Name: Aditi Kohale

Course: C, DSA and C++

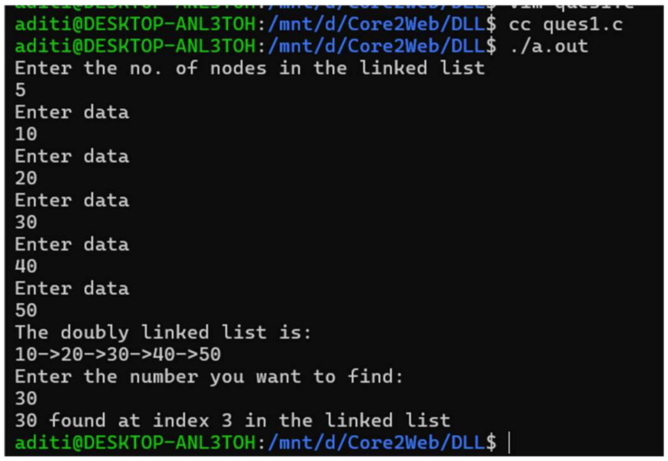
Topic: Linked List – Assignment 4 – Doubly Linked List

Q.1. Find the first occurrence of a number:

Code:

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
        struct node *prev;
        int data;
        struct node *next;
};

struct node *head;

int find_occ(int num)
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
                return -1;
        }
        else
        {
                int index=0;
                struct node *temp=head;
                while(temp!=NULL)
                {
                        index++;
                        if(temp->data==num)
                        {
                                return index;
                        }
                        else
                        {
                                temp=temp->next;
                        }

                }
        }
}

struct node *createNode()
{
        struct node *newNode=(struct node*)malloc(sizeof(struct node));
        printf("Enter data\n");
        scanf("%d",&newNode->data);
        newNode->prev=NULL;
        newNode->next=NULL;
        return newNode;
}

void addNode()
{
        struct node *newNode=createNode();
        if(head==NULL)
        {
                head=newNode;
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=newNode;
                newNode->prev=temp;
        }
}

void printLL()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        printf("%d->",temp->data);
                        temp=temp->next;
                }
                printf("%d\n",temp->data);
        }
}

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();
        int num;
        printf("Enter the number you want to find:\n");
        scanf("%d",&num);
        int occ=find_occ(num);
        if(occ==-1)
        {
                printf("No such number found\n");
        }
        else
        {
                printf("%d found at index %d in the linked list\n",num,occ);
        }
}
```

Output:



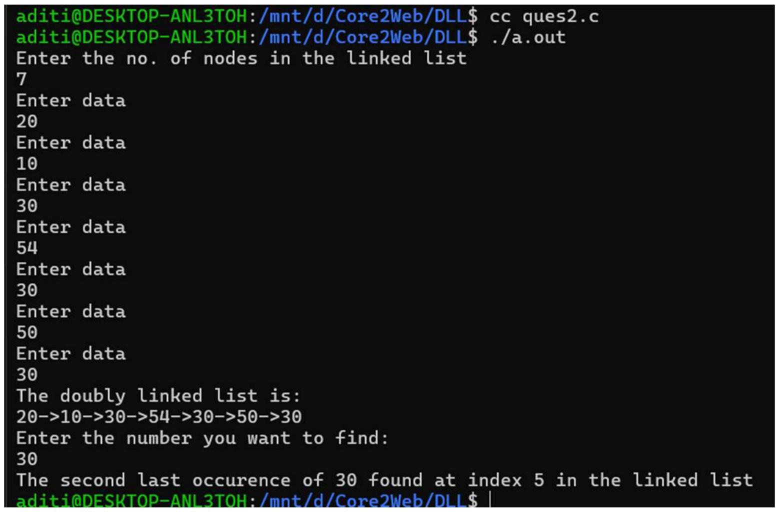
```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ cc ques1.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
5
Enter data
10
Enter data
20
Enter data
30
Enter data
40
Enter data
50
The doubly linked list is:
10->20->30->40->50
Enter the number you want to find:
30
30 found at index 3 in the linked list
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ |
```

Q.2. Find the second occurrence of a number:

Code:

```
#include<stdio.h>

#include<stdlib.h>


struct node

{

        struct node *prev;

        int data;

        struct node *next;

