/*linked list all op(insertion at val)*/

```
#include<stdio.h>
#include<stdlib.h>
struct node{
	int data;
	struct node*link;
};
struct node*header;
struct node*create_list(struct node*);
struct node*display(struct node*);
struct node*insert_beg(struct node*);
struct node*insert_end(struct node*);
struct node*insert_any(struct node*);
struct node*delete_beg(struct node*);
struct node*delete_end(struct node*);
struct node*delete_any(struct node*);
void search(struct node*);
struct node*sort_list(struct node*);
int main()
{
	int ch;
	while(ch!=11)
	{
		printf("main menu\n");
		printf("1.create list\n2.display\n3.insert at beg\n4.insert at end\n5.insert at any value\n6.delete at beg\n7.delete at end\n8.delete from any position\n9.search\n10.sort the list\n11.exit\n");
	   printf("enter your choice\n");
	   scanf("%d",&ch);
	   switch(ch)
	   {
```

```
                case 1:header=create_list(header);
                break;
                case 2:header=display(header);
                break;
                case 3:header=insert_beg(header);
                break;
                case 4:header=insert_end(header);
                break;
                case 5:header=insert_any(header);
                break;
                case 6:header=delete_beg(header);
                break;
                case 7:header=delete_end(header);
                break;
                case 8:header=delete_any(header);
                break;
                case 9:search(header);
                break;
                case 10:header=sort_list(header);
                break;
                case 11:exit(0);
                default:
                        printf("invalid choice\n");
                }
        }
}
struct node*create_list(struct node*header)
{
        struct node*new_node,*ptr;
        int item;
        printf("enter -1 for end\n");
```

```
	printf("enter your data:\n");
	scanf("%d",&item);
	while(item!=-1)
	{
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=item;
		if(header==NULL)
		{
			new_node->link=NULL;
			header=new_node;
		}
		else
		{
			ptr=header;
			while(ptr->link!=NULL)
			{
				ptr=ptr->link;
			}
			ptr->link=new_node;
			new_node->link=NULL;
		}
		printf("enter your data:\n");
	scanf("%d",&item);
	}
	printf("list created\n");
	return header;
}
struct node*display(struct node*header)
{
	printf("the list is below\n");
	struct node*ptr;
```

```
	if(header==NULL)
	{
		printf("list empty\n");
	}
	else
	{
		ptr=header;
		while(ptr!=NULL)
		{
			printf("%d\n",ptr->data);
			ptr=ptr->link;
		}
	}
	return header;
}
struct node*insert_beg(struct node*header)
{
	struct node*new_node;
	int item;
	if(header==NULL)   //memory bank returns null
	{
		printf("overflow:insertion not possible\n");
	}
	else
	{
		printf("enter your data to be inserted:\n");
		scanf("%d",&item);
		new_node=(struct node*)malloc(sizeof(struct node));
		new_node->data=item;
		new_node->link=header;
		header=new_node;
```

```
        }
        printf("node inserted at beg\n");
        return header;
}
struct node*insert_end(struct node*header)
{
        struct node*new_node,*ptr;
        int item;
        if(header==NULL)    //memory bank returns null
        {
                printf("overflow:insertion not possible\n");
        }
        else
        {
                printf("enter the data to be inserted:\n");
                scanf("%d",&item);
                new_node=(struct node*)malloc(sizeof(struct node));
                new_node->data=item;
                ptr=header;
                while(ptr->link!=NULL)
                {
                        ptr=ptr->link;
                }
                ptr->link=new_node;
                new_node->link=NULL;
        }
        printf("node inserted at end\n");
        return header;
}
struct node*insert_any(struct node*header)
{
```

