Doubly linked list insertion, deletion, display and search operations.

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
struct node *prev;
struct node *next;
int data;
};
struct node *head;
void insertion_beg();
void insertion_last();
void insertion_spec();
void deletion_beg();
void deletion_last();
void deletion_spec();
void display();
void search();
void main ()
{
int choice =0;
while(choice!= 9)
{
printf("\n*********Main Menu*********\n");
printf("\nChoose one option from the following list \n");
printf("\n================================================\n");

printf("\n1.Insert at begining\n2.Insert at last\n3.Insert at specific
position\n4.Delete from Beginning\n 5.Delete from last\n6.Delete from
specific position\n7.Search\n8.Display\n9.Exit\n");

printf("\nEnter your choice?\n");

scanf("\n%d",&choice);


switch(choice)

{

case 1:

insertion_beg();

break;

case 2:

insertion_last();

break;

case 3:

insertion_spec();

break;

case 4:

deletion_beg();

break;

case 5:

deletion_last();

break;

case 6:

deletion_spec();

break;
case 7:
search();
break;
case 8:
display();
break;
case 9:
exit(0);
break;
default:
printf("Please enter valid choice");

}


}


}
void insertion_beg()
{
struct node *ptr;
int item;
ptr = (struct node *)malloc(sizeof(struct node));
if(ptr == NULL)
{
printf("\noverflow");
}
```

```
else

{

printf("\nEnter Item value");

scanf("%d",&item);


if(head==NULL)

{

ptr->next = NULL;

ptr->prev=NULL;

ptr->data=item;

head=ptr;

}

else

{

ptr->data=item;

ptr->prev=NULL;

ptr->next = head;

head->prev=ptr;

head=ptr;


}

printf("\ninsertion success\n");


}
```

```
}
void insertion_last()
{
struct node *ptr,*temp;
int item;
ptr = (struct node *) malloc(sizeof(struct node));
if(ptr == NULL)
{
printf("\noverflow");
}
else
{
printf("\nEnter a value");
scanf("%d",&item);
ptr->data=item;
if(head == NULL)
{
ptr->next = NULL;
ptr->prev = NULL;
head = ptr;
}
else
{
temp = head;
while(temp->next!=NULL)
{
```

```
temp = temp->next;


}
temp->next = ptr;
ptr ->prev=temp;
ptr->next = NULL;
}



}
printf("\ninsertion success\n");
}
void insertion_spec()
{
struct node *ptr,*temp;
int item,loc,i;
ptr = (struct node *)malloc(sizeof(struct node));
if(ptr == NULL)
{
printf("\n overflow");
}
else
{
temp=head;
printf("Enter the location");
scanf("%d",&loc);
```

```
for(i=0;i<loc;i++)
{
temp = temp->next;
if(temp == NULL)
{
printf("\n There are less than %d elements", loc);
return;
}


}
printf("Enter value");
scanf("%d",&item);
ptr->data = item;
ptr->next = temp->next;
ptr -> prev = temp;
temp->next = ptr;
temp->next->prev=ptr;
printf("\ninsertion success\n");
}


