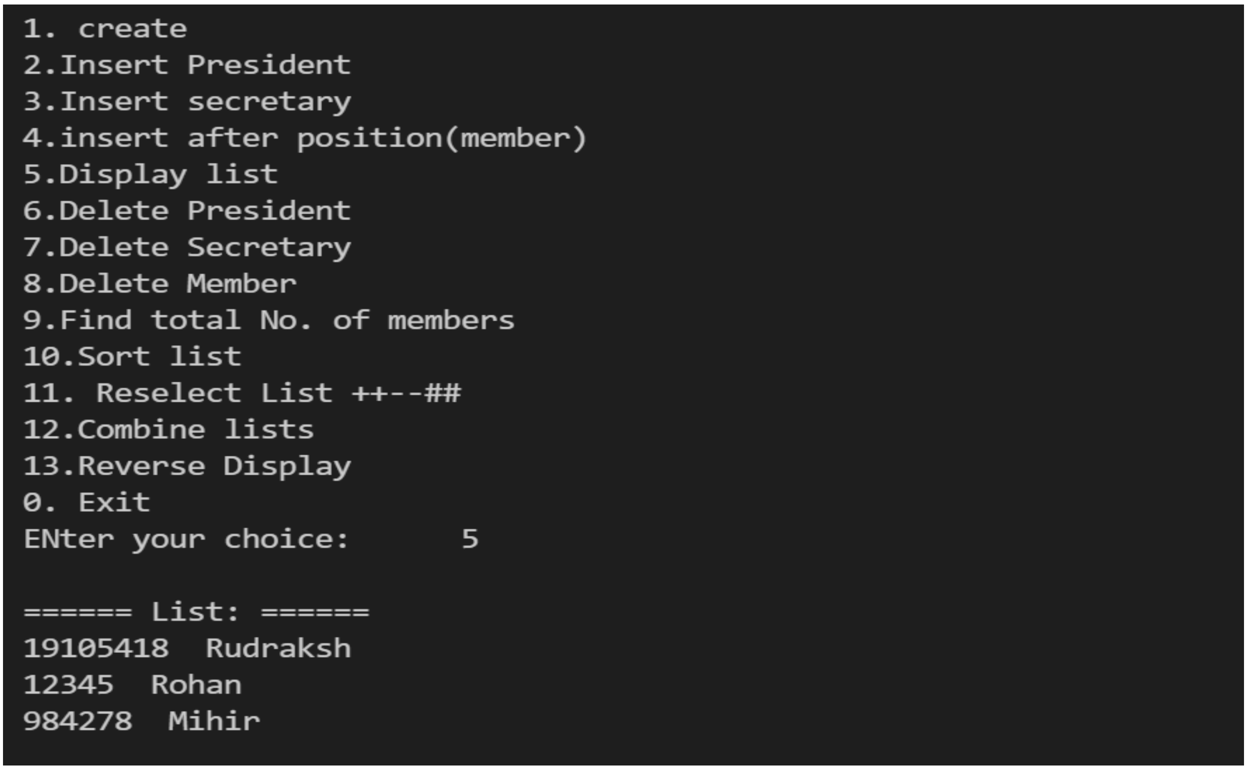
```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      2

Enter MIS number: 19105418
Enter name: Rudraksh
Inserted  Rudraksh at the beginning.
1. create
```

