```c
/*
Title- 1. Write a C program to create singly linked list and sort it in ascending order.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/
#include<stdio.h>
#include<stdlib.h>

struct node* CreateNode();
void CreateLinkedList(struct node**);
void SortLinkedList(struct node**);
void DisplayLinkedList(struct node*);

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



void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n1) Create Linked List\n2) Sort In Assending Order\n3) Display Linked
List\n4) Exit\nEnter the Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: SortLinkedList(&first);
                    break;
            case 3: DisplayLinkedList(first);
                    break;
        }
    }while(choice!=4);

}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the data for newnode: ");
        scanf("%d",&(newnode -> data));
        newnode -> next = NULL;
    }
    return newnode;
}
```

```c
void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}

void SortLinkedList(struct node** head)
{
    struct node *tempnode1=NULL,*tempnode2=NULL;
    if(*head == NULL)
    {
        printf("Linked List not Created.....\n");
    }
    else
    {
        tempnode1 = *head;
        int temp;
        while(tempnode1 != NULL)
        {
            tempnode2 = tempnode1 -> next;
            while(tempnode2 != NULL)
            {
                if((tempnode1->data) > (tempnode2->data))
                {
                    temp = tempnode1->data;
                    tempnode1->data = tempnode2->data;
                    tempnode2->data = temp;
                }
                tempnode2 = tempnode2->next;
            }
            tempnode1 = tempnode1->next;
        }
    }
}


void DisplayLinkedList(struct node* tempnode)
{
    printf("Your Linked List is: ");
    while(tempnode != NULL)
    {
        printf(" -> %d",tempnode->data);
        tempnode = tempnode->next;
    }
}
```

```c
/*
Title- 2. Write a C program to create singly linked list and accept one number from user
and find it in given linked list.
Author- Bhakare Mahesh Santosh
```

```
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

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

struct node* CreateNode();
void CreateLinkedList(struct node**);
void SearchElement(struct node*);


void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("------------------------
****************************----------------------------");
        printf("\n1) Create Linked List\n2) Search Element\n3) Exit\nEnter your choice:
");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: SearchElement(first);
                    break;
        }
    }while(choice != 3);
}

struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry, Memory not allocated..., Please close some other application or
restart your computer and try again....");
    }
    else
    {
        printf("Enter the value for newnode: ");
        scanf("%d",&(newnode->data));
        newnode->next = NULL;
    }
    return newnode;
}

void CreateLinkedList(struct node** head)
{
    struct node *newnode = NULL,*tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
```

```
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}
void SearchElement(struct node* head)
{
    int x,flag = 0,count = 0;
    printf("Enter the element to search: ");
    scanf("%d",&x);
    while(head != NULL)
    {
        count++;
      if(x == (head->data))
      {
          flag = 1;
          break;
      }
      head = head->next;
    }
    if(flag == 1)
    {
        printf("element found at %d position.\n",count);
    }
    else
    {
        printf("element not found in linked list...\n");
    }
}



/*
Title- 3. Write a C program to create two singly linked list and merge it in one singly
linked list.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

struct node* CreateNode();
void CreateLinkedList(struct node**);
void MergeLinkedList(struct node*, struct node*, struct node**);
void DisplayLinkedList(struct node*);

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

void main()
{
    struct node *first = NULL, *second = NULL, *third = NULL;
    int choice;
    do
    {
        printf("\n----------------------------
**********************----------------------------\n");
        printf("1)Create first Linked List\n2) Create second Linked List\n3) Merge Linked
```

```
List\n4) Display Merged Linked List\n5) Display Original Linked List\n6) exit\nEnter your
Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: CreateLinkedList(&second);
                    break;
            case 3: MergeLinkedList(first, second, &third);
                    break;
            case 4: DisplayLinkedList(third);
                    break;
            case 5: DisplayLinkedList(first);
                    DisplayLinkedList(second);
                    break;
        }
    }while(choice!=6);
}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry, Memory not allocated..., Please close some other application or
restart your computer and try again....");
    }
    else
    {
        newnode->next = NULL;
    }
    return newnode;
}

void CreateLinkedList(struct node** head)
{
    struct node *newnode = NULL,*tempnode = *head;
    newnode = CreateNode();
    printf("Enter the value for newnode: ");
    scanf("%d",&(newnode->data));
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}

void MergeLinkedList(struct node* first, struct node* second, struct node** third)
{
    struct node* tempnode = NULL;
    while(*third != NULL)
    {
        tempnode = (*third) ->next;
        free(*third);
        *third = tempnode;
    }
```

```
    while(first != NULL)
    {
        if(*third == NULL)
        {
            *third = CreateNode();
            (*third)->data = first->data;
        }
        else
        {
            tempnode = *third;
            while(tempnode->next != NULL)
            {
                tempnode = tempnode->next;