}
void deletion_beg()
{
struct node *ptr;
if(head == NULL)
{
```

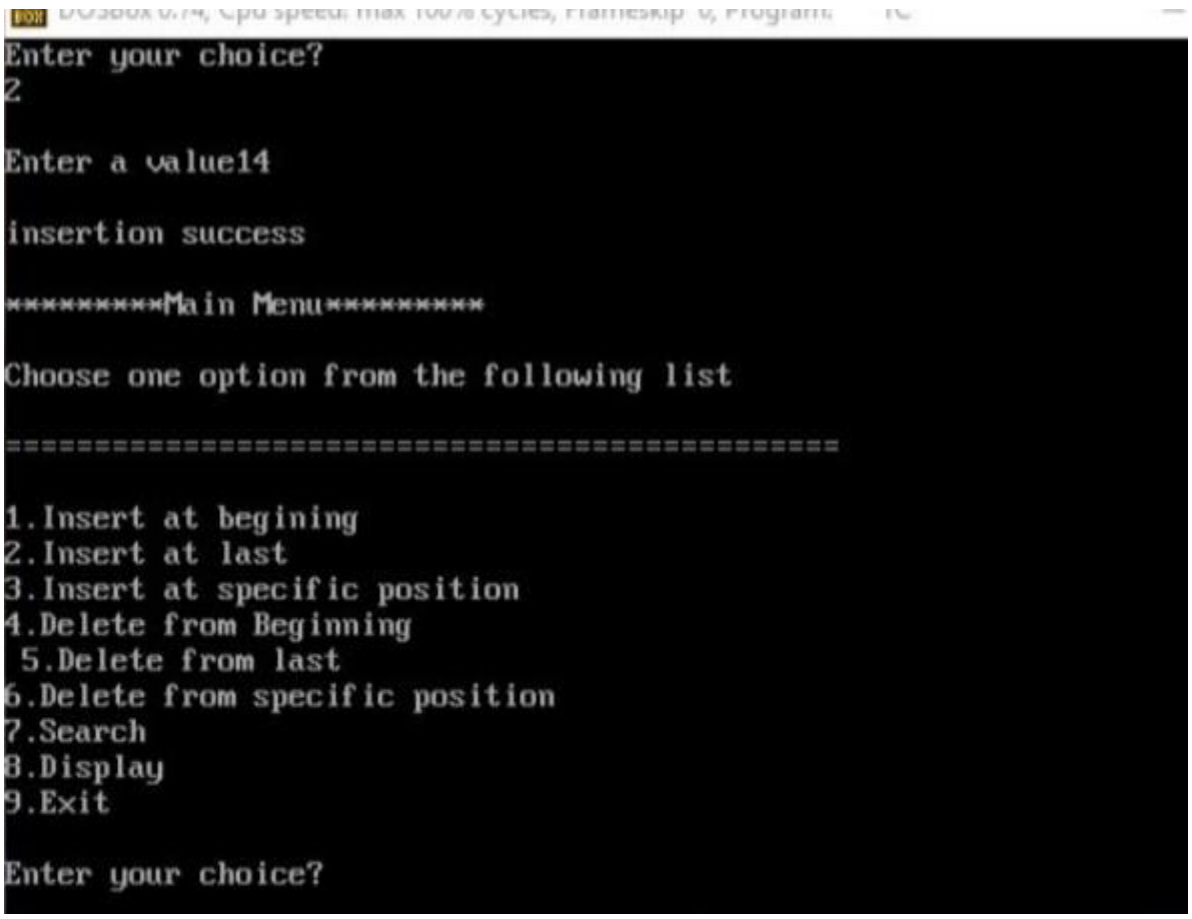
```
printf("\n underflow");
}
else if(head->next == NULL)
{
head = NULL;
free(head);
printf("\ndeletion success\n");
}
else
{
ptr = head;
head = head -> next;
head -> prev = NULL;
free(ptr);
printf("\ndeletion success\n");
}
}
void deletion_last()
{
struct node *ptr;
if(head == NULL)
{
printf("\n underflow");
}
else if(head->next == NULL)
{head = NULL; free(head);
```