Adding Secretary



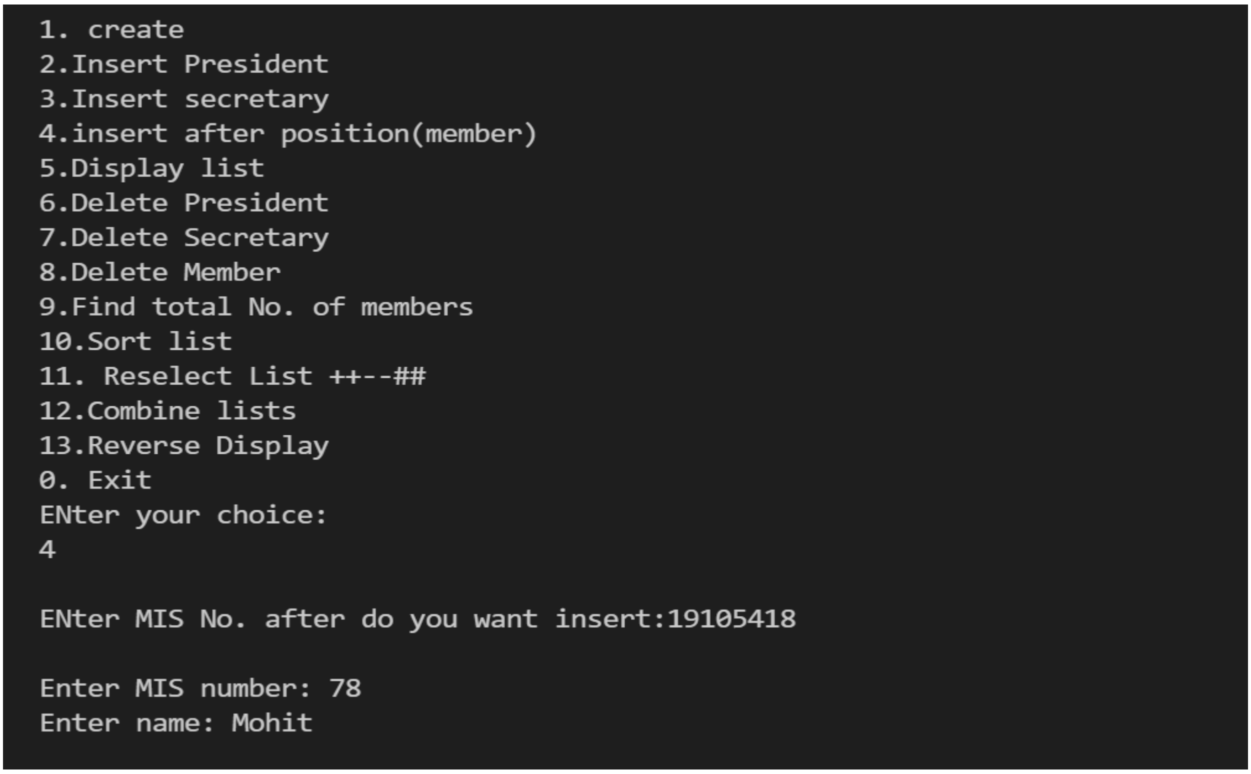
```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      3

Enter MIS number: 984278
Enter name: Mihir
```

Display List

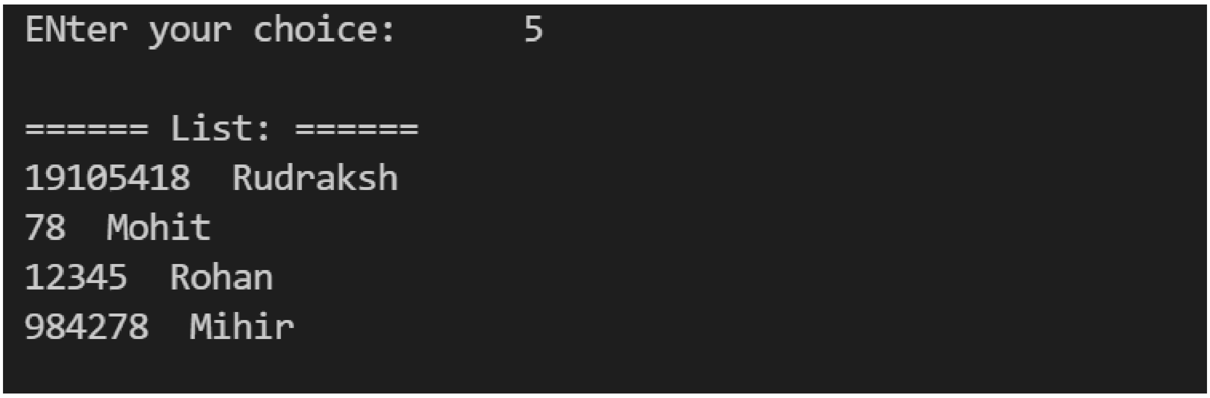


```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      5

====== List: ======
19105418  Rudraksh
12345  Rohan
984278  Mihir
```

