Linear List.

A linear list stores a collection of objects of a certain type, usually denoted as the elements of the list. The elements are ordered within the linear list in a linear sequence. Linear lists are usually simply denoted as lists.

Unlike an array, a list is a data structure allowing insertion and deletion of elements at an arbitrary position of the sequence. If the position is given, for example by a reference, such a modification takes only a constant number of operations, that is, no effortful copying of entries is necessary and all insertion and deletion operations take an equally short time. Conversely, however, one cannot access a single element via an (integral) index in constant time, as in the case of an array, without having searched for it before and having received a reference to it. Furthermore, lists are not limited to a certain maximum number of elements from the beginning on (like an array). So they are a dynamic data structure.

List implementation.
Lists can be implemented in many ways as follows:

1. Array based lists: Being one of the first data structures new programmers learn to work with, it is the most widely known implementation of a linear list. An array is stored in one contiguous block of memory.

In most programming languages, arrays have a fixed size once declared. This means that you can change item values but you can't add or delete new items. Arrays allow for quick random access, meaning that you can get the element at any given position in efficiently (in constant time).

Write a C program to implement various operations of linear list using arrays.

```c
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>

#define MAX 20 //maximum no of elements in the list

//user defined datatypes
struct
{
    int list[MAX];
    int element;  //new element to be inserted
    int pos;  //position of the element to be inserted or deleted
    int length;  //total no of elements
};

typedef enum boolean boolean;

//function prototypes
```
int menu(void); //function to display the list of operations
void create(void); //function to create initial set of elements
void insert(int, int); //function to insert the given element at specified position
void delet(int); //function to delete the element at given position
void find(int);  //function to find the position of the given element, if exists
void display(void); //function to display the elements in the list
boolean islistfull(void); //function to check whether the list is full or not
boolean islistempty(void);  //function to check whether the list is empty or not

void main()
{
    int ch;
    int element;
    int pos;

    l.length = 0;

    while(1)
    {
        ch = menu();
        switch (ch)
        {
            case 1:
                l.length = 0;
                create();
                break;
            case 2:
                if (islistfull() != true)
                {
                    printf("Enter the New element : ");
                    scanf("%d", &element);
                    printf("Enter the Position : ");
                    scanf("%d", &pos);
                    insert(element, pos);
                }
                else
                {
                    printf("List is Full. Cannot insert");
                    printf("nPress any key to continue...");
                    getch();
                }
                break;
            case 3:
                if (islistempty() != true)
                {
                    printf("Enter the position of element to be deleted : ");
                    scanf("%d", &pos);
                    delet(pos);
                }
} else {
    printf("List is Empty.");
    printf("\nPress any key to continue...");
    getch();
    break;
}

case 4:
    printf("No of elements in the list is \%d", l.length);
    printf("\nPress any key to continue...");
    getch();
    break;

case 5:
    printf("Enter the element to be searched : ");
    scanf("\%d", &element);
    find(element);
    break;

case 6:
    display();
    break;

exit(0);
break;
default:
    printf("Invalid Choice");
    printf("\nPress any key to continue...");
    getch();
}
}

//function to display the list of elements
int menu()
{
    int ch;
    clrscr();
    printf("\n\t********************************************\n");
    printf("\n\t******LIST Implementation Using Arrays*******\n");
    printf("\n\t********************************************\n");
    printf("\n\t1. Create\n\t2. Insert\n\t3. Delete\n\t4. Count\n\t5. Find\n\t6. Display\n\t7. Exit\n\tEnter your choice : ");
    scanf("\%d", &ch);
    printf("\n\n");
    return ch;
}

//function to create initial set of elements
void create(void)
{
    int element;
    int flag=1;
    while(flag==1)
    {
        printf("Enter an element : ");
        scanf("%d", &element);
        l.list[l.length] = element;
        l.length++;
        printf("To insert another element press '1' : ");
        scanf("%d", &flag);
    }
}

//function to display the elements in the list
void display(void)
{
    int i;
    for (i=0; i<l.length; i++)
    {
        printf("Element %d : %d \n", i+1, l.list[i]);
        printf("Press any key to continue...");
        getch();
    }
}

//function to insert the given element at specified position
void insert(int element, int pos)
{
    int i;
    if (pos == 0)
    {
        printf("Cannot insert at zeroth position");
        getch();
        return;
    }
    if (pos-1 > l.length)
    {
        printf("Only %d elements exist. Cannot insert at %d position", l.length, pos);
        printf("Press any key to continue...");
        getch();
    }
    else
    {
        for (i=l.length; i>=pos-1; i--)
        {
            l.list[i+1] = l.list[i];
        }
        l.list[pos-1] = element;
    }
}
l.length++; 
}

//function to delete the element at given position
void delet(int pos)
{
    int i;
    if(pos == 0)
    {
        printf("Cannot delete at zeroth position");
        getch();
        return;
    }
    if (pos > l.length)
    {
        printf("Only %d elements exit. Cannot delete", l.length, pos);
        printf("Press any key to continue...");
        getch();
        return;
    }
    for (i=pos-1; i<l.length; i++)
    {
        l.list[i] = l.list[i+1];
    }
    l.length--; 
}

//function to find the position of the given element, if exists
void find(int element)
{
    int i;
    int flag = 1;
    for (i=0; i<l.length; i++)
    {
        if(l.list[i] == element)
        {
            printf("%d exists at %d position",element, i+1);
            flag = 0;
            printf("Press any key to continue...");
            getch();
            break;
        }
    }
    if(flag == 1)
    {
        printf("Element not found. Press any key to continue...");
        getch();
    }
//function to check whether the list is full or not
boolean islistfull(void)
{
    if (l.length == MAX)
        return true;
    else
        return false;
}

//function to check whether the list is empty or not
boolean islistempty(void)
{
    if (l.length == 0)
        return true;
    else
        return false;
}