|  |  |
| --- | --- |
| Assignment No.: | 7 |
| Title: | Implement single linked list for the 'Pinnacle  Club' |
| Subject: | Data Structures Laboratory |
| Class: | S.Y. (C.S.E.) |
| Roll No.: |  |
| Assessment (Marks): |  |
| Signature and Date of Assessment: |  |

Experiment No. : 7

Experiment Title: Implement single linked list for the 'Pinnacle Club'

Objectives:

c, To understand how to implement single linked list.

d. To understand single linked list operations using structures.

Problem Statement: Department of Computer Engineering has student's club named 'Pinnack Club'. Students of Second, third and final year can be granted membership on request. Similarly, one may cancel the membership of club. The First position of list is always reserved for the President of club and last position of list is reserved for the Secretary of club. Write C++ program to maintain information of club member 's using singly linked list. Write functions 1, Add and delete the members.

2, Compute total number of members of club

1. Display members
2. Display list in reverse order Outcomes:

* Understanding the single linked list representation using structure.
* Understanding of single list operation such add, delete operations using structures.

# Structure

struct node

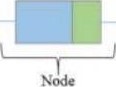
String

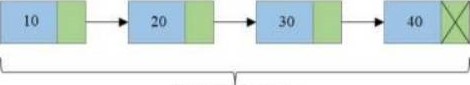
name; Struct node

\* next;

Singly Link List

Singly linked list is a basic linked list type. Singly linked list is a collection of nodes linked together in a sequential way where each node of singly linked list contains a data field and an address field which contains the reference of the nextnode. Singly linked listcan contain multiple data fie Ids but should contain at least single address field pointing to its connected next node.

Data.—\_.Address



Singly Linked List

Insert node at the beginning of Singly Linked List

l. Create a new node, say newNode points to the newly created node.

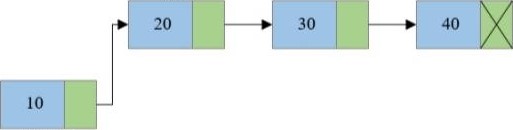
head



10

newNode

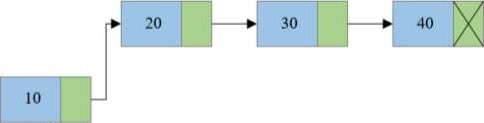
1. Link the newly created node with the head node, i.e. the newNode will now point to head node



head

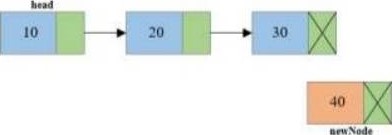
newNode

1. Make the new node as the head node, i.e. now head node will pointto newNode

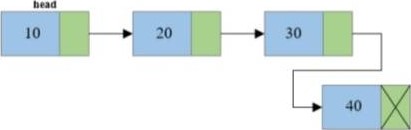


Insert node at the end of Singly Linked List

Create a new node and make sure that the address part of the new node points to NULL i.e., newNode>next=NULL

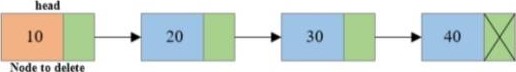


2. Traverse to the last node of the linked list and connect the last node of the list with the new node, i.e., last node will now point to new node. (lastNode->next = newNode).

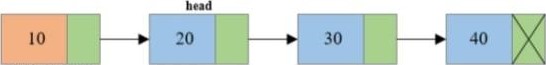


Steps to delete first node from Singly Linked List

Copy the address of first node i.e., head node to some temp variable say, tol)elete.

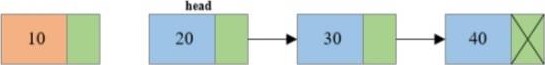


1. Move the head to the second node of the linked list i.e., head = head->next.



Node to delete

1. Disconnect the connection of first node to second node.



Node to delete

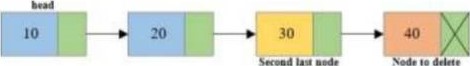
1. Free the memory occupied by the first node.

bead

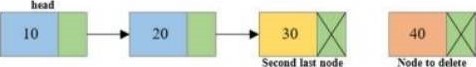
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 20 |  |  |  | |  | 40 |  |
|  |  | 30 |  |  |

Steps to delete last node of a Singly Linked List

l. Traverse to the last node of the linked list keeping track ofthe second last node in some temp variable say secondLastNode.



2. If the last node is the head node, then make the head node as NULL else disconnect the second last node with the last node i.e., secondLastNode->next = NULL.



3, Free the memory occupied by the last node.

head

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 10 |  |  | 20 |  |  | 30 |  |
|  |  |  |  |

Second last node

Conclusion: Thus, we implemented single linked list for the pinnacle club.

Questions

What is SIX?

ii. What is the structure of node of SLL? iii. How we can delete node from SLL? iv, How we can insert node in SLL?