Inserting after position member

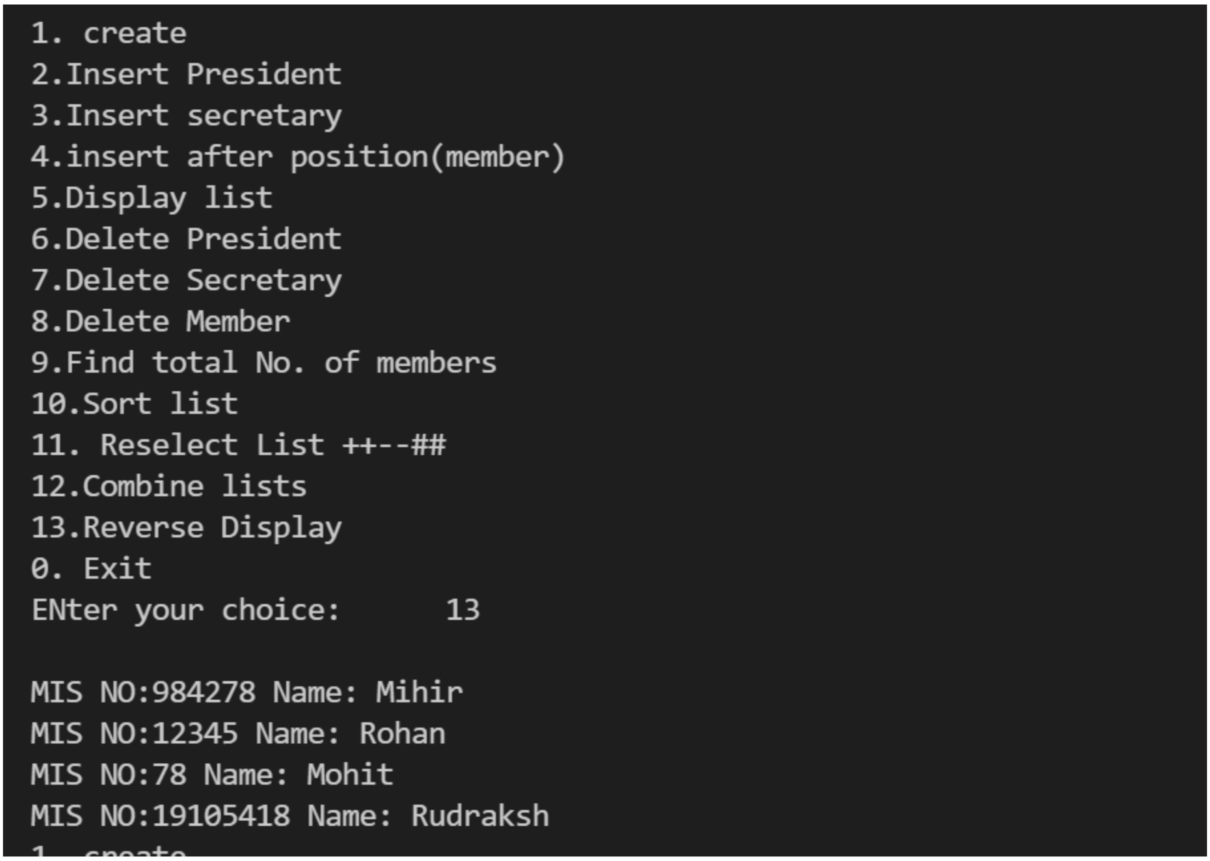


```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:
4

ENter MIS No. after do you want insert:19105418

Enter MIS number: 78
Enter name: Mohit
```



