Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Summer , Year:2022), B.Sc. in CSE (Day)

LAB REPORT NO 5

Course Title: Structured Programming Lab

Course Code: CSE 106 Section:PC-213DA

Lab Experiment Name: Linear Search & Binary Search

Student Details

Name	ID
MD Dulal Hossain	213902116

Lab Date :10-07-2022

Submission Date :18-08-2022

Course Teacher's Name :FarhanaAkther Sunny

[For Teachers use only:Don't Write Anything inside this box]

Lab Report Status

Marks: …………………………………	Signature:.....................
Comments:...	Date:..............................

Algorithm

Create node

```
if(start==NULL)
	{
		start=temp;
	}
	else
	{
		ptr=start;
		while(ptr->next!=NULL)
		{
			ptr=ptr->next;
		}
		ptr->next=temp;
	}
```

Insert Begnning

```
if(temp==NULL)
	{
		printf("\nOut of Memory Space:");
		return;
	}
	printf("\nEnter the data value for the node:" );
	scanf("%d",&temp->info);
	temp->next =NULL;
	if(start==NULL)
	{
		start=temp;
	}
	else
	{
		temp->next=start;
		start=temp;
	}
```

Insert End

```
if(temp==NULL)
	{
		printf("\nOut of Memory Space:");
```

```
        return;
    }
    printf("\nEnter the data value for the node:" );
    scanf("%d",&temp->info );
    temp->next =NULL;
    if(start==NULL)
    {
        start=temp;
    }
    else
    {
        ptr=start;
        while(ptr->next !=NULL)
        {
            ptr=ptr->next ;
        }
        ptr->next =temp;
    }
```

Insert Specfication

```
if(temp==NULL)
    {
        printf("\nOut of Memory Space:");
        return;
    }
    printf("\nEnter the position for the new node to be inserted:");
    scanf("%d",&pos);
    printf("\nEnter the data value of the node:");
    scanf("%d",&temp->info) ;

    temp->next=NULL;
    if(pos==0)
    {
        temp->next=start;
        start=temp;
    }
    else
    {
        for(i=0,ptr=start;i<pos-1;i++) { ptr=ptr->next;
            if(ptr==NULL)
            {
                printf("\nPosition not found:[Handle with care]");
```

```
                    return;
                }
            }
            temp->next =ptr->next ;
            ptr->next=temp;
        }
```

Delete Begnning

```
if(ptr==NULL)
        {
            printf("\nList is Empty:");
            return;
        }
        else
        {
            ptr=start;
            start=start->next ;
            printf("\nThe deleted element is :%d",ptr->info);
            free(ptr);
        }
```

Delete End

```
if(start==NULL)
        {
            printf("\nList is Empty:");
            exit(0);
        }
        else if(start->next ==NULL)
        {
            ptr=start;
            start=NULL;
            printf("\nThe deleted element is:%d",ptr->info);
            free(ptr);
        }
        else
        {
            ptr=start;
            while(ptr->next!=NULL)
            {
```

```
            temp=ptr;
            ptr=ptr->next;
        }
        temp->next=NULL;
        printf("\nThe deleted element is:%d",ptr->info);
        free(ptr);
    }
```

Delete Spefication

```
if(start==NULL)
    {
        printf("\nThe List is Empty:");
        exit(0);
    }
    else
    {
        printf("\nEnter the position of the node to be deleted:");
        scanf("%d",&pos);
        if(pos==0)
        {
            ptr=start;
            start=start->next ;
            printf("\nThe deleted element is:%d",ptr->info  );
            free(ptr);
        }
        else
        {
            ptr=start;
            for(i=0;i<pos;i++) { temp=ptr; ptr=ptr->next ;
                if(ptr==NULL)
                {
                    printf("\nPosition not Found:");
                    return;
                }
            }
            temp->next =ptr->next ;
            printf("\nThe deleted element is:%d",ptr->info );
            free(ptr);
        }
    }
```

