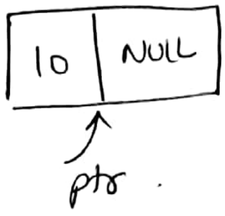
Lecture - 15

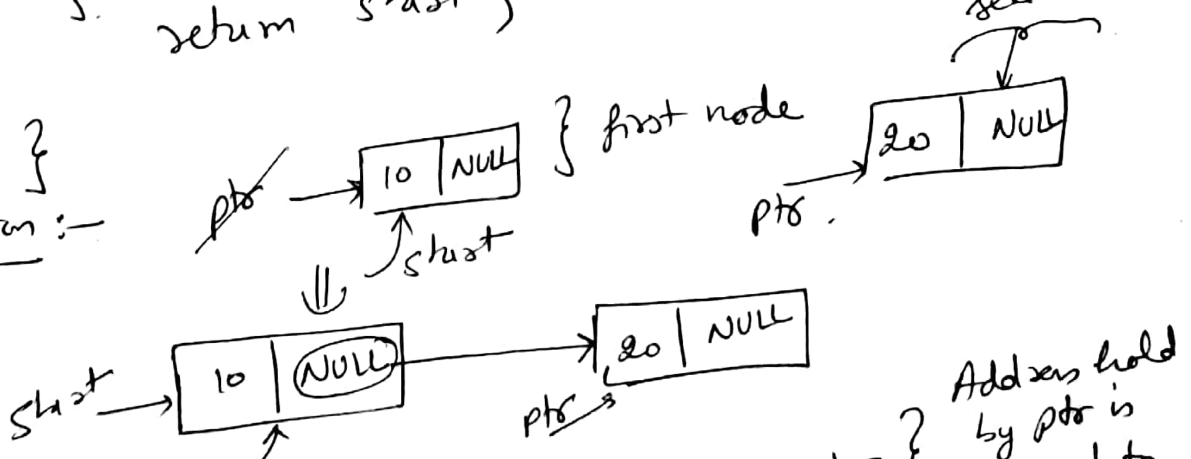
Creating and Traversing a Singly linked list.

INFO	LINK

```
typedef struct list
        {
represent info  ——> int data;
represent link  ——> struct list *next;
part
        } node;


    int main()
    {
     node *start = NULL;
     int n;
     Scanf("%d", &n);
      Start = Creatlist (n);
     void display list (node *start);

     return 0;
     }
 node * Create list (int n)
  {
     int i = 0;
     node *start, *ptr, *temp;
   for( i=0 ; i<n ; i++)
   {
 ptr = (node *) malloc (size of (node));
scanf("%d", &(ptr → data));
  ptr → next = NULL

if ( start == NULL)   // control will come only
                         for 1st time.
  {
  start = ptr;

  }

else
  {
     temp = start ;
     while (temp → next != NULL)
       {
         temp = temp → next;   // update the pointer
                                  temp.
       }
     temp → next = ptr;   // Connecting the
                             two nodes.

   }

}   return start;

}
```



Observation :-



temp → next = ptr } Address hold by ptr is assigned to temp → next.

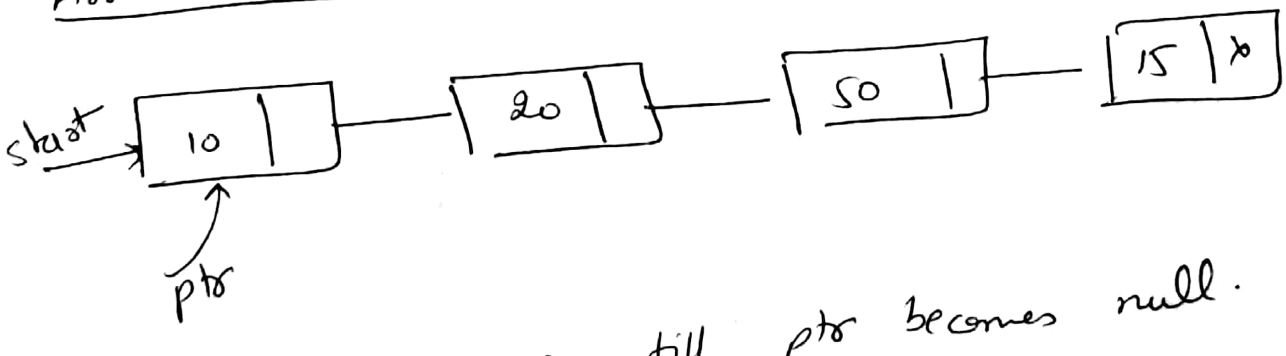
To display the list.

```
void display ( node *start)
  {
      node * ptr = ~~head~~ start;
      while ( ptr ! = NULL)
      {
       printf (" %d", ptr -> data);
         ptr = ptr -> next;

      }
  }
```

Observation

- list has been created.
- Passing the start to the display function

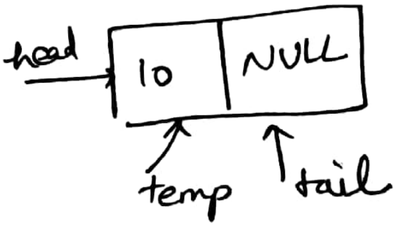
Assume the created list



- So we traverse till ptr becomes null.
- print the data part.
- update the ptr at each iteration.

Creating a singly linke list ~~[illegible]~~ in C++.

```
typedef struct list
    {
      int data;
      struct list * next;
    } node;

class linked_list
{
    private :
       node * head , * tail ;

    public :
    void insert_node ( int n )
    {
      node * temp = new node ; // created a node with help of new. function

      temp -> data = n ;
      temp -> next = NULL ;

    if ( head == NULL)
     {
          head = temp ;
          tail = temp ;

     }
  else
     {    tail -> next = temp ;
          tail = tail -> next ;

  ?  }
```



```
void display ()
 {
  node * ptr ;
  ptr = head ;
  while ( ptr ! = NULL )
  {
    cout << ptr -> data << " " ;
    ptr = ptr -> next ;
  }
 } ;

 int main ( )
    {
      linked_list a ;
      a. insert_node (7);
      a. insert_node (3);
      a. insert_node (11);
      a. display ();
      return 0;
    }
```

This can be done through a loop.