            }
            tempnode->next = CreateNode();
            tempnode->next->data = first->data;
        }
        first = first->next;
    }
    while(second != NULL)
    {
        if(*third == NULL)
        {
            *third = CreateNode();
            (*third)->data = second->data;
        }
        else
        {
            tempnode = *third;
            while(tempnode->next != NULL)
            {
                tempnode = tempnode->next;
            }
            tempnode->next = CreateNode();
            tempnode->next->data = second->data;
        }
        second = second->next;
    }
    printf("Linked List Merged Successfully.....\n");
}

void DisplayLinkedList(struct node* head)
{
    printf("Your Linked List is: ");
    while(head!= NULL)
    {
        printf(" -> %d",head->data);
        head = head->next;
    }
    printf("\n");
}


/*
Title-  4. Write a C program to replace given one value with first occurrence of value in
given linked list.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

struct node* CreateNode();
void CreateLinkedList(struct node**);
```

```
void DisplayLinkedList(struct node*);
void ReplaceElement(struct node*);

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


void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n-------------------------
****************************-------------------------\n");
        printf("1) Create Linked List\n2) Display Linked List\n3) Replace Element\n4)
Exit\nEnter your Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: DisplayLinkedList(first);
                    break;
            case 3: ReplaceElement(first);
                    break;
        }
    }while(choice != 4);
}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the data for newnode: ");
        scanf("%d",&(newnode -> data));
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
```

```
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}


void DisplayLinkedList(struct node* tempnode)
{
    printf("Your Linked List is: ");
    while(tempnode != NULL)
    {
        printf(" -> %d",tempnode->data);
        tempnode = tempnode->next;
    }
}


void ReplaceElement(struct node* head)
{
    int x,flag = 0;
    printf("Enter which element you want to replace: ");
    scanf("%d",&x);
    if(head == NULL)
    {
        printf("Linked List not Created... Please create Linked List..\n");
    }
    else
    {
        while( head != NULL )
        {
            if(x == head->data)
            {
                printf("Please enter new element: ");
                scanf("%d",&x);
                head->data = x;
                flag =1;
                break;
            }
            head = head -> next;
        }
        if(flag ==1)
        {
            printf("Element Replaced Successfully.....\n");
        }
        else
        {
            printf("Element Not found to replace...\n");
        }
    }
}


/*
Title- 5. Write a C program to find number of occurrence of given number in given
linked list.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

struct node* CreateNode();
void CreateLinkedList(struct node**);
```

```c
void ElementOccurence(struct node*);

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


void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n-------------------------
****************************-------------------------\n");
        printf("1) Create Linked List\n2) Element Occurance\n3) Exit\nEnter your Choice:
");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: ElementOccurence(first);
                    break;
        }
    }while(choice != 3);
}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the data for newnode: ");
        scanf("%d",&(newnode -> data));
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
```

```
    }
}
void ElementOccurence(struct node* head)
{
    int element,count=0;
    if(head == NULL)
    {
        printf("Linked List not Created ..... Please Create Linked List.....\n");
    }
    else
    {
        printf("Please enter the element whose occurnces you want to count: ");
        scanf("%d",&element);
        while(head != NULL)
        {
            if(element == head->data)
            {
                count++;
            }
            head = head -> next;
        }
        printf("Occurances of Element is : %d\n",count);
    }
}


/*
Title-  6. Write a C program to print singly linked list in reverse order.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

struct node* CreateNode();
void CreateLinkedList(struct node**);
void ReversedLinkedList(struct node*);

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



void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n1) Create Linked List\n2) Reverse Linked List\n3) Exit\nEnter the
Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: ReversedLinkedList(first);
                    break;
        }
    }while(choice!=3);
```

```
}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the data for newnode: ");
        scanf("%d",&(newnode -> data));
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}


void ReversedLinkedList(struct node* tempnode)
{
   if(tempnode != NULL)
    {
        ReversedLinkedList(tempnode->next);
        printf(" -> %d",tempnode->data);
    }
}


/*
Title- 7. Write a C program to create linked list with multiple strings.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

char* AcceptString();
struct node* CreateNode();
void CreateLinkedList(struct node**);
void DisplayLinkedList(struct node*);
```

```c
struct node
{
    char* str;
    struct node* next;
};



void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("--------------------------------
******************************----------------------------------------\n");
        printf("\n1) Create Linked List\n2) Display Linked List\n3) Exit\nEnter the
Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: DisplayLinkedList(first);
                    break;
