Linked list(deletion):

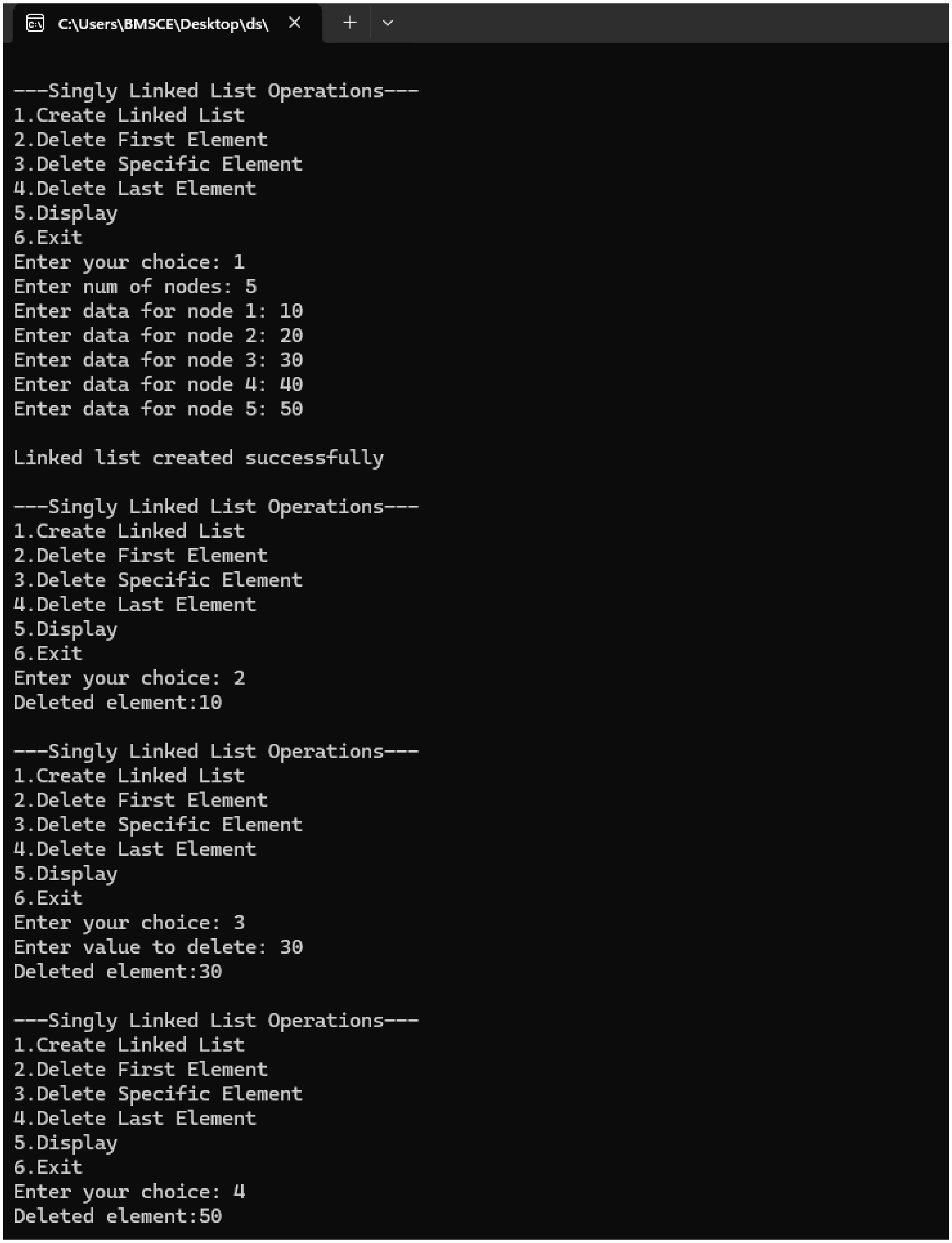
```
#include<stdio.h>

#include<stdlib.h>

struct Node{

int data;

struct Node *next;

};

struct Node *head=NULL;

void createList(int n){

struct Node *newNode,*temp;

int data,i;

if(n<=0){printf("Number of nodes should be greater than 0\n");

return ;}

for(i=1;i<=n;i++)

{

   newNode=(struct Node *)malloc(sizeof(struct Node));

   if(newNode==NULL){

      printf("Memory allocation failed\n");

      return;

   }

   printf("Enter data for node %d: ",i);

   scanf("%d",&data);

   newNode ->data=data;

   newNode->next=NULL;

   if(head==NULL){

      head=newNode;

   }
```

```
    else{
        temp->next=newNode;
    }
    temp=newNode;
    }
    printf("\nLinked list created successfully\n");
}
void deleteFirst(){
struct Node *temp;
if(head==NULL){
    printf("List is empty.Nothing to delete.\n");
    return;
}
temp=head;
head=head->next;
printf("Deleted element:%d\n",temp->data);
free(temp);
}

void deleteLast(){
struct Node *temp,*prev;
if(head==NULL){
    printf("List is empty .Nothing to delete\n");
    return;
}
temp=head;
while(temp->next!=NULL){
    prev=temp;
    temp=temp->next;
```

