EXPERIMENT 1

1.1 OBJECTIVE
1. To create a singly linked list of integers.
2. Delete a given integer from the above linked list.
3. Display the contents of the above list after deletion.

1.2 RESOURCE:
Turbo C

1.3 PROGRAM LOGIC
1. Create a node using structure
2. Dynamically allocate memory to node
3. Create and add nodes to linked list

1.4 PROCEDURE
Go to debug -> run or press CTRL + F9 to run the program.

1.5 SOURCE CODE
program to create a single linked list, delete the contents and display the contents

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<math.h>
/*declaring a structure to create a node*/
struct node
{
    int data;
    struct node *next;
};
struct node *start;

/* inserting nodes into the list*/
/*function to insert values from beginning of the the single linked list*/
void insertbeg(void)
{
    struct node *nn;
    int a;
    /*allocating implicit memory to the node*/
    nn=(struct node *)malloc(sizeof(struct node));
    printf("enter data:");
    scanf("%d",&nn->data);
    a=nn->data;
    if(start==NULL)            /*checking if List is empty*/
    {
        nn->next=NULL;
        start=nn;
    }
    else
    {
        nn->next=start;
        start=nn;
    }
    printf("%d succ. inserted\n",a);
    return;
}
 /*function to insert values from the end of the linked list*/
```

```
void insertend(void)
{
    struct node *nn,*lp;int b;
    nn=(struct node *)malloc(sizeof(struct node));
    printf("enter data:");
    scanf("%d",&nn->data);
    b=nn->data;
    if(start==NULL)
    {
        nn->next=NULL;
        start=nn;
    }
    else
    {
        lp=start;
        while(lp->next!=NULL)
        {
            lp=lp->next;
        }
        lp->next=nn;
        nn->next=NULL;
    }
    printf("%d is succ. inserted\n",b);
    return;
}
```

/*function to insert values from the middle of the linked list*/