        }
    }while(choice!=3);

}


char* AcceptString()
{
    int len = 1;
    char ch;
    char* str = NULL;
    str = (char*)malloc(sizeof(char));
    (*str) = '\0';
    scanf(" ");
    do
    {
        scanf("%c",&ch);
        len++;
        str = (char*)realloc(str, sizeof(char)+len);
        *(str+(len-2)) = ch;
        *(str+(len-1)) = '\0';
    }while(ch != '\n');
    return str;
}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the string for newnode: ");
        newnode->str = AcceptString();
        newnode -> next = NULL;
```

```
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}


void DisplayLinkedList(struct node* tempnode)
{
    printf("Your Linked List is: \n");
    while(tempnode != NULL)
    {
        printf("%s",tempnode->str);
        tempnode = tempnode->next;
    }
}


/*
Title- 8. Write a C program to create two linked lists (B and C) from one linked
list(A) one of two(B) contains only even data from (A) and another contains
only odd data from (A).
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/


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

struct node* CreateNode();
void CreateLinkedList(struct node**);
void EvenOddSeperator(struct node*, struct node**, struct node**);
void DisplayLinkedList(struct node*);

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



void main()
{
    struct node *first = NULL, *even = NULL, *odd = NULL;
    int choice;
```

```
    do
    {
        printf("\n1) Create Linked List\n2) Display Original Linked List\n3) Seperate
Even Odd\n4) Display Even Linked List\n5) Display Odd Linked List\n6) Exit\nEnter the
Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: DisplayLinkedList(first);
                    break;
            case 3: EvenOddSeperator(first, &even, &odd);
                    break;
            case 4: DisplayLinkedList(even);
                    break;
            case 5: DisplayLinkedList(odd);
                    break;
        }
    }while(choice!=6);

}


struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    printf("Enter the data for newnode: ");
    scanf("%d",&(newnode -> data));
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}

void EvenOddSeperator(struct node* head, struct node** even, struct node** odd)
{
    struct node* tempnode = NULL;
    while(*even != NULL)
```

```
{
    tempnode = (*even)->next;
    free(*even);
    *even = tempnode;
}
while(*odd != NULL)
{
    tempnode = (*odd)->next;
    free(*odd);
    *odd = tempnode;
}

if(head == NULL)
{
    printf("List List not available....\n");
}
else
{
    while(head != NULL)
    {

        if((head->data)%2 == 0)
        {
            if(*even == NULL)
            {
                *even = CreateNode();
                (*even)->data = head->data;
                (*even)->next = NULL;
            }
            else
            {
                tempnode = *even;
                while(tempnode->next != NULL)
                {
                    tempnode = tempnode->next;
                }
                tempnode->next = CreateNode();
                tempnode->next->data = head->data;
                tempnode->next->next = NULL;
            }
        }
        else
        {
            if(*odd == NULL)
            {
                *odd = CreateNode();
                (*odd)->data = head->data;
                (*odd)->next = NULL;
            }
            else
            {
                tempnode = *odd;
                while(tempnode->next != NULL)
                {
                    tempnode = tempnode->next;
                }
                tempnode->next = CreateNode();
                tempnode->next->data = head->data;
                tempnode->next->next = NULL;
            }
        }
        head = head->next;
    }
}
```

```
}

void DisplayLinkedList(struct node* head)
{
    while(head != NULL)
    {
        printf("%d -> ",head->data);
        head = head->next;
    }
}



/*
Title- 9. Write a C program to create linked list to store student information and print
5 student information using singly linked list.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/

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

char* AcceptString();
struct node* CreateNode();
void CreateLinkedList(struct node**);
void DisplayLinkedList(struct node*);
struct student
{
    int roll_no;
    char* name;
    char* address;
    float marks;
};
struct node
{
    struct student s;
    struct node* next;
};



void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n1) Create Linked List\n2) Display Linked List\n3) Exit\nEnter the
Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: DisplayLinkedList(first);
                    break;
        }
    }while(choice!=3);

}

char* AcceptString()
{
    int len = 1;
```

```
    char ch;
    char* str = NULL;
    str = (char*)malloc(sizeof(char));
    (*str) = '\0';
    scanf(" ");
    scanf("%c",&ch);
    while(ch != '\n')
    {
        len++;
        str = (char*)realloc(str, sizeof(char)+len);
        *(str+(len-2)) = ch;
        *(str+(len-1)) = '\0';
        scanf("%c",&ch);
    }
    return str;
}

struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the Roll No. for student: ");
        scanf("%d",&(newnode -> s.roll_no));
        printf("Enter the Name for student: ");
        newnode->s.name = AcceptString();
        printf("Enter the Address for student: ");
        newnode->s.address = AcceptString();
        printf("Enter the marks for student: ");
        scanf("%f",&(newnode -> s.marks));
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}
void DisplayLinkedList(struct node* head)
{
    while(head != NULL)
    {
        printf("\t|\t%d  \t|\t%s\t|\t%s\t|\t%f\t|\n",head->s.roll_no,head->s.name,head-
>s.address,head->s.marks);
```

```
        head = head -> next;
    }

}



/*
Title- 10. Write a C program to sort student information in ascending order according to
marks of student using singly linked list.