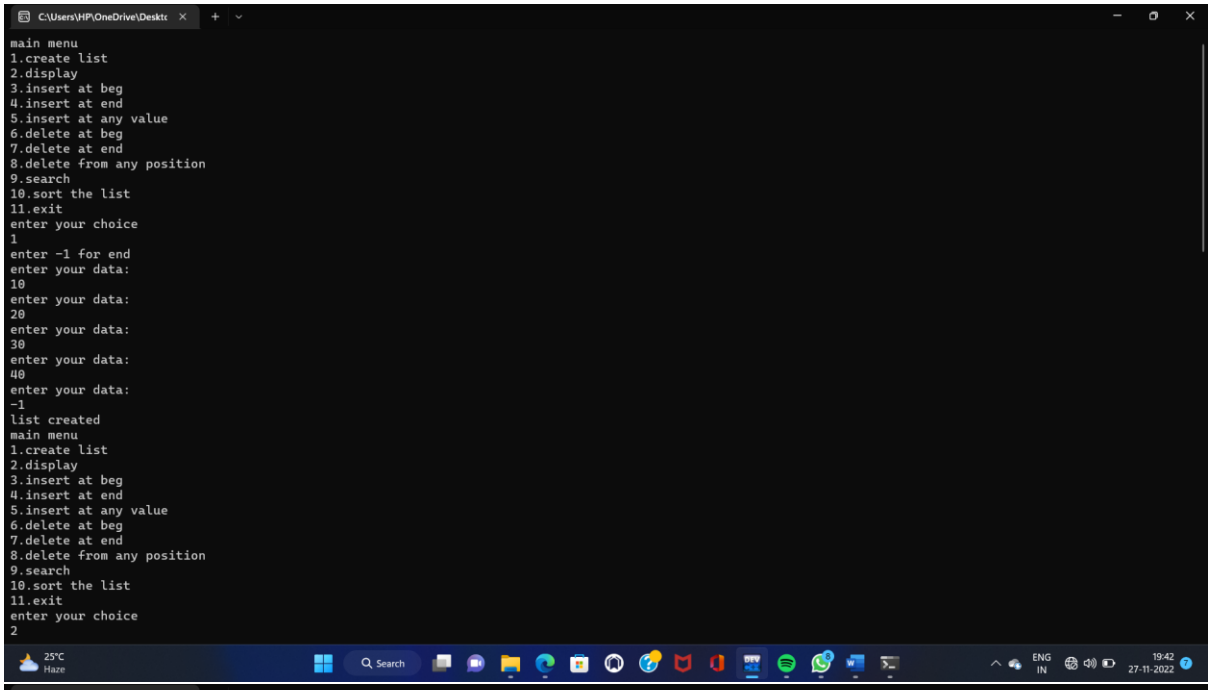
```
        struct node*ptr;
        int i,item,val;
        if(header==NULL)
        {
                printf("overflow:insertion not possible\n");    //memory bank returns null
        }
        else
        {
                printf("enter the value at which you want to insert the node:\n");
                scanf("%d",&val);
                printf("enter the data to be inserted\n");
                scanf("%d",&item);
                ptr=header;
                while(ptr->data!=val)
                {
                        ptr=ptr->link;
                }
                ptr->data=item;
        }
        printf("node inserted at specific pos\n");
        return header;
}
struct node*delete_beg(struct node*header)
{
        struct node*ptr;
        if(header==NULL)
        {
                printf("empty list\n");
        }
        else
        {
```