```
void insertmid(void)
{
    struct node *nn,*temp,*ptemp;int x,v;
    nn=(struct node *)malloc(sizeof(struct node));
    if(start==NULL)
    {
        printf("sll is empty\n"); return;
    }
    printf("enter data before which no. is to be inserted:\n");
    scanf("%d",&x);
    if(x==start->data)
    {
        insertbeg();
        return;
    }
    ptemp=start;
    temp=start->next;
    while(temp!=NULL&&temp->data!=x)
    {
        ptemp=temp;
        temp=temp->next;
    }
    if(temp==NULL)
    {
        printf("%d data does not exist\n",x);
    }
    else
    {
            printf("enter data:");
            scanf("%d",&nn->data);
            v=nn->data;
            ptemp->next=nn;
            nn->next=temp;
            printf("%d succ. inserted\n",v);
      }
      return;
}
 /*deletion operation*/
void deletion(void)
{
      struct node *pt,*t;
      int x;
      if(start==NULL)
      {
            printf("sll is empty\n");
           return;
      }
      printf("enter data to be deleted:");
      scanf("%d",&x);
      if(x==start->data)
      {
            t=start;
             /* assigning first node pointer to next nod pointer to delete a data
            from the  starting of the node*/
            start=start->next;
            free(t);
            printf("%d is succ. deleted\n",x);
            return;
      }
      pt=start;
      t=start->next;
      while(t!=NULL&&t->data!=x)
      {
            pt=t;t=t->next;
      }
      if(t==NULL)
      {
            printf("%d does not exist\n",x);return;
       }
      else
      {
            pt->next=t->next;
      }
      printf("%d is succ. deleted\n",x);
      free(t);
      return;
}
void display(void)
{
      struct node *temp;
      if(start==NULL)
      {
            printf("sll is empty\n");
            return;
      }
      printf("elements are:\n");
      temp=start;
      while(temp!=NULL)
      {
            printf("%d\n",temp->data);
            temp=temp->next;
        }
        return;
    }
     /*  main program*/
    int main()
    {
        int c,a;    start=NULL;