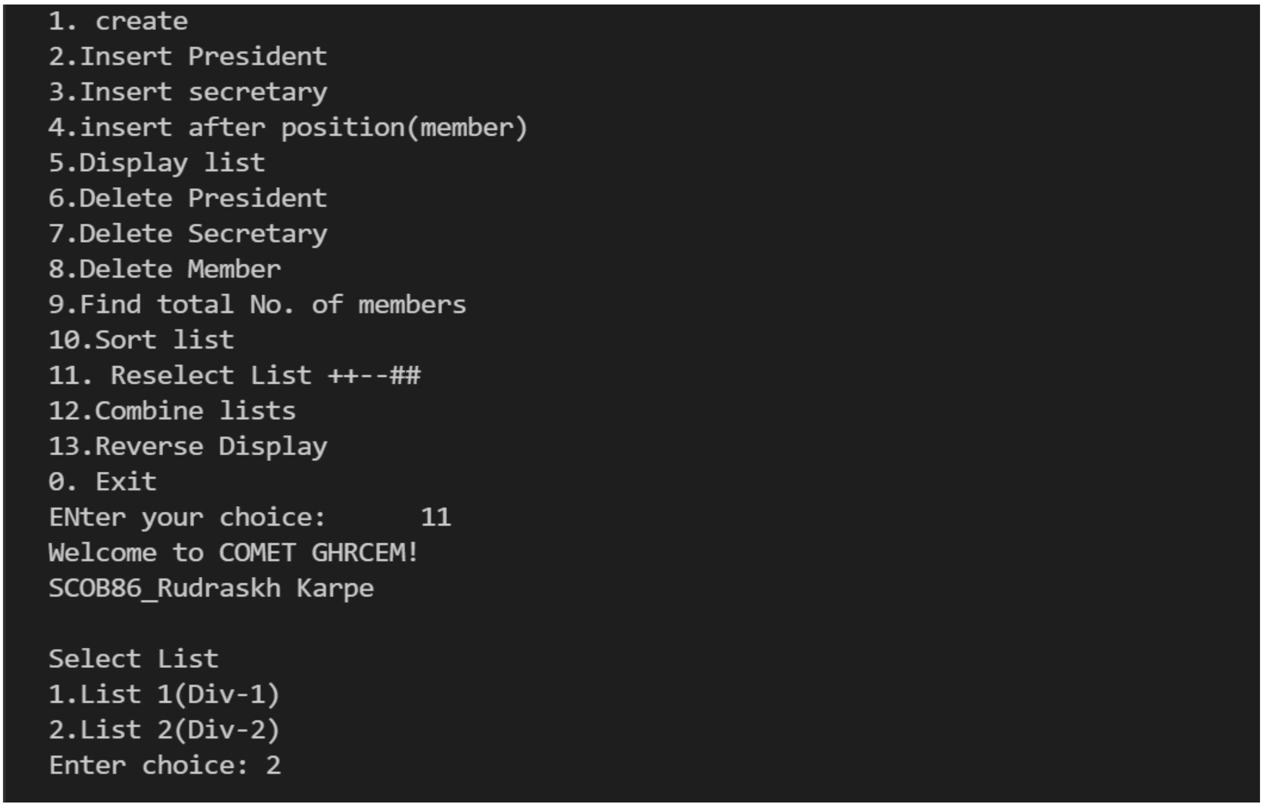
```
ENter your choice:      5

====== List: ======
19105418  Rudraksh
78  Mohit
12345  Rohan
984278  Mihir
```

Reversing the List



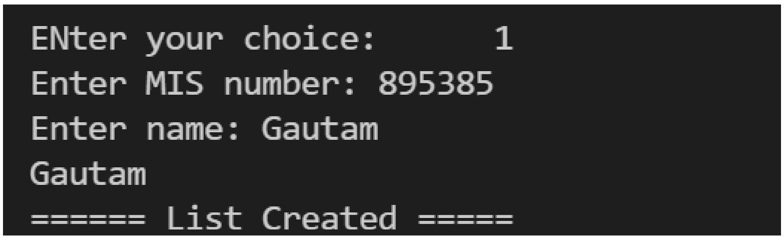
```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      13

MIS NO:984278 Name: Mihir
MIS NO:12345 Name: Rohan
MIS NO:78 Name: Mohit
MIS NO:19105418 Name: Rudraksh
1. create
```

Switching between the List 1 and List 2



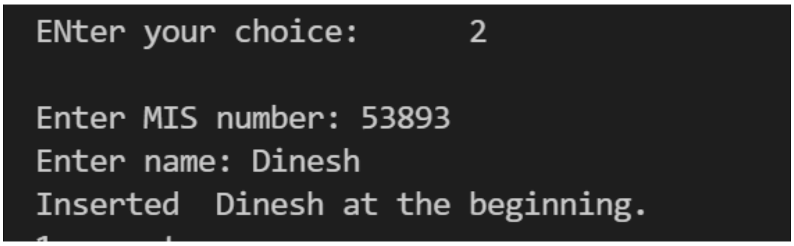
```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      11
Welcome to COMET GHRCEM!
SCOB86_Rudraskh Karpe

Select List
1.List 1(Div-1)
2.List 2(Div-2)
Enter choice: 2
```

