Lab 6: Implementation of different operations in static lists.

Theory: A list is a collection of homogenous set of elements. In case of Static DS, the size of DS is fixed. The content of the DS can be modified but we cannot change the memory space allocated to it. In case of Dynamic DS, the size of the DS is not fixed and can be modified at run time. In this lab, we are going to implement different operations in a static list.

Let, location is the location at which the element is to be inserted/deleted/modified, element be the element we are inserting, MAX be the maximum size of List, SIZE be the current size of List.

Insertion in List:

```
Step 1:Start
Step 2: if SIZE >= MAX then Display List Overflow
Step 3: for the value of i = SIZE -1 to location-1
                A[i + 1] = A[i]
                decrease i by 1
Step 4:A [location - 1] = element
        SIZE = SIZE + 1
Step 5: Stop
```

Deletion in List:

```
Step 1: Start
Step 2: for i =location to SIZE
            A[i-1]=A[i]
            Increase I by 1
Step 3: A[Size-1] = 0
        SIZE=SIZE-1
Step 4: End
```

Modification in List:

```
1. Start.
2. Read location and element to be updated.
3. Set element at given position as:
       A[location] =element
4. Stop
```

Traversing a List :

```
Step1: Start
Step 2: Initialize i=0
Step 3: do until i<=size
            display A[i]
                increase i by 1
Step 4: End
```

Source Code:

```c
#include <stdio.h>
#define MAX 100
struct StaticList
{
    int data[MAX];
    int size;
};
typedef struct StaticList List;
void create_list(List*);
void ins(List*,int,int);
void del(List*,int);
void modify(List*,int,int);
void display(List*);
int main()
{
    List L;
    L.size = 0;
    int choice,element,pos;
    do
    {
        printf("\n");
        printf("Select a option. \n");
        printf("1. READ LIST\n");
        printf("2. INSERT ELEMENT\n");
        printf("3. DELETE ELEMENT\n");
        printf("4. MODIFY ELEMENT\n");
        printf("5. DISPLAY ELEMENTS\n");
        printf("6. EXIT\n");
        printf(">");
        scanf("%d",&choice);
        switch (choice)
        {
            case 1:
                create_list(&L);
                break;
            case 2:
                printf("Enter position : ");
                scanf("%d",&pos);
                printf("Enter element : ");
                scanf("%d",&element);
                ins(&L,pos,element);
                break;
            case 3:
                printf("Enter position : ");
                scanf("%d",&pos);
                del(&L,pos);
                break;
            case 4:
                printf("Enter position : ");
                scanf("%d",&pos);
                printf("Enter new element : ");
                scanf("%d",&element);
                modify(&L, pos, element);
                break;
            case 5:
                display(&L);
                break;
        }
        }while (choice!= 6);
        return 0;
}
```

```
void create_list(List *P)
{
    int i;
    printf("Enter number of elements to be inserted \n");
    scanf("%d",&(P->size));
    printf("Enter the elements of the list \n");
    for (i=0;i<P->size;i++)
    {
        scanf("%d",&(P->data[i]));
    }
    printf("SUCCESS");
}
void ins(List *P,int position, int e)
{
    int i;
    if(P->size>=MAX)
    {
        printf("Sorry, LIST Overflow\n");
    }
    for(i=P->size-1;i>=position-1;i--)
    {
        P->data[i+1]=P->data[i]; // shifting element to right by
position
    }
    P->data[position-1]=e;
    printf("\nSUCCESS");
    P->size++;
}
void display(List *P)
{
    int i;
    printf("\nThe elements of list are \n");
    for(i=0;i<P->size;i++)
        printf("%d \t",P->data[i]);
}
void del(List *P, int position)
{
    int i;
    for(i=position;i<P->size;i++)
        {
            P->data[i-1]=P->data[i];// shifting elements to the left
by 1 position
        }
    P->data[P->size-1]=0;// Store 0 at the last location to mark it
empty
    printf("The element from Location %d has been deleted
\n",position);
    P->size--;
}
void modify(List *P, int position, int e)
{
    int i;
    P->data[position-1]=e;
    printf("\nSUCCESS");
}
```

Conclusion: In this lab, we successfully performed various operations in a static list.