Search

```
if(start==NULL)
        {
                printf("\nThe List is Empty:");
                exit(0);
        }
       else
        {
                printf("\nEnter the element you want to search :");
                scanf("%d",&pos);

              ptr=start;
             for(i=0;i<pos;i++)
                                           { temp=ptr; ptr=ptr->next ;
                  if(ptr==NULL)
                           {
                       printf("\nPosition Found:");
                       return;
                   } else
                   printf ("\nposition not found");
                }

        }
```

Source Code

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

void create();
void display();
void insert_begin();
void insert_end();
void insert_pos();
void delete_begin();
void delete_end();
void delete_pos();
void search_element ();

struct node
{
        int info;
        struct node *next;
};
struct node *start=NULL;
int main()
{
        int choice;
        while(1){

                printf("\n              MENU                         \n");
                printf("\n 1.Create.");
                printf("\n 2.Display.");
                printf("\n 3.Insert at the beginning.");
                printf("\n 4.Insert at the end.");
                printf("\n 5.Insert at specified position.");
                printf("\n 6.Delete from beginning.");
                printf("\n 7.Delete from the end.");
                printf("\n 8.Delete from specified position.");
                printf("\n 9.Search the Element. ");
                printf("\n 10.Exit.");
                printf("\n--------------------------------------");
                printf("\nEnter your choice:\t");
                scanf("%d",&choice);
                switch(choice)
                {
                        case 1:
                                        create();
                                        break;
                        case 2:
                                        display();
                                        break;
                        case 3:
                                        insert_begin();
                                        break;
                        case 4:
                                        insert_end();
                                        break;
                        case 5:
                                        insert_pos();
                                        break;
                        case 6:
                                        delete_begin();
                                        break;
                        case 7:
                                        delete_end();
                                        break;
                        case 8:
                                        delete_pos();
                                        break;
                                                                        case 9:
                                                                                                     search_element();
                                                                                                     break;
                        case 10:
                                        exit(0);
                                        break;