Adding Member

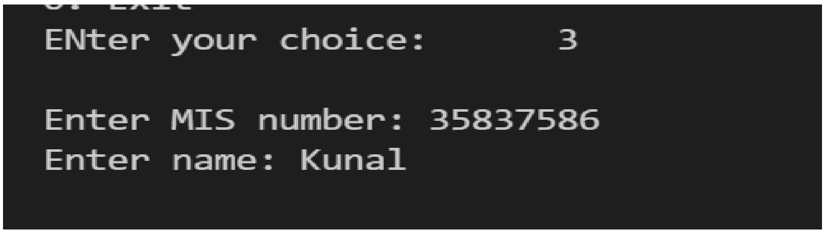


```
ENter your choice:      1
Enter MIS number: 895385
Enter name: Gautam
Gautam
====== List Created =====
```

Adding President

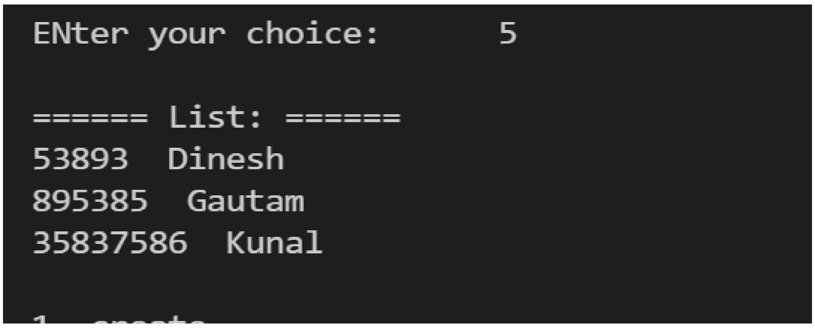


```
ENter your choice:      2

Enter MIS number: 53893
Enter name: Dinesh
Inserted  Dinesh at the beginning.
```