};


struct node *head;


int find_occ(int num)

{

        if(head==NULL)

        {
                printf("Linked list is empty\n");
                return -1;
        }
        else
        {
                int last=-1;
                int second_last=-1;
                int index=1;
                struct node *temp=head;
                while(temp!=NULL)
                {
                        if(temp->data==num)
                        {
                                second_last=last;
                                last=index;
                        }
                        index++;
                        temp=temp->next;

                }
                return second_last;
        }
}

struct node *createNode()
{
        struct node *newNode=(struct node*)malloc(sizeof(struct node));
        printf("Enter data\n");
        scanf("%d",&newNode->data);
        newNode->prev=NULL;

        newNode->next=NULL;

        return newNode;

}


void addNode()

{

        struct node *newNode=createNode();

        if(head==NULL)

        {

                head=newNode;

        }

        else

        {

                struct node *temp=head;

                while(temp->next!=NULL)

                {

                        temp=temp->next;

                }

                temp->next=newNode;

                newNode->prev=temp;

        }

}


void printLL()

{

        if(head==NULL)

        {

                printf("Linked list is empty\n");
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        printf("%d->",temp->data);
                        temp=temp->next;
                }
                printf("%d\n",temp->data);
        }
}

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();
        int num;
        printf("Enter the number you want to find:\n");
        scanf("%d",&num);
        int occ=find_occ(num);
        if(occ==-1)
        {
                printf("No such number found\n");
        }
        else
        {
                printf("The second last occurence of %d found at index %d in the linked list\n",num,occ);
        }
}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ cc ques2.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
7
Enter data
20
Enter data
10
Enter data
30
Enter data
54
Enter data
30
Enter data
50
Enter data
30
The doubly linked list is:
20->10->30->54->30->50->30
Enter the number you want to find:
30
The second last occurence of 30 found at index 5 in the linked list
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ |
```

Q.3. WAP that searches the occurences of a particular element from the linked list and return the count:

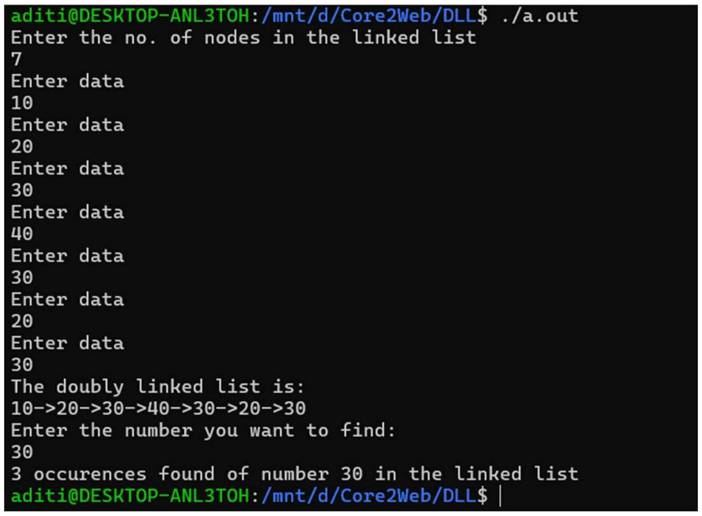
Code:

```
#include<stdio.h>

#include<stdlib.h>


struct node

{

        struct node *prev;
        int data;

        struct node *next;

};


struct node *head;


int find_occ(int num)

{

        if(head==NULL)

        {

                printf("Linked list is empty\n");

                return -1;

        }

        else

        {

                int count=0;

                struct node *temp=head;

                while(temp!=NULL)

                {

                        if(temp->data==num)

                        {

                                count++;

                        }

                        temp=temp->next;

                }

                return count;

