DS Week-4

Write a program that uses functions to perform the following operations on singly linked list.:

i) Creation

ii) Insertion

iii) Deletion

iv) Traversal

Program:

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
	int data;
	struct node * link;
};

struct node* head=NULL,*tail=NULL,*cur,*prev,*next;
void create()
{
	int n;
	printf("Enter the number of nodes:\n");
	scanf("%d",&n);
	for(int i=0;i<n;i++)
	{
```

```
                cur=(struct node*)malloc(sizeof(struct node));
                printf("enter current node data:");
                scanf("%d",&(cur->data));
                cur->link=NULL;
                if(head==NULL)
                {
                        head=tail=cur;
                }
                else
                {
                        tail->link=cur;
                        tail=cur;
                }
        }
}

//insert at begin
void insert_at_begin()
{
        cur=(struct node*)malloc(sizeof(struct node));
        printf("Enter the cur element");
        scanf("%d",&(cur->data));
        cur->link=head;
        head=cur;
```

```
}

//insert at end
void insert_at_end()
{
        cur=(struct node*)malloc(sizeof(struct node));
        printf("Enter data");
        scanf("%d",&(cur->data));
        cur->link=NULL;
        tail->link=cur;
        tail=cur;
}

//insert at position
void insert_at_a_position()
{
        int pos,c=1;
        cur=(struct node*)malloc(sizeof(struct node));
        printf("Enter the cur data element: \n");
        scanf("%d",&(cur->data));
        printf("Enter the pos to insert:\n");
        scanf("%d",&pos);
        next=head;
        while(c<pos)
        {
                prev=next;
                next=next->link;
                c++;
        }
        prev->link=cur;
        cur->link=next;
}

//insert before
void insert_before()
{
        int value;
        cur=(struct node*)malloc(sizeof(struct node));
        printf("Enter the element to be inserted:\n");
        scanf("%d",&(cur->data));
        printf("Enter data to insert before");
        scanf("%d",&value);
        next=head;
        while(next->data!=value && next!=NULL)
        {
                prev=next;
                next=next->link;
        }
        prev->link=cur;

        cur->link=next;

}


//insert after

void insert_after()

{

        int value;

        cur=(struct node*)malloc(sizeof(struct node));

        printf("Enter the cur value to be inserted:\n");

        scanf("%d",&cur->data);

        printf("Enter after which node we need to perform insertion\n");

        scanf("%d",&value);

        next=head;

        while(next->data!=value && next!=NULL)

        {

                next=next->link;

        }

        cur->link=next->link;

        next->link=cur;

}


//deletion at the beginning of list

void delete_at_begin()
```

```
{
        cur=head;
        head=cur->link;
        cur->link=NULL;
        printf("Deleted element is %d\n",cur->link);
        free(cur);
}

//deletion at the ending of list
void delete_at_end()
{
        cur=head;
        while(cur->link!=tail)
        {
                cur=cur->link;
        }
        cur->link=NULL;
        next=tail;
        printf("Deleted element is %d\n",next->data);
        free(next);
        tail=cur;
}

//deletion at a position of list
```

```
void delete_at_position()
{
        int pos,c=1;
        printf("Enter position of deletion");
        scanf("%d",&pos);
        next=head;
        while(c<pos)
        {
                prev=next;
                next=next->link;
                c++;
        }
        prev->link=next->link;
        printf("Deleted element is %d\n",next->data);
        next->link=NULL;
        free(next);
}

//deletion before a given node
void delete_before_node()
{
        int value;
        printf("Enter before which node we need to delete");
        scanf("%d",&value);
        next=head;
        while(next->link->data!=value)
        {
                prev=next;
                next=next->link;
        }
        prev->link=next->link;
        next->link=NULL;
        printf("Deleted element is %d\n",next->data);
        free(next);
}

//deletion after a given node
void delete_after_node()
{
        int value;
        printf("Enter the value after which node we  need to delete\n");
        scanf("%d",&value);
        next=head;
        while(next->data!=value)
        {
                prev=next;
                next=next->link;
        }
        prev=next->link;

        next->link=prev->link;

        printf("Deleted data is %d\n",prev->data);

        prev->link=NULL;

        free(prev);

}


//traversal of a single linked list

void traversal()

{

        if(head==NULL)

        printf("List is empty");

        else

        {

                next=head;

        }

        while(next!=NULL)

        {

                printf("%d*->",next->data);

                next=next->link;

        }

        printf("NULL\n");

}
```

```
void reverse(struct node*head)
{
        if(head!=NULL)
        {
                reverse(head->link);
                printf("%d  ",head->data);
        }
}
void search_for_element()
{
        int value,flag=0,c=0;
        printf("Enter value to be searched:");
        scanf("%d",&value);
        next=head;
        while(next!=NULL)
        {
                if(next->data==value)
                {
                        flag=1;
                        break;
                }
                next=next->link;
                c++;
        }
        if(flag==1)