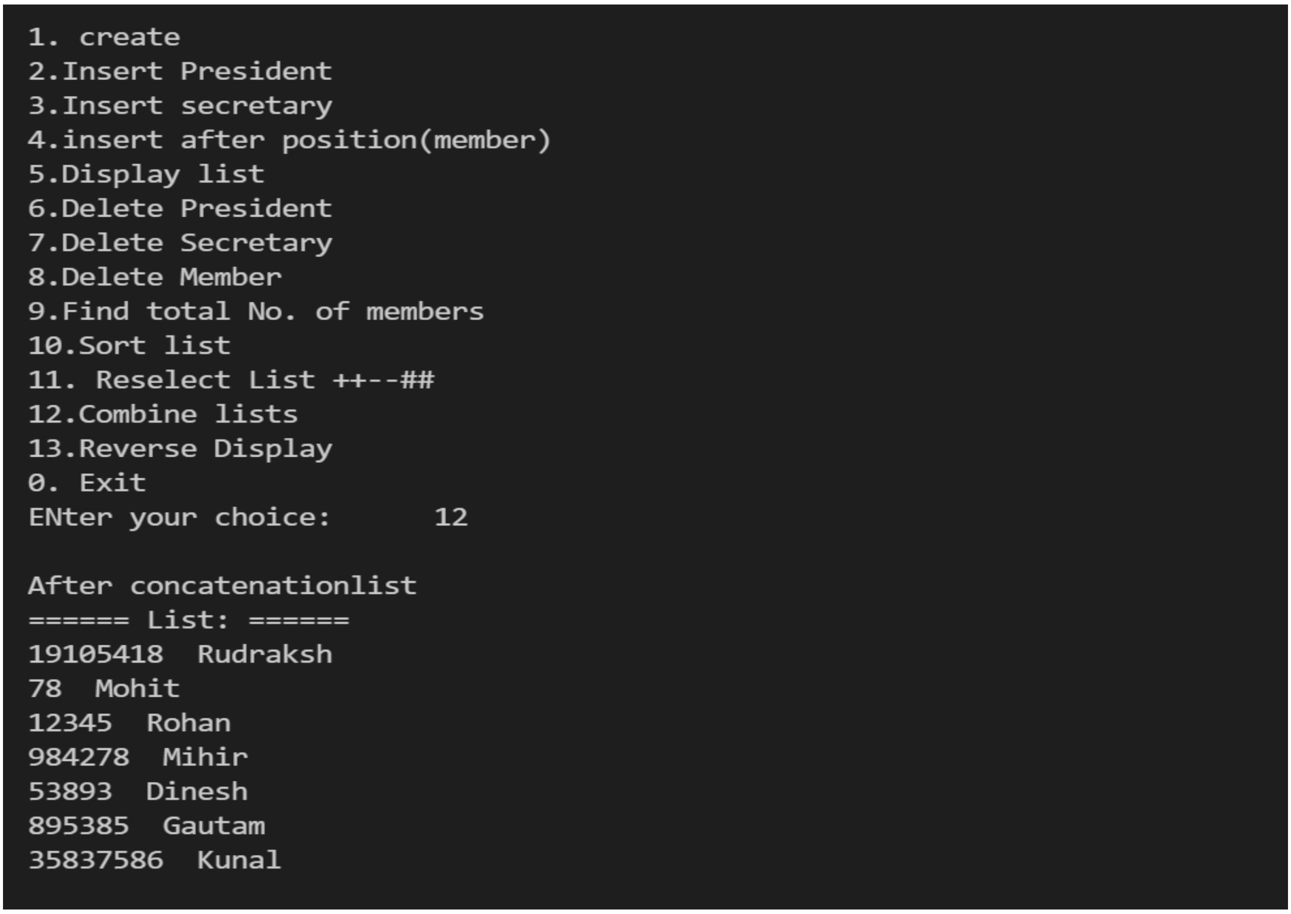
```
0. Exit
ENter your choice:      3

Enter MIS number: 35837586
Enter name: Kunal
```

Displaying List 2



```
ENter your choice:      5

====== List: ======
53893   Dinesh
895385   Gautam
35837586   Kunal

1. create
```