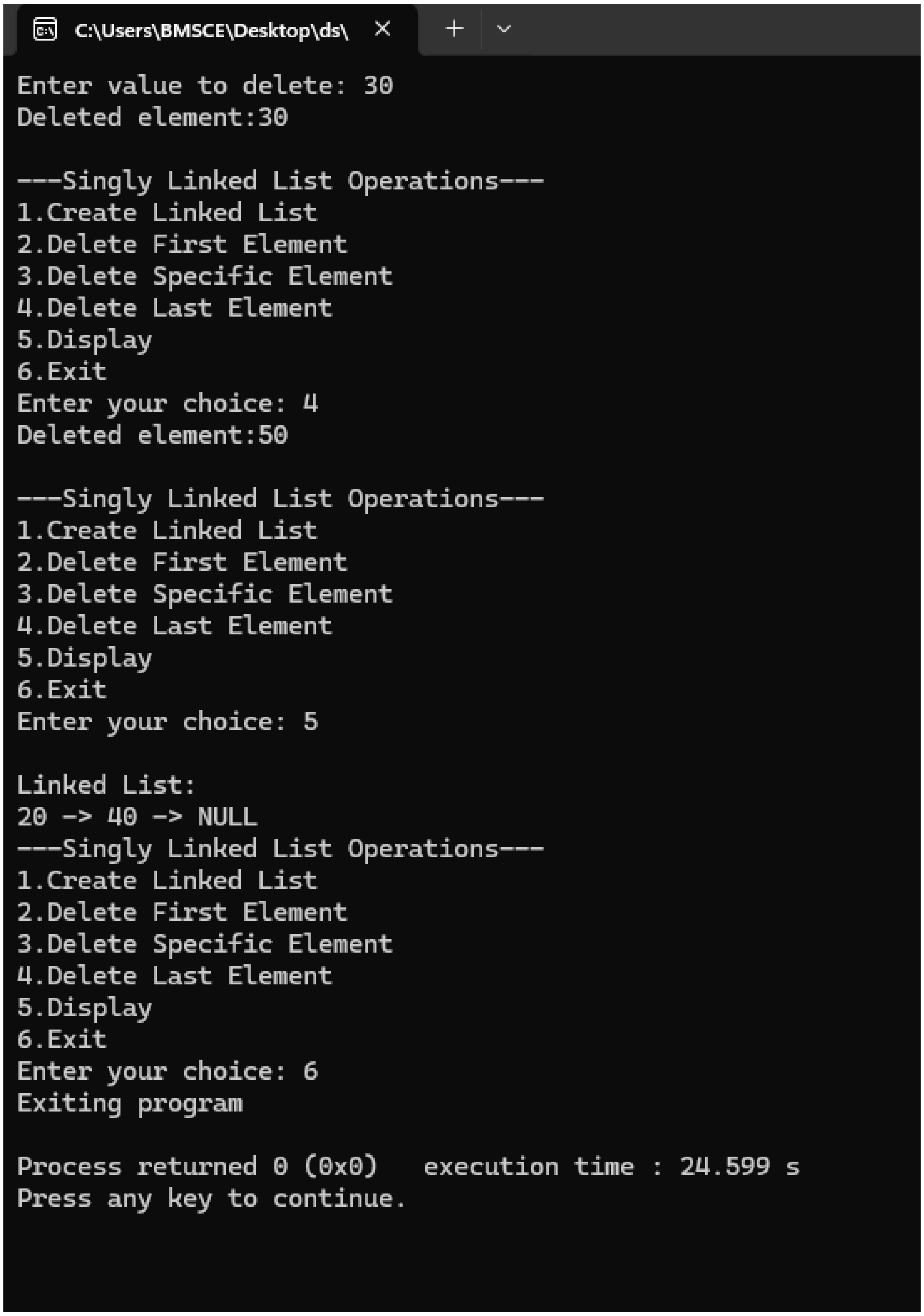
```
}
printf("Deleted element:%d\n",temp->data);
prev->next=NULL;
free(temp);
}

void deleteSpecific(int value){
struct Node *temp=head,*prev=NULL;
if(head==NULL){
   printf("List is empty .Nothing to delete\n");
   return;
}
if(head->data==value){
   head=head->next;
   printf("Deleted element:%d\n",temp->data);
   free(temp);
   return;
}
while(temp!=NULL && temp->data!=value){
   prev=temp;
   temp=temp->next;
}
if (temp == NULL) {
   printf("Element %d not found in the list\n", value);
   return;
}

prev->next = temp->next;
printf("Deleted element:%d\n", temp->data);
```

```
free(temp);
}



void displayList(){
struct Node *temp=head;
if(head==NULL){
    printf("List is empty\n");
}
printf("\nLinked List: ");
while (temp!=NULL){
    printf("%d -> ",temp->data);
    temp=temp->next;
}
printf("NULL");
}

int main(){
int choice,n,value;
while(1){
    printf("\n---Singly Linked List Operations---\n");
    printf("1.Create Linked List\n"
            "2.Delete First Element\n"
            "3.Delete Specific Element\n"
             "4.Delete Last Element\n"
               "5.Display\n"
                "6.Exit\n");

    printf("Enter your choice: ");
    scanf("%d",&choice);

    switch(choice){

    case 1:printf("Enter num of nodes: ");

           scanf("%d" ,&n);

           createList(n);

           break;

    case 2:

            deleteFirst();

            break;

    case 3:printf("Enter value to delete: ");

             scanf("%d",&value);

            deleteSpecific(value);

             break;

    case 4:


       deleteLast();break;

    case 5:

        displayList();

        break;

    case 6:printf("Exiting program\n");

            exit(0);


    default:printf("Invalid choice Try again\n");

    }

}

return 0;

}
```

