WAP to Implement Singly Linked List with following operations
a) Create a Linked List.
b) Deletion of first element, specified element and Last element in the list.
c) Display the contents of the Linked List.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
   NODE x;
   x = (NODE) malloc (sizeof (struct node));
   if (x == NULL)
   {
    printf ("mem full \n");
     exit(0);
   }
   return x;
}
void freenode (NODE x)
{
  free (x);
}
```

```
NODE insert_front(NODE first, int item)
{
    NODE temp;
    temp = getnode();
    temp -> info = item;
    temp -> link = NULL;
    if (first == NULL)
    return temp;
    temp -> link = first;
    first = temp;
    return first;
}
NODE insert_rear(NODE first, int item)
{
    NODE temp, cur;
    temp = getnode();
    temp -> info = item;
    temp -> link = NULL;
    if (first == NULL)
    return temp;
    cur = first;
    while (cur -> link != NULL)
    cur = cur -> link;
    cur -> link = temp;
    return first;
}
NODE delete_front(NODE first)
{
    NODE temp;
    if (first == NULL)
    {
      printf("List is empty cannot delete\n");
      return first;
    }
```

```
    temp = first;
    temp = temp -> link;
    printf ("\n Item deleted at front-end is %d \n", first -> info);
    free (first);
    return temp;
}
NODE delete_rear (NODE first)
{
    NODE cur, prev;
    if (first == NULL)
    {
        printf ("\n list is empty cannot delete \n");
        return first;
    }
    if (first -> link == NULL)
    {
        printf ("\n Item deleted is %d \n", first -> info);
        free (first);
        return NULL;
    }
    prev = NULL;
    cur = first;
    while (cur -> link != NULL)
    {
        prev = cur;
        cur = cur -> link;
    }
    printf ("\n Item deleted at rear-end is %d", cur -> info);
    free (cur);
    prev -> link = NULL;
    return first;
}
```

```
void display(NODE first)
{
    NODE temp;
    if(first==NULL)
    printf("list empty cannot display items\n");
    printf("\n");
    for(temp=first; temp!=NULL; temp=temp->link);
    {
        printf("%d\n",temp->info);
    }
}
void main()
{
    int item,choice,pos;
    NODE first=NULL;
    printf("SINGLY LINKED LIST");
    for(;;)
    {
        printf("\n 1.Insert_rear\n2: Insert_front\n3. Delete_front
            \n4: Delete_rear\n5: display_list\n6: EXIT\n");
        printf("enter the choice\n");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: printf("\n enter the item at rear-end\n");
                    scanf("%d",&item);
                    first=insert_rear(first,item);
                    break;
            case 2: printf("\n enter the item at front-end\n");
                    scanf("%d",&item);
                    first=insert_front(first,item);
                    break;
```

```
case 3: first = delete_front(first);
        break;
case 4: first = delete_rear(first);
        break;
case 5: display(first);
        break;
default: exit(0);
        break;
    }
  }
}
```