                        default:
                                        printf("\n Wrong Choice:\n");
                                        break;
                }
        }
        return 0;
}
void create()
{
        struct node *temp,*ptr;
        temp=(struct node *)malloc(sizeof(struct node));
        if(temp==NULL)
        {
                printf("\nOut of Memory Space:");
                exit(0);
        }
        printf("\nEnter the data value for the node:");
        scanf("%d",&temp->info);
        temp->next=NULL;
        if(start==NULL)
        {
                start=temp;
        }
        else
        {
                ptr=start;
                while(ptr->next!=NULL)
                {
                        ptr=ptr->next;
                }
                ptr->next=temp;
        }
}
void display()
{
        struct node *ptr;
        if(start==NULL)
        {
                printf("\nList is empty:");
                return;
        }
        else
        {
                ptr=start;
                printf("\nThe List elements are:");
                while(ptr!=NULL)
                {
                        printf("%d ",ptr->info );
                        ptr=ptr->next ;
                }
        }
}
void insert_begin()
{
        struct node *temp;
        temp=(struct node *)malloc(sizeof(struct node));
        if(temp==NULL)
        {
                printf("\nOut of Memory Space:");
                return;
        }
        printf("\nEnter the data value for the node:" );
        scanf("%d",&temp->info);
        temp->next =NULL;
        if(start==NULL)
        {
                start=temp;
        }
        else
```

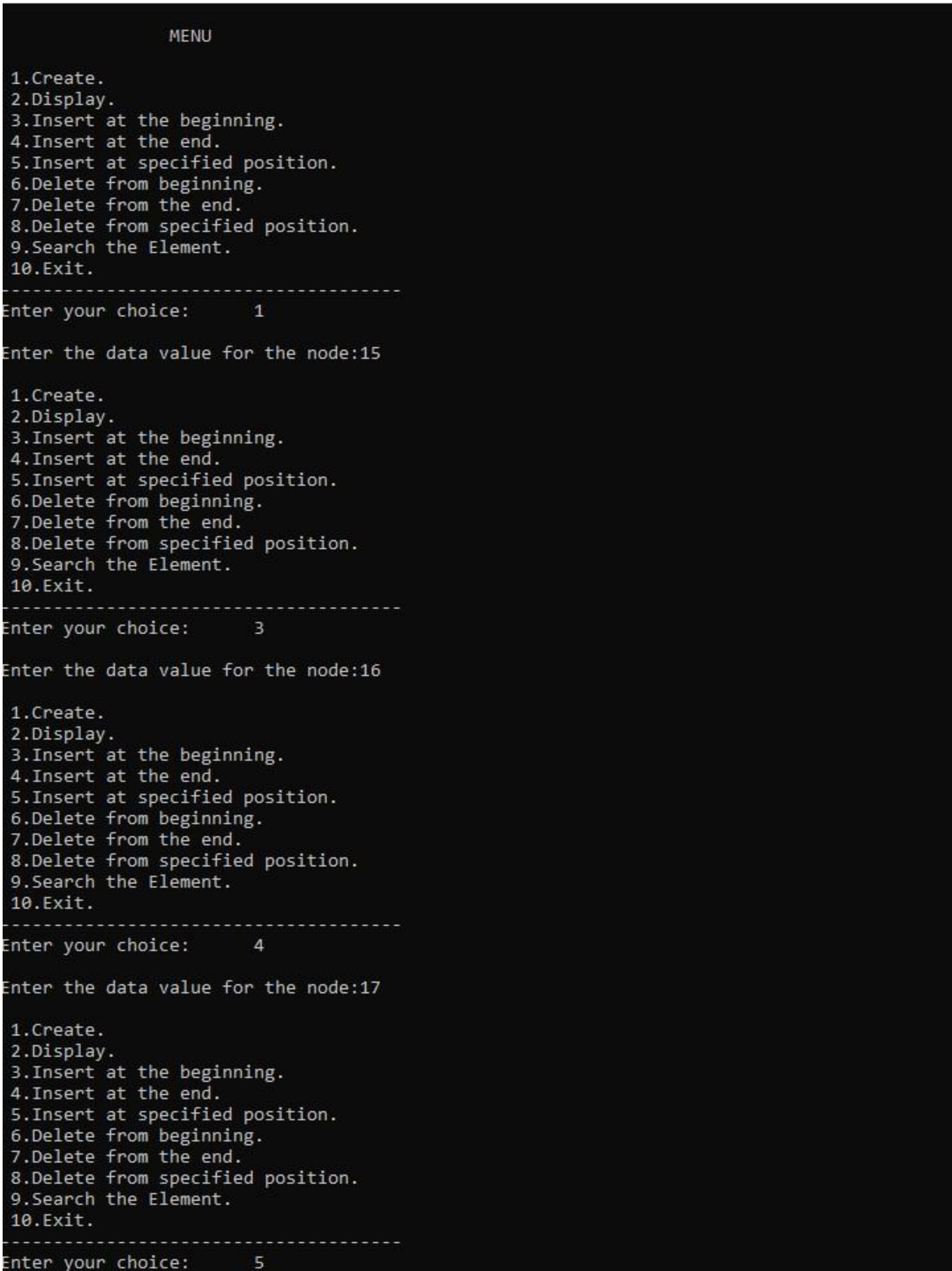
```
	{
		temp->next=start;
		start=temp;
	}
}
void insert_end()
{
	struct node *temp,*ptr;
	temp=(struct node *)malloc(sizeof(struct node));
	if(temp==NULL)
	{
		printf("\nOut of Memory Space:");
		return;
	}
	printf("\nEnter the data value for the node:" );
	scanf("%d",&temp->info );
	temp->next =NULL;
	if(start==NULL)
	{
		start=temp;
	}
	else
	{
		ptr=start;
		while(ptr->next !=NULL)
		{
			ptr=ptr->next ;
		}
		ptr->next =temp;
	}
}
void insert_pos()
{
	struct node *ptr,*temp;
	int i,pos;
	temp=(struct node *)malloc(sizeof(struct node));
	if(temp==NULL)
	{
		printf("\nOut of Memory Space:");
		return;
	}
	printf("\nEnter the position for the new node to be inserted:");
	scanf("%d",&pos);
	printf("\nEnter the data value of the node:");
```