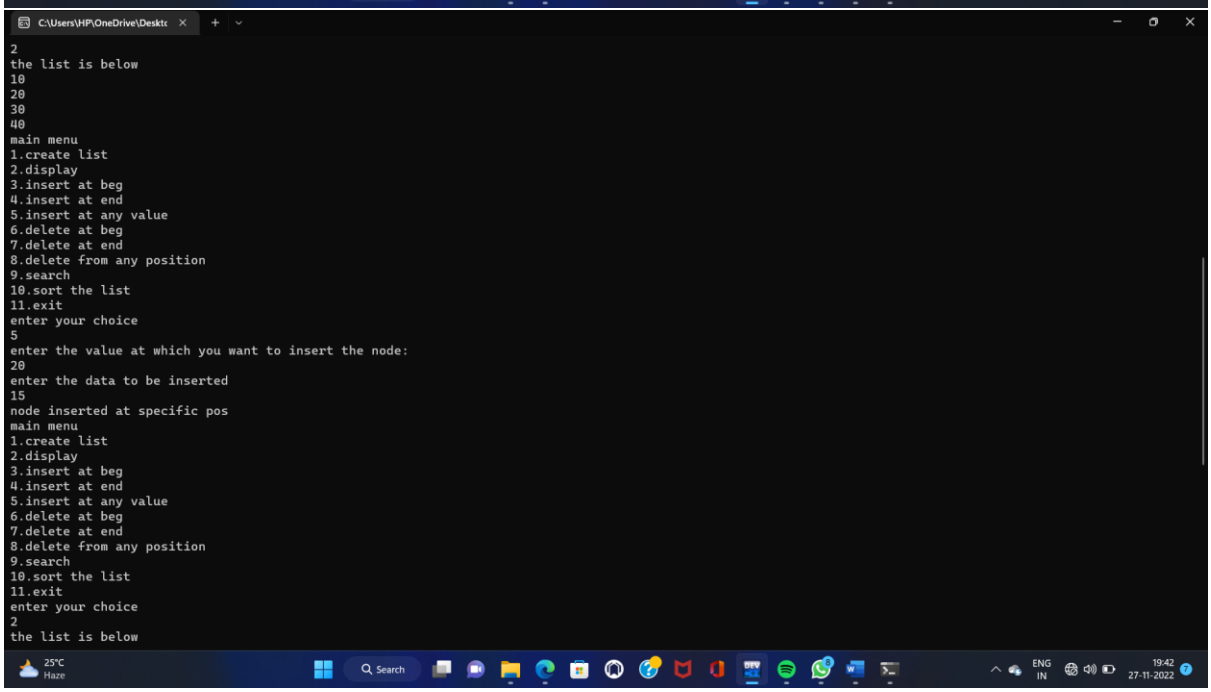
```
                ptr=header;
                header=header->link;
                free(ptr);
        }
        printf("node deleted from beg\n");
        return header;
}
struct node*delete_end(struct node*header)
{
        struct node*ptr,*ptr1;
        if(header==NULL)
        {
                printf("empty list\n");
        }
        else
        {
                ptr=header;
                while(ptr->link!=NULL)
                {
                        ptr1=ptr;
                        ptr=ptr->link;
                }
                ptr1->link=NULL;
                free(ptr);
        }
        printf("node deleted from end\n");
        return header;
}
struct node*delete_any(struct node*header)
{
        struct node*ptr1,*ptr;
```