Author- Bhakare Mahesh Santosh
ID- 492
Batch- TechnOrbit(PPA-8)
*/


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

char* AcceptString();
struct node* CreateNode();
void CreateLinkedList(struct node**);
void DisplayLinkedList(struct node*);
void SortLinkedList(struct node*);
struct student
{
    int roll_no;
    char* name;
    char* address;
    float marks;
};
struct node
{
    struct student s;
    struct node* next;
};



void main()
{
    struct node* first = NULL;
    int choice;
    do
    {
        printf("\n1) Create Linked List\n2) Display Linked List\n3) Sort Linked List\n4)
Exit\nEnter the Choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1: CreateLinkedList(&first);
                    break;
            case 2: DisplayLinkedList(first);
                    break;
            case 3: SortLinkedList(first);
                    break;
        }
    }while(choice!=4);

}

char* AcceptString()
{
    int len = 1;
    char ch;
    char* str = NULL;
```

```
    str = (char*)malloc(sizeof(char));
    (*str) = '\0';
    scanf(" ");
    scanf("%c",&ch);
    while(ch != '\n')
    {
        len++;
        str = (char*)realloc(str, sizeof(char)+len);
        *(str+(len-2)) = ch;
        *(str+(len-1)) = '\0';
        scanf("%c",&ch);
    }
    return str;
}

struct node* CreateNode()
{
    struct node* newnode = NULL;
    newnode = (struct node*)malloc(sizeof(struct node));
    if(newnode == NULL)
    {
        printf("Sorry,Memory not allocated... Please close the other applications or
restart your computer and try again...\n");
    }
    else
    {
        printf("Enter the Roll No. for student: ");
        scanf("%d",&(newnode -> s.roll_no));
        printf("Enter the Name for student: ");
        newnode->s.name = AcceptString();
        printf("Enter the Address for student: ");
        newnode->s.address = AcceptString();
        printf("Enter the marks for student: ");
        scanf("%f",&(newnode -> s.marks));
        newnode -> next = NULL;
    }
    return newnode;
}


void CreateLinkedList(struct node** head)
{
    struct node* newnode = NULL;
    struct node* tempnode = *head;
    newnode = CreateNode();
    if(*head == NULL)
    {
        *head = newnode;
    }
    else
    {
        while(tempnode->next != NULL)
        {
            tempnode = tempnode->next;
        }
        tempnode->next = newnode;
    }
}
void DisplayLinkedList(struct node* head)
{
    while(head != NULL)
    {
        printf("\t|\t%d  \t|\t%s\t|\t%s\t|\t%f\t|\n",head->s.roll_no,head->s.name,head-
>s.address,head->s.marks);
        head = head -> next;
    }
```

```
}

void SortLinkedList(struct node* head)
{
    struct node* tempnode = NULL;
    struct student temp;
    while(head != NULL)
    {
        tempnode = head->next;
        while(tempnode != NULL)
        {
            if(head->s.marks > tempnode->s.marks)
            {
                temp = head->s;
                head->s = tempnode->s;
                tempnode->s = temp;
            }
            tempnode = tempnode->next;
        }
        head = head->next;
    }
}
```