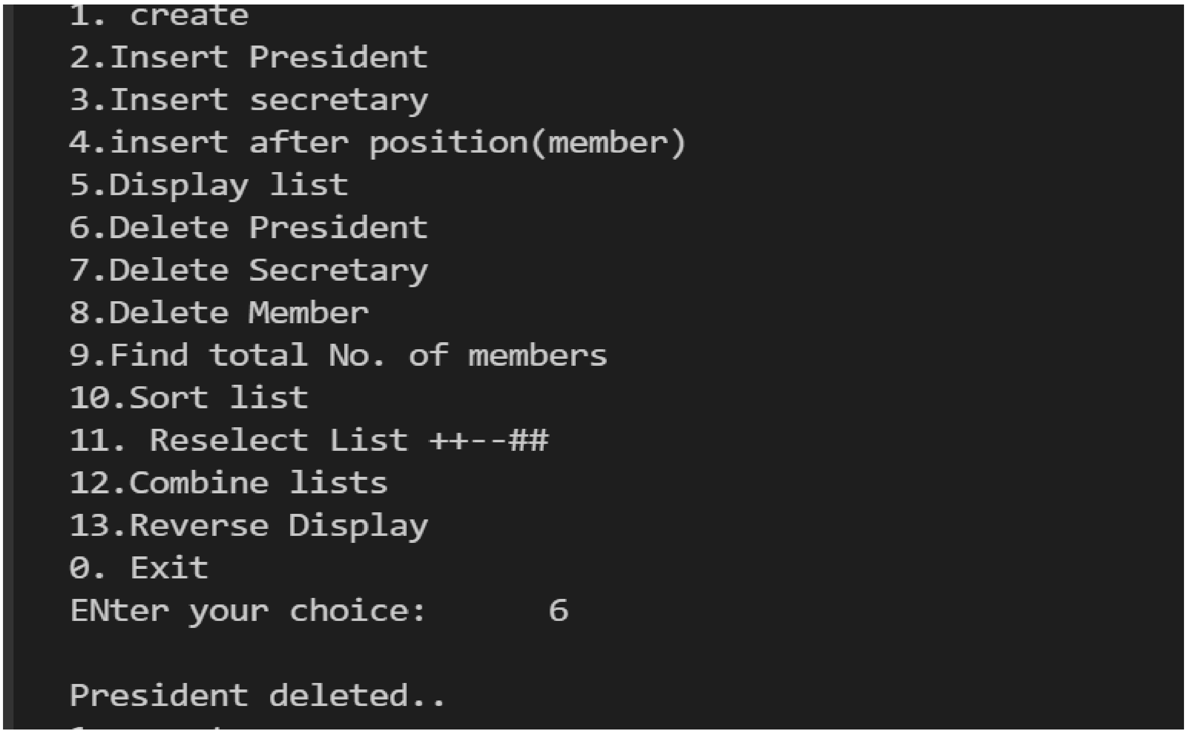
Concatenating List 1 and List 2



```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      12

After concatenationlist
====== List: ======
19105418   Rudraksh
78   Mohit
12345   Rohan
984278   Mihir
53893   Dinesh
895385   Gautam
35837586   Kunal
```