        }


}
```

```
struct node *createNode()
{
	struct node *newNode=(struct node*)malloc(sizeof(struct node));
	printf("Enter data\n");
	scanf("%d",&newNode->data);
	newNode->prev=NULL;
	newNode->next=NULL;
	return newNode;
}

void addNode()
{
	struct node *newNode=createNode();
	if(head==NULL)
	{
		head=newNode;
	}
	else
	{
		struct node *temp=head;
		while(temp->next!=NULL)
		{
			temp=temp->next;
		}
		temp->next=newNode;
		newNode->prev=temp;
	}
}
```

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

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();

        int num;

        printf("Enter the number you want to find:\n");

        scanf("%d",&num);

        int occ=find_occ(num);

        if(occ==-1)

        {

                printf("No such number found\n");

        }

        else

        {

                printf("%d occurences found of number %d in the linked list\n",occ,num);

        }

}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
7
Enter data
10
Enter data
20
Enter data
30
Enter data
40
Enter data
30
Enter data
20
Enter data
30
The doubly linked list is:
10->20->30->40->30->20->30
Enter the number you want to find:
30
3 occurences found of number 30 in the linked list
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$
```

Q.4. WAP that adds the sum of the digits of the data in the doubly linked list:

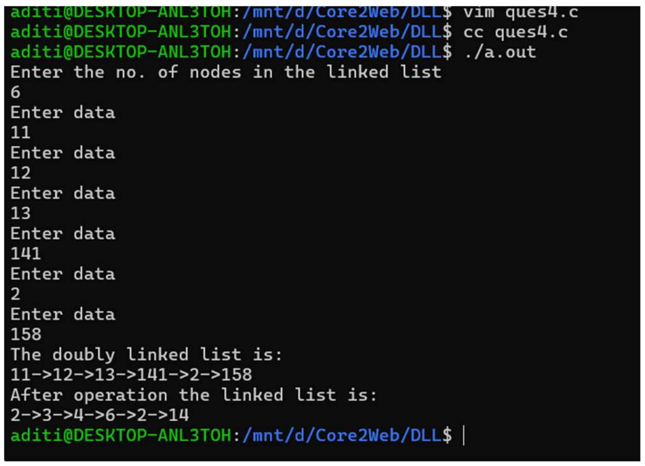
Code:

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
	struct node *prev;
	int data;
	struct node *next;
};

struct node *head;

int func_DLL()
{
	if(head==NULL)
	{
		printf("Linked list is empty\n");
		return -1;
	}
	else
	{
		struct node *temp=head;
		while(temp!=NULL)
		{
			int data=temp->data;
			int sum=0;
```

```
				int last_digit=0;
				while(data!=0)
				{
					last_digit=data%10;
					sum=sum+last_digit;
					data=data/10;
				}
				temp->data=sum;
				temp=temp->next;
			}
			return 0;
		}
}



struct node *createNode()
{
		struct node *newNode=(struct node*)malloc(sizeof(struct node));
		printf("Enter data\n");
		scanf("%d",&newNode->data);
		newNode->prev=NULL;
		newNode->next=NULL;
		return newNode;
}

void addNode()
{
		struct node *newNode=createNode();
```

```
        if(head==NULL)
        {
                head=newNode;
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=newNode;
                newNode->prev=temp;
        }
}

void printLL()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        printf("%d->",temp->data);
                        temp=temp->next;
                }
                printf("%d\n",temp->data);
        }
}

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();
        func_DLL();
        printf("After operation the linked list is:\n");
        printLL();
}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ vim ques4.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ cc ques4.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
6
Enter data
11
Enter data
12
Enter data
13
Enter data
141
Enter data
2
Enter data
158
The doubly linked list is:
11->12->13->141->2->158
After operation the linked list is:
2->3->4->6->2->14
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$
```

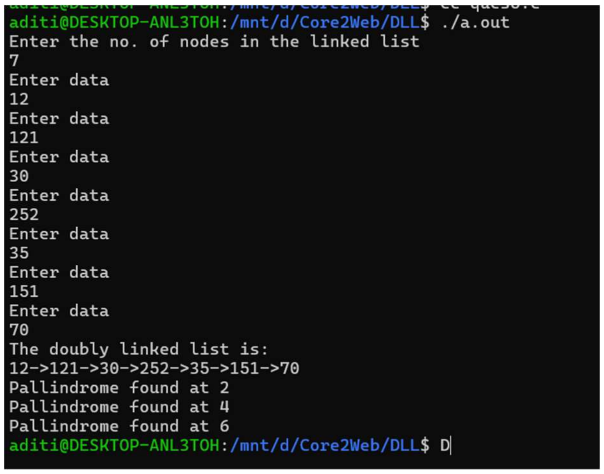
Q.5. WAP that searches all the palindrome data in the linked list:

Code:

```
#include<stdio.h>
#include<stdlib.h>

struct node
{
	struct node *prev;
	int data;
	struct node *next;
};

struct node *head;

int pallindrome_DLL()
{
	if(head==NULL)
	{
		printf("Linked list is empty\n");
		return -1;
	}
	else
	{
		struct node *temp=head;
		int count=1;
		while(temp!=NULL)
		{
			int data=temp->data;
```

```
                        int rev=0;
                        int rem=0;
                        while(data!=0)
                        {
                                rem=data%10;
                                rev=rem+(rev*10);
                                data=data/10;
                        }
                        if(rev==temp->data)
                        {
                                printf("Pallindrome found at %d\n",count);
                        }

                        count++;
                        temp=temp->next;
                }
                return 0;
        }
}




struct node *createNode()
{
        struct node *newNode=(struct node*)malloc(sizeof(struct node));
        printf("Enter data\n");
        scanf("%d",&newNode->data);
        newNode->prev=NULL;
        newNode->next=NULL;
        return newNode;
}

void addNode()
{
        struct node *newNode=createNode();
        if(head==NULL)
        {
                head=newNode;
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=newNode;
                newNode->prev=temp;
        }
}

void printLL()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        printf("%d->",temp->data);
                        temp=temp->next;
                }
                printf("%d\n",temp->data);
        }
}

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();
        pallindrome_DLL();
}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
7
Enter data
12
Enter data
121
Enter data
30
Enter data
252
Enter data
35
Enter data
151
Enter data
70
The doubly linked list is:
12->121->30->252->35->151->70
Pallindrome found at 2
Pallindrome found at 4
Pallindrome found at 6
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ D
```

Q.6. WAP that takes a doubly linked list from the user and take a number from the user and print the names of the length of that number:

Code:

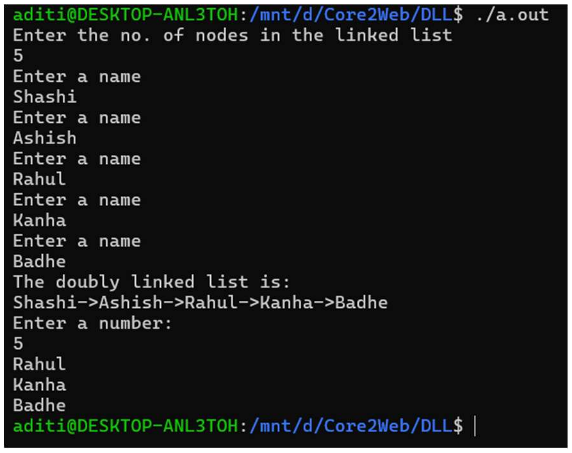
```
#include<stdio.h>

#include<stdlib.h>


struct node

{

            struct node *prev;

       char str[20];

            struct node *next;

};


struct node *head;


int mystrlen(char *str)

{

            int len=0;

            while(*str!='\0')
```

```
        {
                len++;
                str++;
        }

        return len;
}

int check(int n)
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
                return -1;
        }
        else
        {
                struct node *temp=head;
                while(temp!=NULL)
                {
                        int len=mystrlen(temp->str);
                        if(len==n)
                        {
                                printf("%s\n",temp->str);
                        }
                        temp=temp->next;
                }

        }
```

```
}




struct node *createNode()
{
        struct node *newNode=(struct node*)malloc(sizeof(struct node));
        printf("Enter a name\n");
        fgets(newNode->str,15,stdin);
        int len=mystrlen(newNode->str);
        if((*newNode).str[len-1]=='\n')
        {
                (*newNode).str[len-1]='\0';
        }
        newNode->prev=NULL;
        newNode->next=NULL;
        return newNode;
}

void addNode()
{
        struct node *newNode=createNode();
        if(head==NULL)
        {
                head=newNode;
        }
        else
        {
                struct node *temp=head;
```