OUTPUT:



```
---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 1
Enter num of nodes: 5
Enter data for node 1: 10
Enter data for node 2: 20
Enter data for node 3: 30
Enter data for node 4: 40
Enter data for node 5: 50

Linked list created successfully

---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 2
Deleted element:10

---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 3
Enter value to delete: 30
Deleted element:30

---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 4
Deleted element:50
```



C:\Users\BMSCE\Desktop\ds\

```
Enter value to delete: 30
Deleted element:30

---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 4
Deleted element:50

---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 5

Linked List:
20 -> 40 -> NULL
---Singly Linked List Operations---
1.Create Linked List
2.Delete First Element
3.Delete Specific Element
4.Delete Last Element
5.Display
6.Exit
Enter your choice: 6
Exiting program

Process returned 0 (0x0)   execution time : 24.599 s
Press any key to continue.
```

Observation:

Date__/__/__
Page____

16/11/25 Lab pg - ~~6~~ 5 ~~b~~ a

WAP to implement Singly Linked List with following operations:
a. Create a linked list
b. Deletion of first element, specified element & last element in the list
c. Display the contents of the linked list.

=> Pseudocode :-

```
create Linked List
 struct node {
     int data;
     node * next;
  };

 void deletefirst () {
        struct node * temp;
         if (head == NULL)
          {
              printf ("list is empty");
              return;
          }
         temp = head;
         head = head -> next;
         free (temp);
      }

void deletelast () {
        struct node *temp, * prev;
        if (head == NULL)
         {
```

Date __/__/__
Page ______

```
{ printf ("empty");
   return;
}

if (head -> next == NULL)
  {
      printf ("Deleted element");
       free (head);
       head = NULL;
        return;
     }
   temp = head;
   while (temp -> next != null) {
          prev = temp;
          temp = temp -> next;
       printf ("Deleted element");
        prev -> next = NULL;
         free (temp);
       }
void deletespecific (int value) {
     if (head == null) {
           printf (empty);
          return }
      if (head -> data = value) {
           head = head -> next
             printf(Deleted element");
             free (temp);
              return;   }
      while (temp != NULL) && (temp->data != value){
           prev = temp;
           temp = temp -> next;
      }
   if (temp == NULL) {
        Printf ("Element not found");
         return;   }
```