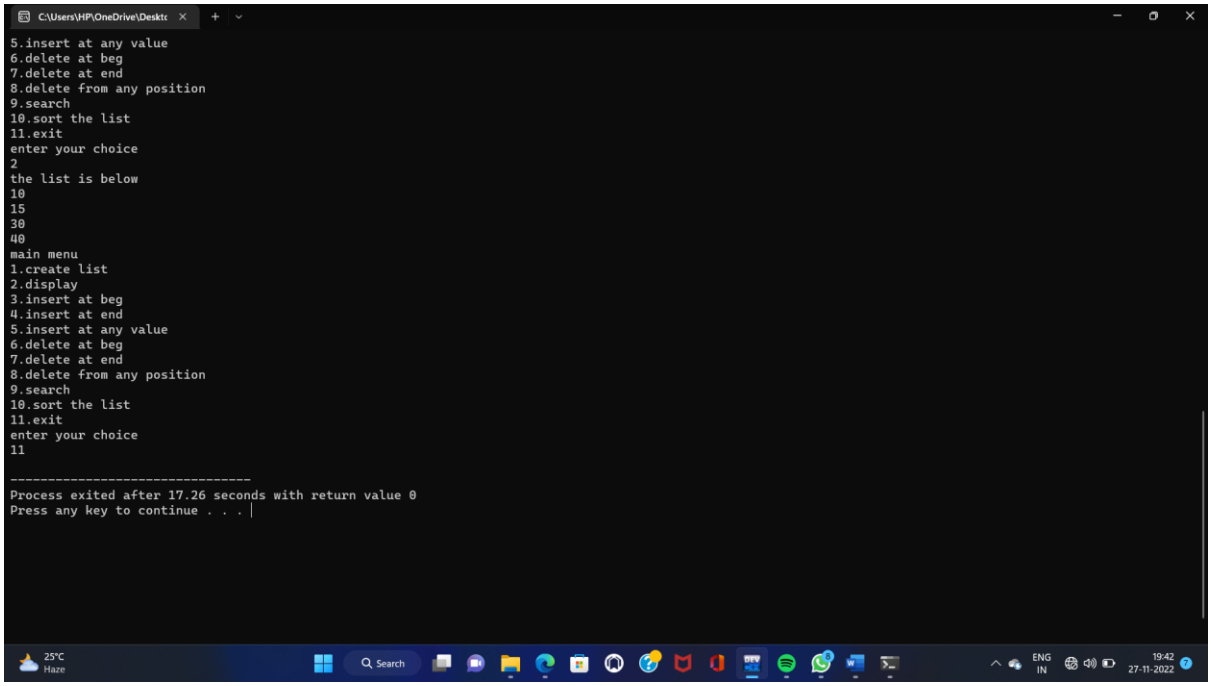
```
	int loc,i;
	if(header==NULL)
	{
		printf("empty list\n");
	}
	else
	{
		printf("enter the location after which you want to delete a node:\n");
		scanf("%d",&loc);
		ptr=header;
		for(i=0;i<=loc;i++)
		{
			ptr1=ptr;
			ptr=ptr->link;
		}
		ptr1->link=ptr->link;
		free(ptr);
	}
	printf("node deleted from the specific position\n");
	return header;
}
void search(struct node*header)
{
	int loc,item,i=0,flag=0;
	struct node*ptr;
	if(header==NULL)
	{
		printf("empty list\n");
	}
	else
	{
```

```
                printf("enter the item to be searched\n");
                scanf("%d",&item);
                ptr=header;
                while(ptr->link!=NULL)
                {
                        if(ptr->data==item)
                        {
                                flag=1;
                                loc=i+1;
                                break;
                        }
                        else
                        {
                                flag=0;
                        }
                        ++i;
                        ptr=ptr->link;
                }
                if(flag==0)
                {
                        printf("element not found\n");
                }
                else
                {
                        printf("element fount at loc %d\n",loc);
                }
        }
}
struct node*sort_list(struct node*header)
{
        struct node*ptr1,*ptr2;
```

```
        int temp;
        if(header==NULL)
        {
                printf("empty list\n");
        }
        else
        {
                ptr1=header;
                while(ptr1->link!=NULL)
                {
                        ptr2=ptr1->link;
                        while(ptr2!=NULL)    //two nodes must present
                        {
                                if(ptr1->data>ptr2->data)
                                {
                                        temp=ptr1->data;
                                        ptr1->data=ptr2->data;
                                        ptr2->data=temp;
                                }
                                ptr2=ptr2->link;
                        }
                        ptr1=ptr1->link;
                }
        }
        printf("list sorted\n");
        return header;
}
```


C:\Users\HP\OneDrive\Desktc
main menu
1.create list
2.display
3.insert at beg
4.insert at end
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
1
enter -1 for end
enter your data:
10
enter your data:
20
enter your data:
30
enter your data:
40
enter your data:
-1
list created
main menu
1.create list
2.display
3.insert at beg
4.insert at end
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
2
25°C
Haze
Search
ENG
IN
19:42
27-11-2022


C:\Users\HP\OneDrive\Desktc
2
the list is below
10
20
30
40
main menu
1.create list
2.display
3.insert at beg
4.insert at end
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
5
enter the value at which you want to insert the node:
20
enter the data to be inserted
15
node inserted at specific pos
main menu
1.create list
2.display
3.insert at beg
4.insert at end
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
2
the list is below
25°C
Haze
Search
ENG
IN
19:42
27-11-2022


C:\Users\HP\OneDrive\Deskto
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
2
the list is below
10
15
30
40
main menu
1.create list
2.display
3.insert at beg
4.insert at end
5.insert at any value
6.delete at beg
7.delete at end
8.delete from any position
9.search
10.sort the list
11.exit
enter your choice
11

Process exited after 17.26 seconds with return value 0
Press any key to continue . . .
25°C
Haze
Search
ENG
IN
19:42
27-11-2022