```
                    while(temp->next!=NULL)
                    {
                              temp=temp->next;
                    }
                    temp->next=newNode;
                    newNode->prev=temp;
          }
}

void printLL()
{
          if(head==NULL)
          {
                    printf("Linked list is empty\n");
          }
          else
          {
                    struct node *temp=head;
                    while(temp->next!=NULL)
                    {
                              printf("%s->",temp->str);
                              temp=temp->next;
                    }
                    printf("%s\n",temp->str);
          }
}

void main()
{
	int count;

	printf("Enter the no. of nodes in the linked list\n");

	scanf("%d",&count);

	getchar();

	for(int i=1;i<=count;i++)

	{

		addNode();

	}

	printf("The doubly linked list is:\n");

	printLL();

	int num;

	printf("Enter a number:\n");

	scanf("%d",&num);

	check(num);

}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
5
Enter a name
Shashi
Enter a name
Ashish
Enter a name
Rahul
Enter a name
Kanha
Enter a name
Badhe
The doubly linked list is:
Shashi->Ashish->Rahul->Kanha->Badhe
Enter a number:
5
Rahul
Kanha
Badhe
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ 
```

Q.7. Take a doubly linked list from the user and reverse its data elements:

Code:

```
#include<stdio.h>
```

```
#include<stdlib.h>

#include<string.h>


struct node

{

            struct node *prev;

      char str[20];

            struct node *next;

};


struct node *head;


int mystrlen(char *str)

{

            int len=0;

            while(*str!='\0')

            {

                        len++;

                        str++;

            }


            return len;

}


char *mystrrev(char *str)

{

            char *start=str;

            char *temp=str;

            while(*temp!='\0')
```

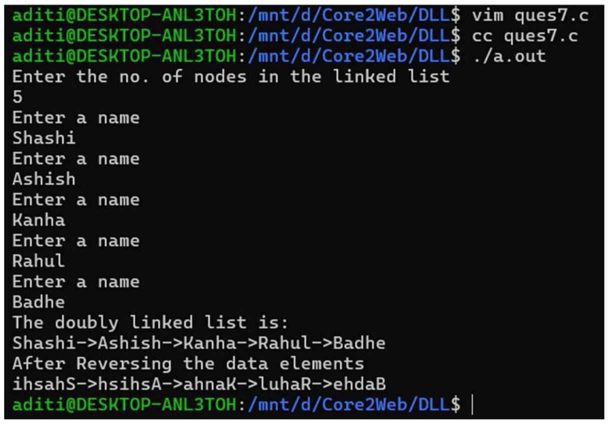
```
        {
                temp++;
        }
        temp--;
        while(start<temp)
        {
                char tempvar=*start;
                *start=*temp;
                *temp=tempvar;
                start++;
                temp--;
        }
        return str;



}

int reverse()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
                return -1;
        }
        else
        {
                struct node *temp=head;
            while(temp!=NULL)
                {
```

```
				strcpy(temp->str,mystrrev(temp->str));
				temp=temp->next;
			}
		}
}



struct node *createNode()
{
		struct node *newNode=(struct node*)malloc(sizeof(struct node));
		printf("Enter a name\n");
		fgets(newNode->str,15,stdin);
		int len=mystrlen(newNode->str);
		if((*newNode).str[len-1]=='\n')
		{
			(*newNode).str[len-1]='\0';
		}
		newNode->prev=NULL;
		newNode->next=NULL;
		return newNode;
}

void addNode()
{
		struct node *newNode=createNode();
		if(head==NULL)
		{
			head=newNode;
```

```
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=newNode;
                newNode->prev=temp;
        }
}

void printLL()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        printf("%s->",temp->str);
                        temp=temp->next;
                }
                printf("%s\n",temp->str);
        }
```

```
}

void main()
{
        int count;
        printf("Enter the no. of nodes in the linked list\n");
        scanf("%d",&count);
        getchar();
        for(int i=1;i<=count;i++)
        {
                addNode();
        }
        printf("The doubly linked list is:\n");
        printLL();
        reverse();
        printf("After Reversing the data elements\n");
        printLL();
}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ vim ques7.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ cc ques7.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
5
Enter a name
Shashi
Enter a name
Ashish
Enter a name
Kanha
Enter a name
Rahul
Enter a name
Badhe
The doubly linked list is:
Shashi->Ashish->Kanha->Rahul->Badhe
After Reversing the data elements
ihsahS->hsihsA->ahnaK->luhaR->ehdaB
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$
```

Q.8. Take a doubly linked list from user, then take a number from the user and keep the elements equal to that length only in the linked list:

Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<string.h>


struct node

{

		struct node *prev;

	char str[20];

		struct node *next;

};


struct node *head;


int mystrlen(char *str)

{

		int len=0;

		while(*str!='\0')

		{

				len++;

				str++;

		}


		return len;

}


int func(int num)
```