```
        scanf("%d",&temp->info) ;

        temp->next=NULL;
        if(pos==0)
        {
                temp->next=start;
                start=temp;
        }
        else
        {
                for(i=0,ptr=start;i<pos-1;i++) { ptr=ptr->next;
                        if(ptr==NULL)
                        {
                                printf("\nPosition not found:[Handle with care]");
                                return;
                        }
                }
                temp->next =ptr->next ;
                ptr->next=temp;
        }
}
void delete_begin()
{
        struct node *ptr;
        if(ptr==NULL)
        {
                printf("\nList is Empty:");
                return;
        }
        else
        {
                ptr=start;
                start=start->next ;
                printf("\nThe deleted element is :%d",ptr->info);
                free(ptr);
        }
}
void delete_end()
{
        struct node *temp,*ptr;
        if(start==NULL)
        {
                printf("\nList is Empty:");
                exit(0);
```

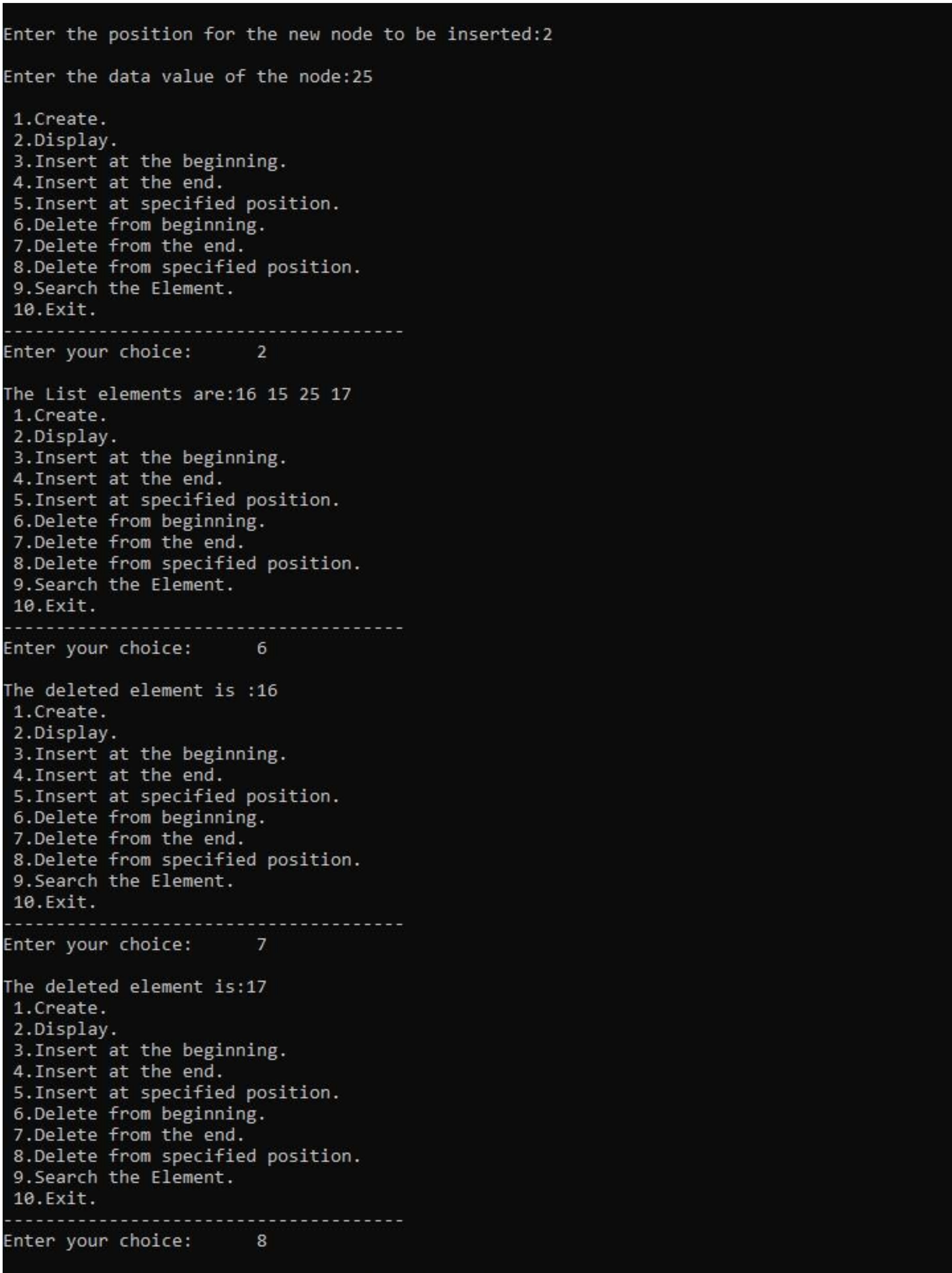
```
	}
	else if(start->next ==NULL)
	{
		ptr=start;
		start=NULL;
		printf("\nThe deleted element is:%d",ptr->info);
		free(ptr);
	}
	else
	{
		ptr=start;
		while(ptr->next!=NULL)
		{
			temp=ptr;
			ptr=ptr->next;
		}
		temp->next=NULL;
		printf("\nThe deleted element is:%d",ptr->info);
		free(ptr);
	}
}
void delete_pos()
{
	int i,pos;
	struct node *temp,*ptr;
	if(start==NULL)
	{
		printf("\nThe List is Empty:");
		exit(0);
	}
	else
	{
		printf("\nEnter the position of the node to be deleted:");
		scanf("%d",&pos);
		if(pos==0)
		{
			ptr=start;
			start=start->next ;
			printf("\nThe deleted element is:%d",ptr->info  );
			free(ptr);
		}
		else
		{
			ptr=start;
```

