

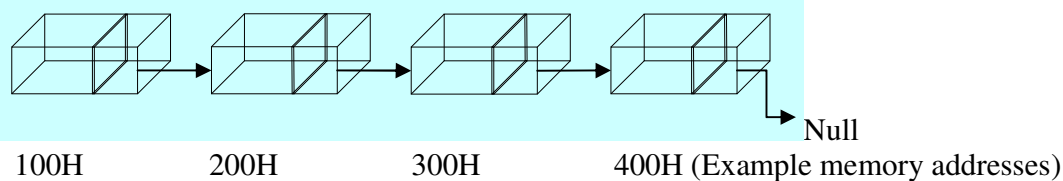
# Linked Lists

## Terminology:

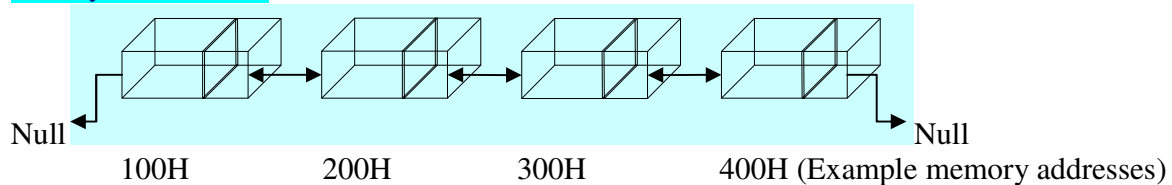
1. Definition: Linked List is a linear collection of nodes. It is a Dynamic Data structure.
2. Node: Node is a data item with two parts, namely INFO(Stores the data) and NEXT(Stored the address of the next node)
3. Linear: Only one values follows the current value (As opposed by a tree where two or more values can follow the current value)
4. Dynamic Data structure: In which the memory is allocated at run-time by the user using the *new* operator.
5. Memory Leak: Not de-allocating (by the use of *delete* command) memory causes reduction of available memory.
6. Heap/ Free Store: Area in the RAM reserved for Dynamic memory allocation.
7. Garbage Collection: De-allocating (by the use of *delete* command) reserved memory and making it free for further use.
8. Unnamed Memory locations: Allocated memory of which the base address is not stored (so it cannot be accessed later)

## Types of Linked Lists

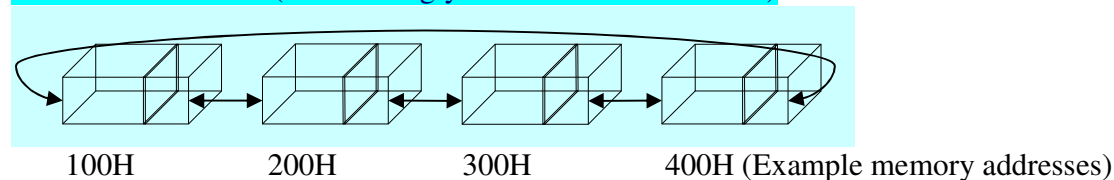
### Singly Linked Lists



### Doubly Linked Lists



### Circular Linked List (Can be Singly or a Double Linked List)



### Differences between linked lists and Arrays (Also advantages of a linked list)

Factor	Array	Linked List
Memory Allocation	Static (By compiler)	Dynamic (By user), May cause memory leaks
Memory Area	Decided by the compiler	Uses Free Store
Size	Fixed	Can grow/Shrink
Organization	Sequential	Can be circular
Algorithms	Complicated insertion and deletions	Nodes can easily be inserted/deleted without shifting

### Program(Function) Index

*Note:- The beginning (creation of a Linked list) in many programs is the same.*

1. Creation & Traversal of a simple linked list (Also counting the nodes)
2. Inserting a node in a linked list
3. Deleting a node from a linked list
4. Merging/Concatenating two linked lists
5. Splitting a linked list
6. Reversing a linked list
7. Swapping 2 nodes of a linked list
8. Sorting a linked list
9. Displaying a linked list in reverse (and not reversing it)
10. Creating (using while) and displaying(using recursion)
11. Creation and traversal of a doubly linked list
12. Creation and traversal of a circular list

## Program 1

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL;

void main()
{
    clrscr();
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    int count=0;
    cout<<"List contents:\n";
    for(ptr=start; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<'\t';
        count++;
    }
    cout<<"\nNumber of nodes :"<<count;
    getch();
}
```

## Program 2

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL;

void main()
{
    clrscr();
    int pos, value;
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }
    cout<<"Enter position : ";
    cin>>pos;
    cout<<"Enter value : ";
    cin>>value;
    if(pos==1)
    {
        ptr=new NODE;
        ptr->info=value;
        ptr->next=start;
        start=ptr->next;
    }
    else
    {
        int count=1;
        for(ptr=start; ptr; ptr=ptr->next)
        {
            if(count==pos-1)
            {
                NODE *temp=new NODE;
                temp->info=value;
                temp->next=ptr->next;
                ptr->next=temp;
            }
            count++;
        }
    }
}
```

```
    for(ptr=start; ptr; ptr=ptr->next)
    {      cout<<ptr->info<<'\t';
    }

    getch();
}
```