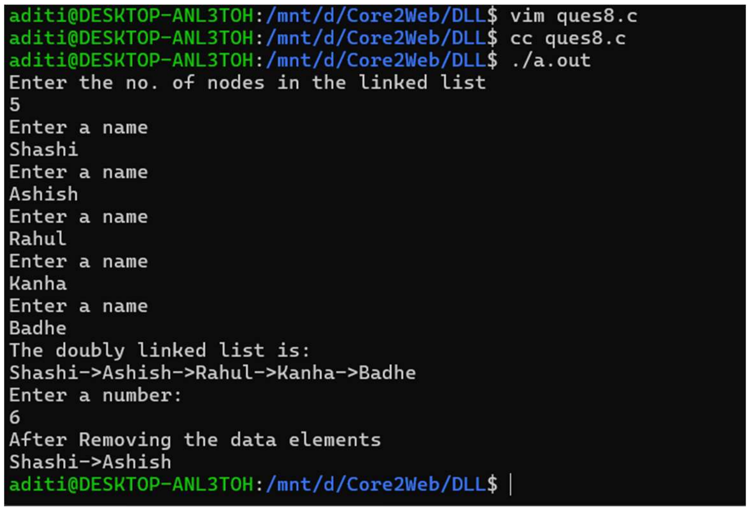
```
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
                return -1;
        }
        else
        {
                struct node *temp=head;
                while(temp!=NULL)
                {
                        if(mystrlen(temp->str)==num)
                        {
                                temp=temp->next;
                        }
                        else
                        {
                                if(temp==head)
                                {
                                        head=temp->next;
                                        free(head->prev);
                                        head->prev=NULL;
                                }
                                else if(temp->next==NULL)
                                {
                                        temp->prev->next=NULL;
                                        free(temp);
                                        temp=NULL;
                                }
```

```
                    else
                    {
                            temp=temp->prev;
                            temp->next=temp->next->next;
                            free(temp->next->prev);
                            temp->next->prev=temp;
                            temp=temp->next;
                    }
                }
            }
            return 0;

        }
}




struct node *createNode()
{
        struct node *newNode=(struct node*)malloc(sizeof(struct node));
        printf("Enter a name\n");
        fgets(newNode->str,15,stdin);
        int len=mystrlen(newNode->str);
        if((*newNode).str[len-1]=='\n')
        {
            (*newNode).str[len-1]='\0';
        }
        newNode->prev=NULL;
        newNode->next=NULL;
        return newNode;
}

void addNode()
{
        struct node *newNode=createNode();
        if(head==NULL)
        {
                head=newNode;
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=newNode;
                newNode->prev=temp;
        }
}

void printLL()
{
        if(head==NULL)
        {
                printf("Linked list is empty\n");
        }
```

```
		else
		{
			struct node *temp=head;
			while(temp->next!=NULL)
			{
				printf("%s->",temp->str);
				temp=temp->next;
			}
			printf("%s\n",temp->str);
		}
}

void main()
{
		int count;
		printf("Enter the no. of nodes in the linked list\n");
		scanf("%d",&count);
		getchar();
		for(int i=1;i<=count;i++)
		{
			addNode();
		}
		printf("The doubly linked list is:\n");
		printLL();
		printf("Enter a number:\n");
		int num;
		scanf("%d",&num);
		func(num);
		printf("After Removing the data elements\n");
```

```
        printLL();
}
```

Output:



```
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ vim ques8.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ cc ques8.c
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ ./a.out
Enter the no. of nodes in the linked list
5
Enter a name
Shashi
Enter a name
Ashish
Enter a name
Rahul
Enter a name
Kanha
Enter a name
Badhe
The doubly linked list is:
Shashi->Ashish->Rahul->Kanha->Badhe
Enter a number:
6
After Removing the data elements
Shashi->Ashish
aditi@DESKTOP-ANL3TOH:/mnt/d/Core2Web/DLL$ |
```