```
                    for(i=0;i<pos;i++) { temp=ptr; ptr=ptr->next ;
                        if(ptr==NULL)
                        {
                            printf("\nPosition not Found:");
                            return;
                        }
                    }
                    temp->next =ptr->next ;
                    printf("\nThe deleted element is:%d",ptr->info );
                    free(ptr);
                }
        }
}
void search_element ()

{
        int i,pos;
        struct node *temp,*ptr;
        if(start==NULL)
        {
                printf("\nThe List is Empty:");
                exit(0);
        }
       else
        {
                printf("\nEnter the element you want to search :");
                scanf("%d",&pos);

              ptr=start;
              for(i=0;i<pos;i++)
                                        { temp=ptr; ptr=ptr->next ;
                 if(ptr==NULL)
                      {
                    printf("\nPosition Found:");
                    return;
                  } else
                  printf ("\nposition not found");
               }

        }
}
```

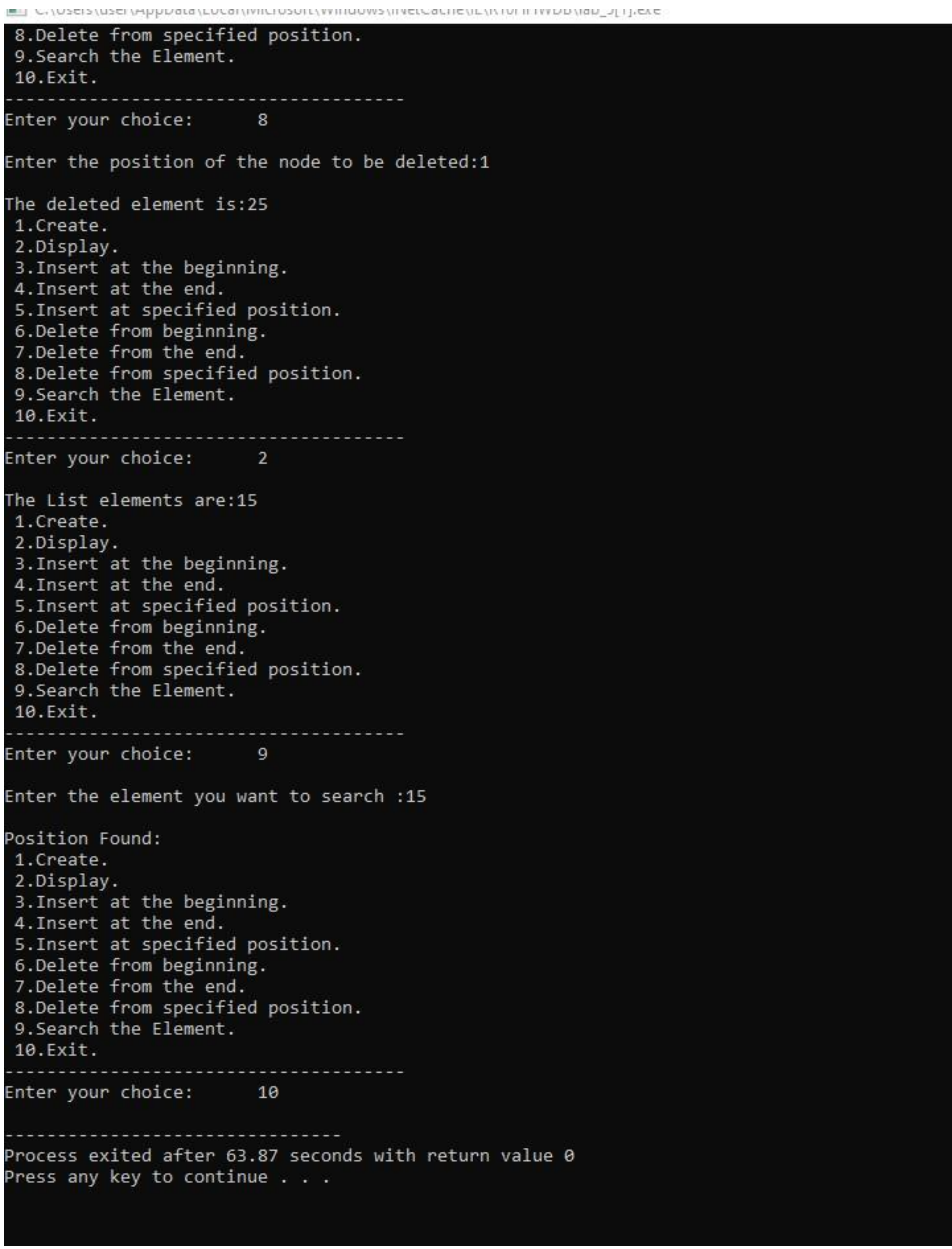
Output



```
                MENU

 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      1

Enter the data value for the node:15

 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      3

Enter the data value for the node:16

 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      4

Enter the data value for the node:17

 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      5
```



```
Enter the position for the new node to be inserted:2

Enter the data value of the node:25

 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      2

The List elements are:16 15 25 17
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      6

The deleted element is :16
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      7

The deleted element is:17
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      8
```



```
C:\Users\user\AppData\Local\Microsoft\Windows\INetCache\IE\[illegible]\lab_5[1].exe
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      8

Enter the position of the node to be deleted:1

The deleted element is:25
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      2

The List elements are:15
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      9

Enter the element you want to search :15

Position Found:
 1.Create.
 2.Display.
 3.Insert at the beginning.
 4.Insert at the end.
 5.Insert at specified position.
 6.Delete from beginning.
 7.Delete from the end.
 8.Delete from specified position.
 9.Search the Element.
 10.Exit.
--------------------------------------
Enter your choice:      10

--------------------------------
Process exited after 63.87 seconds with return value 0
Press any key to continue . . .
```