Code:-

#include<iostream>

#include<math.h>

#include<string.h>

using namespace std;

struct node

{

 string s\_name;

 struct node \*next;

};

node \*temp,\*head;

node\* Create(string name)

{

    temp = new(struct node);

    if (temp == NULL)

    {

        cout<<"Memory Allocation Error"<<endl;

        return 0;

    }

    else

    {

        temp -> s\_name = name;

        temp -> next = NULL;

        return temp;

    }

}

void AddPresi()

{

    string name1;

    cout<<"Enter the name=";

    cin>>name1;

    temp=Create(name1);

    if (head == NULL)

    {

        head = temp;

        head -> next = NULL;

        cout<<"We have " <<name1<< " as a new President."<<endl;

    }

    else

    {

        temp -> next = head;

        head = temp;

        cout<<"We have " <<name1<< " as a new President."<<endl;

    }

}

void AddMember()

{

    int i;

    int counter = 1;

    cout<<"\nEnter the location where member to be added=";

    cin>>i;

    int pos = i - 1;

    string n;

    struct node \*ptr;

    struct node \*t = head;

    while ((t -> next) != NULL)

    {

            t = t -> next;

            counter++;

    }

    t = head;

    if (i == 1)

    {

        AddPresi();

    }

    else if (pos > counter || i <= 0)

    {

        cout<<"Entered position is out of scope."<<endl;

    }

    else

    {

        cout<<"Enter Name: ";

        cin>>n;

        temp = Create(n);

        while (pos--)

        {

            ptr = t;

            t = t -> next;

        }

        temp -> next = t;

        ptr -> next = temp;

        cout<<"Member Inserted at Position: "<<i<<endl;

    }

}

void AddSecr()

{

    string name;

    cout<<"Enter Name: ";

    cin>>name;

    struct node \*temp1 = head;

    temp = Create(name);

    if (head == NULL)

    {

        head = temp;

        head -> next = NULL;

    } else

    {

        while ((temp1 -> next) != NULL)

        {

            temp1 = temp1 -> next;

        }

        temp -> next = NULL;

        temp1 -> next = temp;

        cout<<"We have " <<name<< " as a new Secretary."<<endl;

    }

}

void DelMember()

{

    int i;

    int counter = 1;

    cout<<"\nEnter the location where member to be added=";

    cin>>i;

    int pos = i - 1;

    string n;

    struct node \*ptrl,\*ptrr;

    struct node \*t = head;

    while ((t -> next) != NULL)

    {

            t = t -> next;

            counter++;

    }

    t = head;

    if (i == 1)

    {

        ptrl = head;

        head = head -> next;

        delete ptrl;

    } else if (pos > counter || i <= 0)

    {

        cout<<"Entered member doesn't exist."<<endl;

    } else

    {

        while (pos--)

        {

            ptrl = t;

            t = t -> next;

            ptrr = t -> next;

        }

        ptrl -> next = ptrr;

        delete t;

        cout<<"Member Deleted at Position: "<<i<<endl;

    }

}

void AddDelete()

{

    int ch,ch1;

    char ch2;

    cout<<"\nWhich operation you want to carry out?\n";

    cout<<"\n1.Add Member\n2.Delete Member\n";

    cin>>ch;

    switch(ch)

    {

        case 1: do

                {

                    cout<<"which member you want to add?\n1.Add President\n2.Add Member\n3.Add Secretary\n";

                    cin>>ch1;

                    switch(ch1)

                    {

                        case 1:AddPresi();

                               break;

                        case 2:AddMember();

                               break;

                        case 3:AddSecr();

                               break;

                    }

                    cout<<"Do you want to keep adding?\nYes\nNo\n";

                    cin>>ch2;

                }while(ch2=='Y' || ch2=='y');

                break;

        case 2: do

                {

                    DelMember();

                    cout<<"Do you want to keep Deleting?\nYes\nNo\n";

                    cin>>ch2;

                }while(ch2=='Y' || ch2=='y');

                break;

    }

}

void TotalNumber()

{

    temp = head;

    int count = 0;

    while (temp != NULL)

    {

        count++;

        temp = temp -> next;

    }

     cout<<"Total number of members are="<<count<<endl;

}

void Display()

{

   temp = head;

    cout<<"President: ";

    cout<<temp -> s\_name<<" -> ";

    if(temp -> next != NULL)

    {

        temp = temp -> next;

    }

    while (temp -> next != NULL)

    {

        cout<< temp -> s\_name<<" -> ";

        temp = temp -> next;

    }

    cout<<"Secretary: ";

    cout<< temp -> s\_name<<" -> ";

    cout<<"NULL"<<endl;

}

void Rev(node \*t)

{

     if(t -> next != NULL)

    {

        Rev (t -> next);

    }

    if(t == head)

        cout<<"Secretary: "<<t -> s\_name<<endl;

    else if(t -> next == NULL)

        cout<<"President: "<<t -> s\_name<<" -> ";

    else

        cout<<"Member: "<<t -> s\_name<<" -> ";

}

void DisplayReverse()

{

    Rev(head);

}

int main()

{

     int ch;

     //head=NULL;

     char ch2;

     do

     {

         cout<<"Enter choice: \n1.Add and Delete Member\n2.Total number of Members\n3.Display Members\n4.Display Members in Reverse Order\n";

         cin>>ch;

        switch(ch)

        {

            case 1:AddDelete();

                break;

            case 2:TotalNumber();

                break;

            case 3:Display();

                break;

            case 4:DisplayReverse();

                break;

        }

        cout<<"Do you want to continue operations?\nYes\nNo\n";

        cin>>ch2;

     } while(ch2=='Y' || ch2=='y');

    return 0;

}

Output:-

Enter choice:

1.Add and Delete Member

2.Total number of Members

3.Display Members

4.Display Members in Reverse Order

1

Which operation you want to carry out?

1.Add Member

2.Delete Member

1

which member you want to add?

1.Add President

2.Add Member

3.Add Secretary

1

Enter the name=Anurag

We have Anurag as a new President.

Do you want to keep adding?

Yes

No