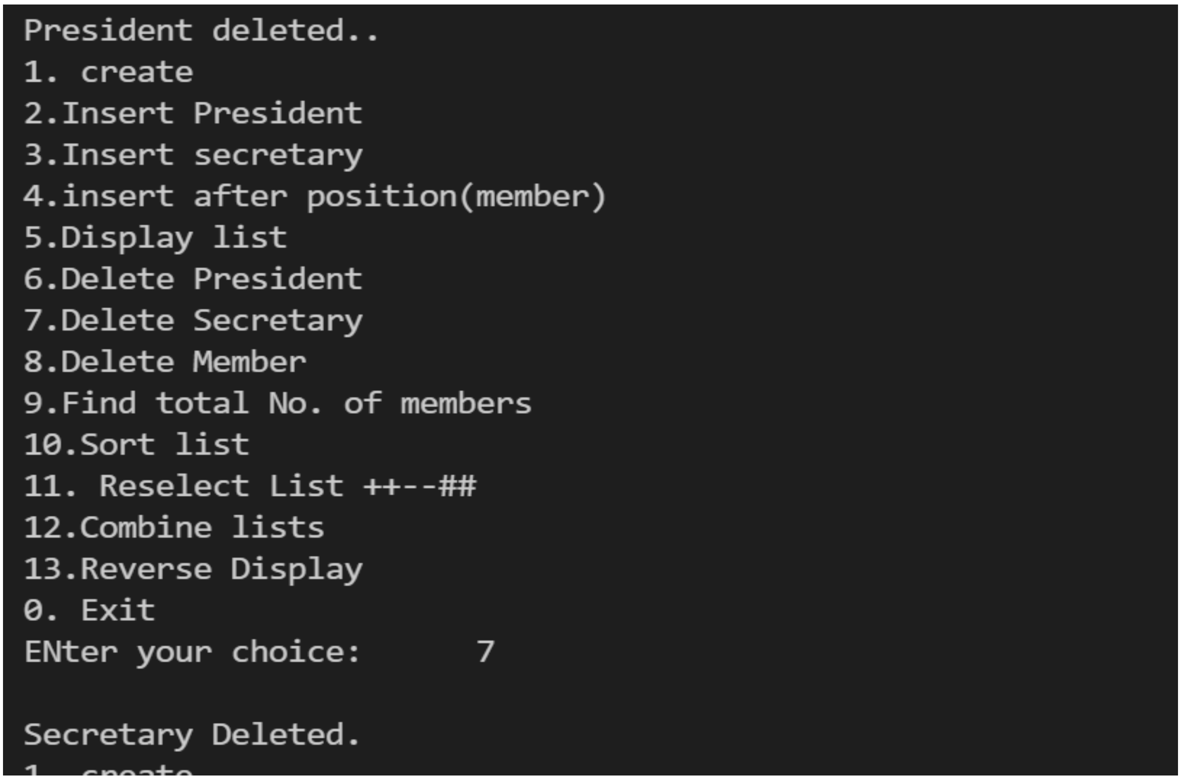
rudraksh.karpe.cs@ghrcem.raisoni.net

Deleting the President



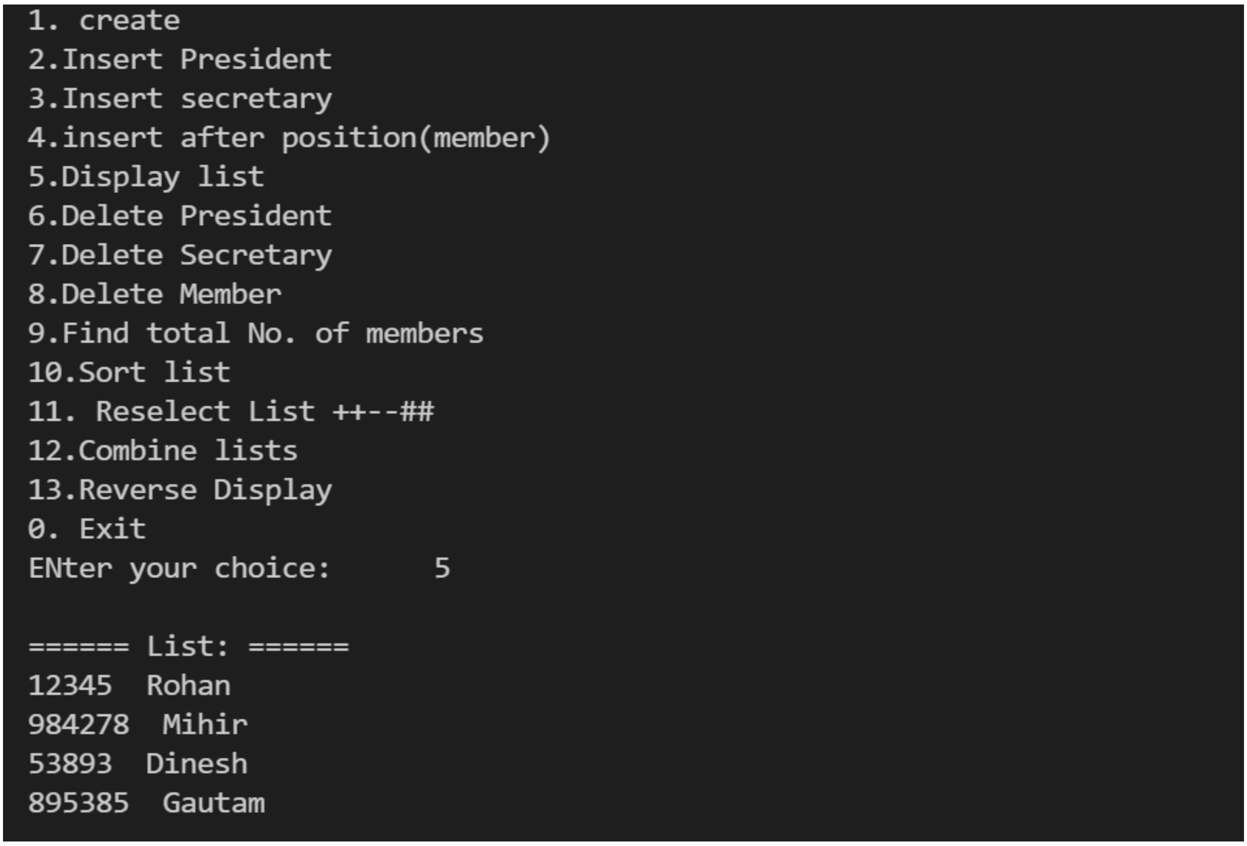
```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      6

President deleted..
```

Deleting Secretary



```
President deleted..
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      7

Secretary Deleted.
```

List after Deleting Members



```
1. create
2.Insert President
3.Insert secretary
4.insert after position(member)
5.Display list
6.Delete President
7.Delete Secretary
8.Delete Member
9.Find total No. of members
10.Sort list
11. Reselect List ++--##
12.Combine lists
13.Reverse Display
0. Exit
ENter your choice:      5

====== List: ======
12345   Rohan
984278  Mihir
53893   Dinesh
895385  Gautam
```

rudraksh.karpe.cs@ghrcem.raisoni.net