Date
Page

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

struct Node {
    int data;
    struct Node *next;
  };

struct Node * head = NULL;

void createList (int n) {
      struct Node *newNode, *temp;
      int data, i;

   if (n<=0) {
        printf ("Numbers of nodes should be
               greater than 0. \n");
        return;
   }

 for (i=1; i<=n; i++) {
     newNode= (struct Node *) malloc (sizeof
                          (struct Node));
   if (newNode ==NULL) {
        printf ("Memory allocation failed \n");
        return;
   }

    printf ("Enter data for node %d: ", i);
    scanf ("%d", &data);
    newNode->data = data;
    newNode -> next = NULL;
if (head == NULL) {
    head = newNode;
  }
  else {
    temp-> next = newNode;
  }
  temp = newNode;
 }
 printf("\n Linked List created succesfully\n");
}


void deleteFirst () {
    struct Node *temp = head
    if (head == NULL) {
        printf(" List is empty. Nothing
            to delete.\n");
        return;
    }

    temp = head;
    head = head -> next;
    printf ("Deleted element : %d\n",
                         temp->data);
    free (temp);
 }

void deleteLast () {
    struct Node *temp = head, *prev;
    if (head == NULL) {
        printf ("List is empty. Nothing to
                delete.\n");
        return;
    }
```

```
only one node
if (head->next == NULL) {
    printf ("Deleted element: %d\n",
                    head->data);

    free(head);
    head = NULL;
    return;
}
temp = head;
while (temp->next != NULL) {
        prev = temp;
        temp = temp->next;
}
printf ("Deleted element: %d\n", temp->
                                data);
prev->next = NULL;
free (temp);
}


void deletespecific (int value) {
    struct Node *temp = head, *prev =
                                NULL;
    if (head == NULL) {
        printf ("List is empty. Nothing
                to delete.\n");
        return;
    }

    if (head->data == value) {
        head = head->next;
deletefront   printf ("Deleted element: %d\n",
                    temp->data);
        free(temp);
        return;
    }
```

```
while (temp != NULL && temp->data != value)
  {
      prev = temp;
      temp = temp->next;
  }
  if (temp == NULL) {
      printf ("Element %d not found in the
                  List\n", value);
      return;
  }
  prev->next = temp->next;
  printf ("Deleted element: %d \n", temp->data);
  free (temp);
}

void displayList () {
      struct Node *temp = head;
      if (head == NULL) {
         printf ("List is empty \n");
         return;
      }
      printf ("\ Linked List: ");
      while (temp != NULL) {
         printf ("%d -> ", temp->data);
         temp = temp->next;
      }
      printf ("NULL\n");
}

int Main () {
      int choice, n, value;

   while (1) {
```

```
printf("\n--- Singly Linked List ---\n");
printf("1. Create Linked List \n");
printf("2. Delete First Element \n");
printf("3. Delete Specific Element \n");
printf("4. Delete Last Element \n");
printf("5. Display List \n");
printf("6. Exit \n");
printf(" Enter your choice: ");
scanf("%d", &choice);

switch (choice) {
    case 1:
        printf("Enter no. of nodes: ");
        scanf("%d", &n);
        createList(n);
        break;

    case 2:
        deleteFirst();
        break;

    case 3:
        printf("Enter value to delete: ");
        scanf("%d", &value);
        deletespecific(value);
        break;

    case 4:
        deleteLast();
        break;

    case 5:
        displayList();
        break;
case 6:
    printf ("Exiting program.\n");
    exit(0);

default:
    printf("Invalid choice! Plz try again\n");
  }
}
return 0;
}
```

Output:-

```
--- Singly linked list Operations ---
1. Create Linked list
2. Delete First Element
3. Delete Specific Element
4. Delete Last Element
5. Display list
6. Exit
Enter your choice : 1
Enter num of nodes : 4
Enter data for node 1 : 10
Enter data for node 2 : 20
Enter data for node 3 : 30
Enter data for node 4 : 40

Linked List created succesfully.


Enter your choice: 2
Deleted element: 10
```

Date ___ Page ___

```
Enter your choice : 4
 Deleted element : 40

Enter your choice : 3
 Enter value to delete : 20

Enter your choice : 5
 Linked list : 30 -> 40 -> NULL

Enter your choice : 6
 Exiting program.
```

[illegible]