DOUBLE LINKED LIST

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
        struct node *prev;
        int data;
        struct node *next;
}*head;
void begin_insertion();
void last_insertion();
void specified_insertion();
void begin_deletion();
void last_deletion();
void specified_deletion();
void display();
void search();
void operation();
int main()
{
        printf("*******operations*********");
        printf("\n1.begin
insertion\n2.last_insertion\n3.specified_insertion\n4.begin_deletion\n5.last_deletion\n6.specifie
d_deletion\n7.display\n8.search\n9.exit\n");
        printf("********************\n");
        operation();
return 0;}
void operation()
{

        int choice=0;
```

```
while(choice!=9)
{
        printf("enter your choice:");
        scanf("%d",&choice);
        switch (choice)
        {
                case 1:
                        begin_insertion();
                        break;
            case 2:
                last_insertion();
                break;
            case 3:
                specified_insertion();
                break;
            case 4:
                begin_deletion();
                break;
            case 5:
                last_deletion();
                break;
            case 6:
                specified_deletion();
                break;
            case 7:
                display();
                break;
            case 8:
                search();
                break;
            case 9:
```

```
                exit(0);
            default:
                printf("invaild number!!!! try again!!!\n");
                operation();
            }
        }

}
void begin_insertion()
{
        struct node *ptr;
        int item;
        ptr=(struct node*)malloc(sizeof(struct node *));
        if(ptr==NULL)
        {
                printf("over flow\n");
        }
        else
        {
        printf("enter a number to be inserted:");
        scanf("%d",&item);
        ptr->data=item;
        if(head==NULL)
        {
                ptr->prev=NULL;
                ptr->next=NULL;
                head=ptr;
        }
        else
        {
        ptr->prev=NULL;
```

```
        ptr->next=head;
        head->prev=ptr;
        head=ptr;
        printf("element insertion is completed\n");
        }
}
}
void last_insertion()
{
        struct node *ptr,*temp;
        int item;
        ptr=(struct node*)malloc(sizeof(struct node*));
        if(ptr==NULL)
        {
           printf("over flow\n");
        }
        else
        {
                printf("enter a number to be inserted:");
           scanf("%d",&item);
        ptr->data=item;
        if(head==NULL)
        {
                ptr->prev=NULL;
                ptr->next=NULL;
              head=ptr;
           }
           else
           {
                temp=head;
                while(temp->next!=NULL)
```

```
                {
                        temp=temp->next;
                        }
                        temp->next=ptr;
                        ptr->prev=temp;
                        ptr->next=NULL;
                        printf("insertion is completed\n");
                }
        }
}
void specified_insertion()
{
int item,loc,i;
struct node *ptr,*temp;
ptr=(struct node*)malloc(sizeof(struct node*));
if(ptr==NULL)
        {
        printf("\nover flow");
        }
else
        {
        printf("enter a number to be inserted:");
        scanf("%d",&item);
        ptr->data=item;
        printf("enter location where node has to be inserted:\n");
        scanf("%d",&loc);
        temp=head;
        for(i=1;i<loc;i++)
        {
        temp=temp->next;
         if(temp==NULL)
```

```
            {
                printf("can't inserted\n");
                return;
                 }
    }
    ptr->next=temp->next;
    ptr->prev=temp;
    temp->next->prev=ptr;
    temp->next=ptr;
    printf(" node inserted\n");
}
}
void begin_deletion()
{
        struct node *ptr;
        if (head==NULL)
        {
                printf("list is empty\n");
        }
        else if(head->next==NULL)
        {
                ptr=head;
                head->prev=NULL;
                head=NULL;
                free(ptr);
                printf("only one node is deleted\n");
        }
        else
        {
                ptr=head;
                ptr->next->prev=NULL;
```

```
                head=ptr->next;
                free(ptr);
                printf("first node is deleted\n");
        }
        }

void last_deletion()
{
        struct node *ptr;
        if(head==NULL)
        {
                printf("list is empty\n");
        }
        else if(head->next==NULL)
        {
                head=NULL;
                ptr=head;
                head=NULL;
                free(ptr);
                printf("only one node id deleted\n");
        }
        else
        {
                ptr=head;
                while(ptr->next!=NULL)
                {
                        ptr=ptr->next;
                }
                ptr->prev->next=NULL;
                free(ptr);
                printf("deleted last node from list\n");
```

```
        }
}
void specified_deletion()
{
        struct node *ptr,*temp;
        int var;
        printf("Enter the value of var:");
        scanf("%d",&var);
        ptr=head;
        while(ptr->data!=var)
        {
                ptr=ptr->next;
        if(ptr==NULL)
        {
                printf("can't delete\n");
                return;
        }
   }
   if(ptr->next==NULL)
  {
         ptr->prev->next=NULL;
   free(ptr);
   printf("deleted element is %d\n",var);
  }
  else
  {
        temp=ptr->next;
        ptr->prev->next=temp;
        temp->prev=ptr->prev;
        free(ptr);
        printf("deleted node is %d\n",var);
```

```
    }
}
void display()
{
        struct node *temp;
        if(head==NULL)
        printf("List is empty\n");
        else
        {
                printf("Elements in linked list\n");
                temp=head;
                while(temp!=NULL)
                {
                        printf("%d\n",temp->data);
                        temp=temp->next;
                }

        }
}
void search()
{
        struct node *temp;
        int var,c=0;
        printf("Enter the value of var:");
        scanf("%d",&var);
        temp=head;
        while(temp->next!=NULL)
        {
                c++;
                if(temp->data==var)
        {
```

```
            printf("Element is found at %d node\n",c);
            break;
        }
            temp=temp->next;
        }
        if(temp->data==var&&temp->next==NULL)
        {
            printf("Element is  found at %d node\n",c+1);
        }
        else if(temp->data!=var)
        {
            printf("element is not found\n");
            }
}
```