        do
        {
            printf("1:insert\n2:delete\n3:display\n4:exit\nenter choice:");
            scanf("%d",&c);
            switch(c)
            {
                case 1:
                printf("1:insertbeg\n2:insert end\n3:insert mid\nenter choice:");
                scanf("%d",&a);
                switch(a)
                {
                     case 1:insertbeg(); break;
                    case 2:insertend(); break;
                    case 3:insertmid(); break;
                }
                break;
                case 2:deletion(); break;
                case 3:display(); break;
                case 4:printf("program ends\n");break;
                default:printf("wrong choice\n");
                break;
            }
        }while(c!=4);return 0;
    }
```

1.6 PRE LAB QUESTIONS

1. What is data structure
2. How the memory is allocated dynamically
3. What is linked list
4. What is node
5. What are the types of linked list

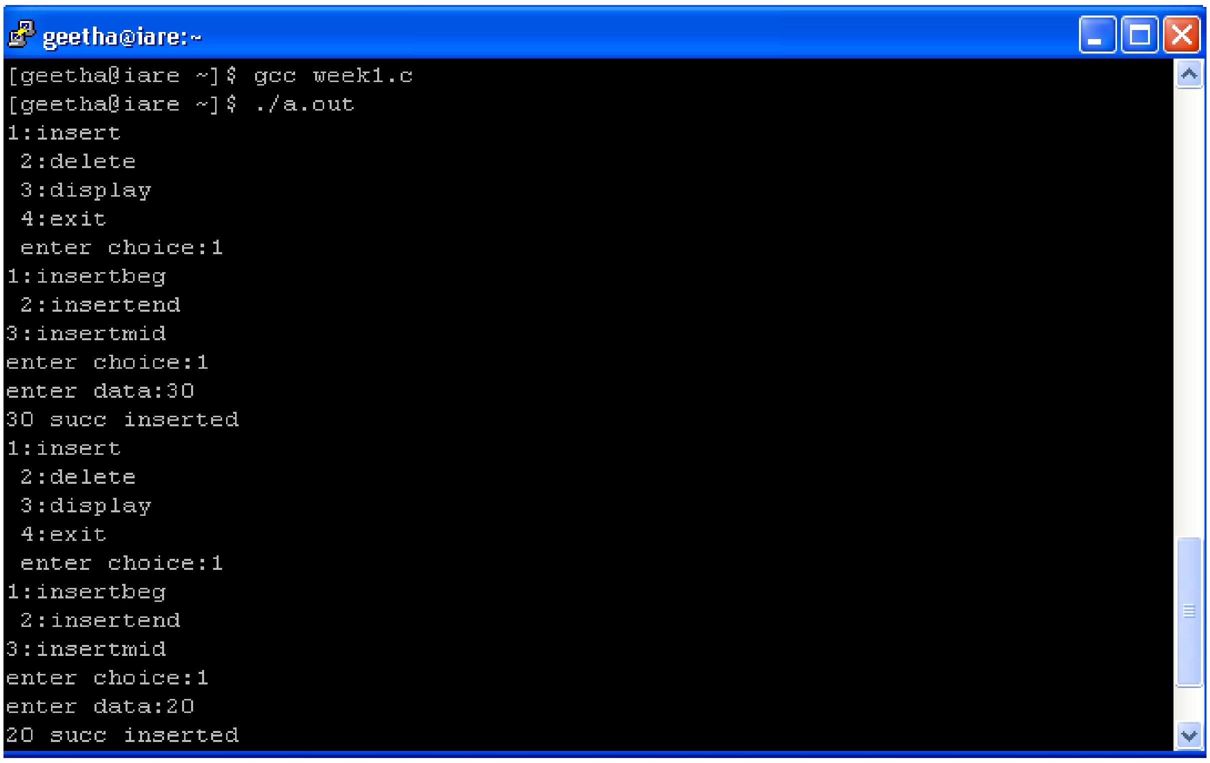
1.7 LAB ASSIGNMENT

1. Write a program to insert a node at first , last and at specified position of linked list
2. Write a program to delete a node from first, last and at specified position of linked list

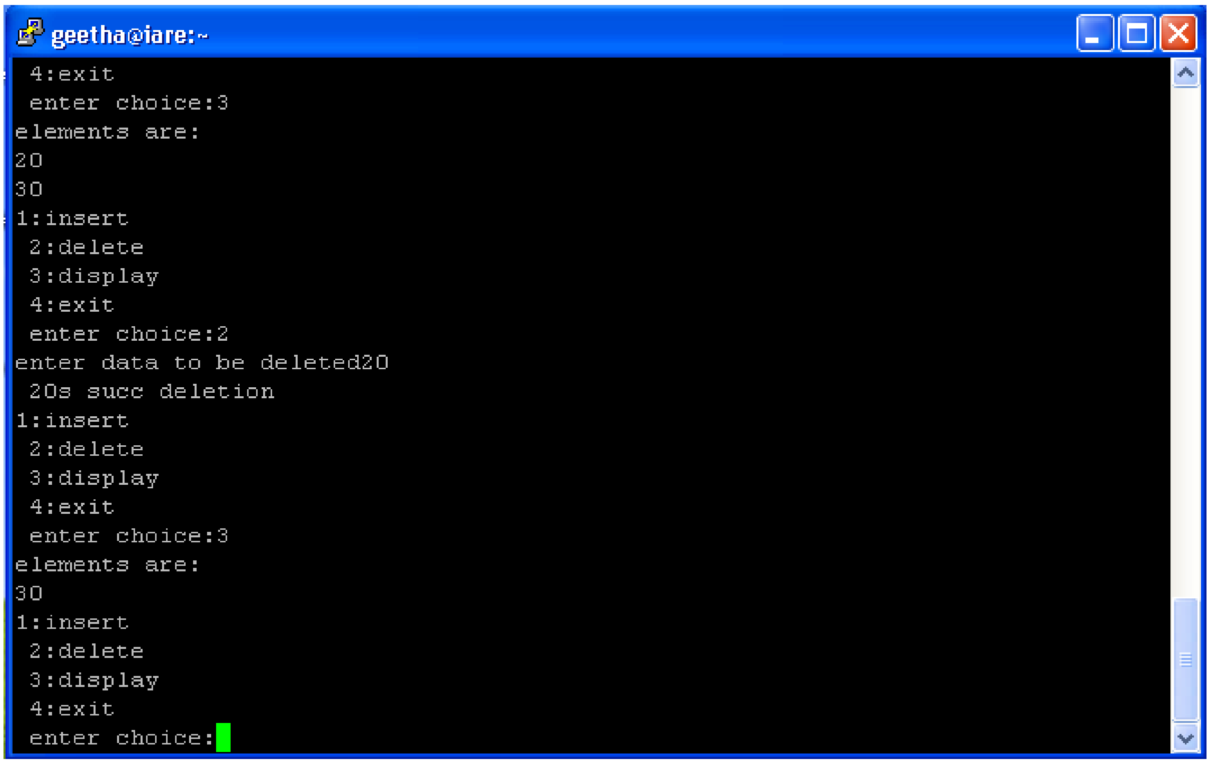
1.8 POST LAB QUESTIONS

1. How to represent linked list
2. How will you traverse linked list in reverse order
3. List the advantages and disadvantages of linked list?

1.9 INPUT AND OUTPUT



```
[geetha@iare ~]$ gcc week1.c
[geetha@iare ~]$ ./a.out
1:insert
 2:delete
 3:display
 4:exit
 enter choice:1
1:insertbeg
 2:insertend
3:insertmid
enter choice:1
enter data:30
30 succ inserted
1:insert
 2:delete
 3:display
 4:exit
 enter choice:1
1:insertbeg
 2:insertend
3:insertmid
enter choice:1
enter data:20
20 succ inserted
```



```
 4:exit
 enter choice:3
elements are:
20
30
1:insert
 2:delete
 3:display
 4:exit
 enter choice:2
enter data to be deleted20
 20s succ deletion
1:insert
 2:delete
 3:display
 4:exit
 enter choice:3
elements are:
30
1:insert
 2:delete
 3:display
 4:exit
 enter choice:
```