```
printf("\ndeletion success\n");
}
else
{
ptr = head;
if(ptr->next != NULL)
{
ptr = ptr -> next;
}
ptr -> prev -> next = NULL;
free(ptr);
printf("\ndeletion success\n");
}
}
void deletion_spec()
{
struct node *ptr, *temp;
int val;
printf("\n Enter the item to be deleted : ");
scanf("%d", &val);
ptr = head;
while(ptr -> data != val)
ptr = ptr -> next;
if(ptr -> next == NULL)
{
printf("\ndeletion not possible\n");
```

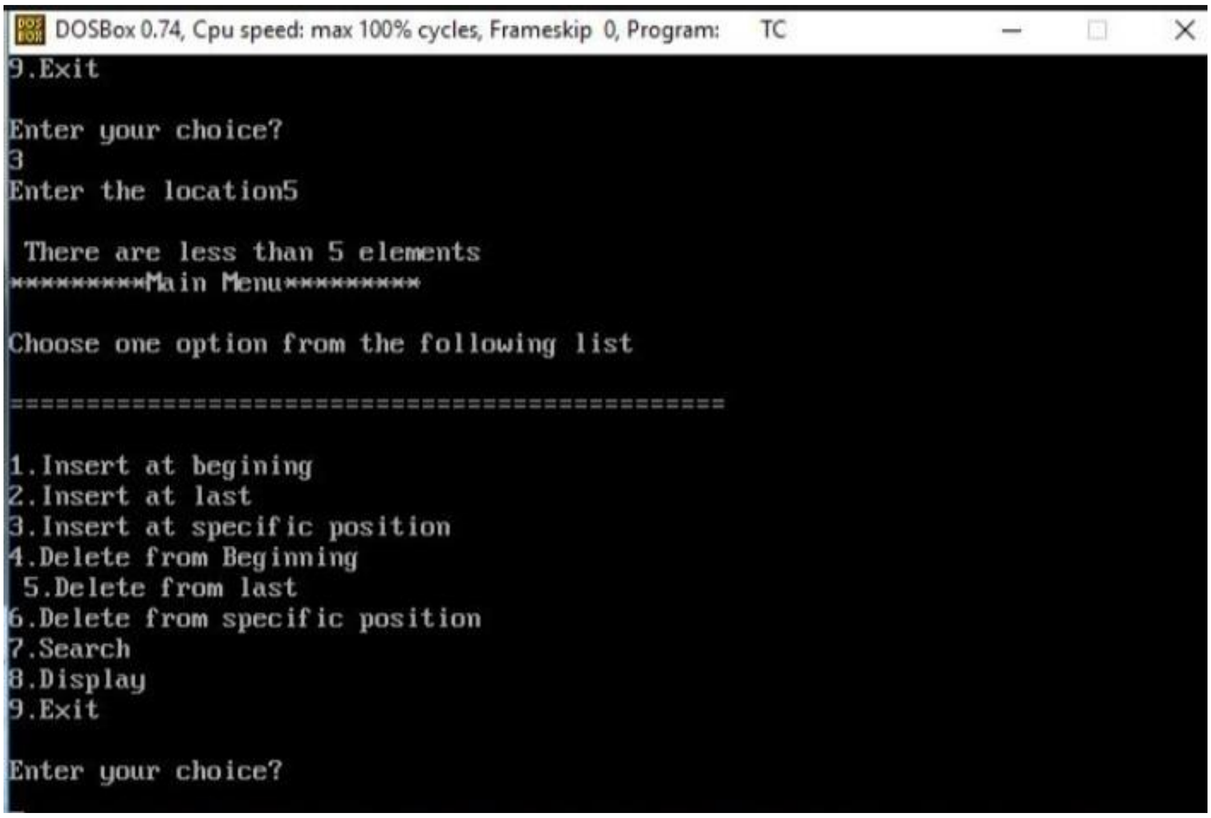
```
}
else if(ptr -> next -> next == NULL)
{
ptr ->next = NULL;
}
else
{
temp = ptr -> next;
ptr -> next = temp -> next;
temp -> next -> prev = ptr;
free(temp);
printf("\ndeletion success\n");
}
}
void display()
{
struct node *ptr;
printf("\n display values\n");
ptr = head;
while(ptr != NULL
{
printf("%d\n",ptr->data);

ptr=ptr->next;


}
```

```
}
void search()
{
struct node *ptr;
int item,i=0,flag;
ptr = head;
if(ptr == NULL)
{
printf("\nEmpty List\n");
}
else
{
printf("\nEnter item to be search?\n");
scanf("%d",&item);
while (ptr!=NULL)
{
if(ptr->data == item)
{
printf("\nitem found at location %d ",i+1); flag=0;
break;
}
else
{
flag=1;
} i++;
ptr = ptr -> next;
}
if(flag==1)
{
printf("\nItem not found\n");
}
}
}
```

OUTPUT



```
Enter your choice?
2

Enter a value14

insertion success

*********Main Menu*********

Choose one option from the following list

===============================================

1.Insert at begining
2.Insert at last
3.Insert at specific position
4.Delete from Beginning
 5.Delete from last
6.Delete from specific position
7.Search
8.Display
9.Exit

Enter your choice?
```


DOSBox 0.74, Cpu speed: max 100% cycles, Frameskip 0, Program: TC

```
9.Exit

Enter your choice?
3
Enter the location5

 There are less than 5 elements
*********Main Menu**********

Choose one option from the following list

===============================================

1.Insert at begining
2.Insert at last
3.Insert at specific position
4.Delete from Beginning
 5.Delete from last
6.Delete from specific position
7.Search
8.Display
9.Exit

Enter your choice?
```