        {

                printf("Element present in the list at %d position",c+1);

        }

        else

                printf("Element not present in the list");

}


void sorting()

{

        struct node*p1,*last=NULL;

        int i,c,t;

        do

        {

        c=0;

        p1=head;

        while(p1->link!=last)

        {

                if(p1->data>p1->link->data)

                {

                        t=p1->data;

                        p1->data=p1->link->data;

                        p1->link->data=t;

                }
        p1=p1->link;
        }
        last=p1;
        }while(c);

}

int main()
{
        int ch;
        while(1)
        {
                printf("program for single linked list\n");
                printf("1-create\n2-insert at begin\n3-insert at end\n4-insert at position\n5-insert
before");
                printf("\n6-insert after\n7-delete at begin\n8-delete at end\n9-delete at
pos\n10-delete before\n");
                printf("\n11-delete after\n12-traversal\n13-display in
reverse\n14-search\n15-sort\n");
                printf("enter your choice\n");
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1:create();
                        break;
```

```
case 2:insert_at_begin();

break;

case 3:insert_at_end();

break;

case 4:insert_at_a_position();

break;

case 5:insert_before();

break;

case 6:insert_after();

break;

case 7:delete_at_begin();

break;

case 8:delete_at_end();

break;

case 9:delete_at_position();

break;

case 10:delete_before_node();

break;

case 11:delete_after_node();

break;

case 12:traversal();

break;

case 13:reverse(head);

break;
```

```
                              case 14:search_for_element();
                              break;
                              case 15:sorting();
                              break;
                              case 16:exit(0);
                    }
          }
}
```

OUTPUT:

```
C:\TDM-GCC-64\dslab>gcc sllfunctions.c -o sllfunctions

C:\TDM-GCC-64\dslab>sllfunctions
program for single linked list
1-create
2-insert at begin
3-insert at end
4-insert at position
5-insert before
6-insert after
```

```
7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

1

Enter the number of nodes:

3

enter current node data:12

enter current node data:32

enter current node data:25

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after
```

```
7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->32*->25*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before
```

```
11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

2

Enter the cur element:11

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse
```

```
14-search

15-sort

enter your choice

12

11*->12*->32*->25*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

3
```

```
Enter data22

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

11*->12*->32*->25*->22*->NULL

program for single linked list

1-create

2-insert at begin
```

```
3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

4

Enter the cur data element:

15

Enter the pos to insert:

3

program for single linked list

1-create

2-insert at begin

3-insert at end
```

```
4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

11*->12*->15*->32*->25*->22*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin
```

```
8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

5

Enter the element to be inserted:

25

Enter data to insert before15

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos
```

```
10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

11*->12*->25*->15*->32*->25*->22*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal
```

```
13-display in reverse

14-search

15-sort

enter your choice

6

Enter the cur value to be inserted:

33

Enter after which node we need to perform insertion

15

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse
```

```
14-search

15-sort

enter your choice

12

11*->12*->25*->15*->33*->32*->25*->22*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

7
```

```
Deleted element is 0

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->25*->15*->33*->32*->25*->22*->NULL

program for single linked list

1-create

2-insert at begin
```

```
3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

8

Deleted element is 22

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after
```

```
7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->25*->15*->33*->32*->25*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before
```

```
11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

9

Enter position of deletion4

Deleted element is 33

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal
```

```
13-display in reverse

14-search

15-sort

enter your choice

12

12*->25*->15*->32*->25*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice
```

```
10

Enter before which node we need to delete15

Deleted element is 25

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->15*->32*->25*->NULL

program for single linked list
```

```
1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

11

Enter the value after which node we  need to delete

15

Deleted data is 32

program for single linked list

1-create

2-insert at begin
```

```
3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->15*->25*->NULL

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after
```

```
7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

13

25  15  12  program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before
```

```
11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

14

Enter value to be searched:15

Element present in the list at 2 position for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search
```

```
15-sort

enter your choice

15

program for single linked list

1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

12

12*->15*->25*->NULL

program for single linked list
```

```
1-create

2-insert at begin

3-insert at end

4-insert at position

5-insert before

6-insert after

7-delete at begin

8-delete at end

9-delete at pos

10-delete before


11-delete after

12-traversal

13-display in reverse

14-search

15-sort

enter your choice

16
```

Hi WHAA,

As of now, you have a(n) B in the class. This assignment is worth 15.00 points. If you get more than 14.50 (97%) on this assignment, your class grade will increase to a(n) A. If you get less than 7.00 (47%) on this assignment, your grade will drop at least one grade. Not doing this assignment will result in a(n) C.