### Program 3

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    int pos;
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }
    cout<<"Enter position to delete: ";
    cin>>pos;
    if(pos==1)
    {
        temp=start;
        start=start->next;
        delete temp;
    }
    else
    {
        int count=1;
        for(ptr=start; ptr; ptr=ptr->next)
        {
            if(count==pos-1)
            {
                temp=ptr->next;
                ptr->next=ptr->next->next;
                delete temp;
                break;
            }
            count++;
        }
    }
    for(ptr=start; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<"\t";
    }
    getch();
}
```

## Program 4

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start1=NULL, *start2=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start1==NULL)
        {
            ptr=start1=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    //creation of 5 more nodes
    for(i=1; i<=5; i++)
    {
        if(start2==NULL)
        {
            ptr=start2=new NODE;
        }
        else
        {
            ptr->next=new NODE;
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }
    //Concatenation
    for(ptr=start1; ptr; ptr=ptr->next)
    {
        if(ptr->next==NULL)
        {
            ptr->next=start2;
            break;
        }
    }
}
```

```
        }  
    for(ptr=start1; ptr; ptr=ptr->next)  
    {        cout<<ptr->info<<'\t';  
    }  
    getch();  
}
```

## Program 5

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start1=NULL, *start2=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    //creation of 6 nodes
    for(int i=1; i<=6; i++)
    {
        if(start1==NULL)
        {
            ptr=start1=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==6)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    //Making two halves of the list
    int count=0;
    for(ptr=start1; ptr; ptr=ptr->next)
    {
        count++;
        if(count==3)
        {
            start2=ptr->next;
            ptr->next=NULL;
            break;
        }
    }

    cout<<"List 1 : \n";
    for(ptr=start1; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<"\t";
    }

    cout<<"\nList 2 : \n";
    for(ptr=start2; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<"\t";
    }

    getch();
}
```

## Program 6

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    //Reversing
    NODE *p=NULL, *n=start->next;
    ptr=start;
    ptr->next=NULL;
    while(ptr)
    {
        p=ptr;
        ptr=n;
        n=ptr->next;
        ptr->next=p;
    }
    start=p;

    cout<<"Reversed List : \n";
    for(ptr=start; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<'\t';
    }

    getch();
}
```

## Program 7

```
#include<iostream.h>
#include<conio.h>
// Note: to swap 2 nodes you must store the address
// of affected nodes (4 if not the first and the last)
// in different variables and then perform swapping
//The program below swaps the first and the last node

struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *temp=NULL,
 *first, *second, *secondLast, *last;

void main()
{ clrscr();
  //creation of 5 nodes
  for(int i=1; i<=5; i++)
  {
    if(start==NULL)
    {
      ptr=start=new NODE;
      first=start;
    }
    else
    {
      ptr=ptr->next;
    }
    ptr->info=i;
    //can be cin>>ptr->info to make the user enter the value
    if(i==5)
    {
      ptr->next=NULL;
      last=ptr;
    }
    else
    {
      ptr->next=new NODE;
    }
    if(i==2) second=ptr;
    if(i==4) secondLast=ptr;
  }

  //Swapping
  start=last;
  last->next=second;
  secondLast->next=first;
  first->next=NULL;

  cout<<"Swapped List : \n";
  for(ptr=start; ptr; ptr=ptr->next)
  {
    cout<<ptr->info<<'\t';
  }

  getch();
}
```

## Program 8

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    cout<<"Original List : \n";
    for(ptr=start; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<"\t";
    }
    cout<<endl;

    //Sorting in descending order knowing that there are 5 nodes
    for(i=0; i<4; i++)
    {
        ptr=start;
        for( int j=0; j<4-i; j++)
        {
            if(ptr->info < ptr->next->info)
            {
                int temp=ptr->info;
                ptr->info=ptr->next->info;
                ptr->next->info=temp;
            }
            ptr=ptr->next;
        }
    }

    cout<<"Sorted List : \n";
    for(ptr=start; ptr; ptr=ptr->next)
```

```
        {      cout<<ptr->info<<'\t';  
        }  
    getch();  
}
```

## Program 9

```
#include<iostream.h>
#include<conio.h>
struct NODE
{
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *temp=NULL;

void main()
{
    clrscr();
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {
        if(start==NULL)
        {
            ptr=start=new NODE;
        }
        else
        {
            ptr=ptr->next;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {
            ptr->next=NULL;
        }
        else
        {
            ptr->next=new NODE;
        }
    }

    cout<<"Original List : \n";
    for(ptr=start; ptr; ptr=ptr->next)
    {
        cout<<ptr->info<<'\t';
    }
    cout<<endl;

    cout<<"Displayed in reverse : \n";
    for(i=1; i<=5; i++)
    {
        ptr=start;
        for( int j=0; j<=4-i; j++)
        {
            ptr=ptr->next;
        }
        cout<<ptr->info<<"\t";
    }

    getch();
}
```

---

## Program 10

Convert the loop for creation to *while*. (Using user inputs (Y/N) to control the loop is better).

Use the following function to display

```
void display(NODE *ptr)
{
    if(ptr)
    {
        cout<<ptr->info;
        display(ptr->next);
    }
}
```

Call it as shown below in main():

```
display(start)
```

## Program 11

```
#include<iostream.h>
#include<conio.h>
struct NODE
{   NODE *prev;
    int info;
    NODE *next;
}*start=NULL, *ptr=NULL, *save=NULL;

void main()
{   clrscr();
    int pos;
    //creation of 5 nodes
    for(int i=1; i<=5; i++)
    {   if(start==NULL)
        {   ptr=start=new NODE;
            ptr->prev=NULL;
        }
        else
        {   save=ptr;
            ptr=ptr->next;
            ptr->prev=save;
        }
        ptr->info=i;
        //can be cin>>ptr->info to make the user enter the value
        if(i==5)
        {   ptr->next=NULL;
        }
        else
        {   ptr->next=new NODE;
        }
    }

    for(ptr=start; ptr; ptr=ptr->next)
    {   cout<<ptr->info<<'\t';
    }

    getch();
}
```

## Program 12

**Creation:**

Give `ptr->next=start` instead of `ptr->next=NULL` at the last node.

**Traversal:**

Give a do-while loop which runs till ptr reaches start again.

[END]