LAB- 2 DSA

Name- Shreya Mishra

SRN- PES1UG21CS574

odll_client.c

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include"odll.h"

int main(int argc, char const *argv[])
{
    int choice;

    NODE *head;
    initList(&head);
    float cgpa;
    char name[30];
    char srn[14];
    do{
        printf("1. Insert in DLL\n");
        printf("2. Search in DLL\n");
        printf("3. Exit\n");
        printf("Enter Choice : ");
        scanf("%d",&choice);

        switch(choice){
            case 1:

                printf("Enter Name : ");
                scanf("%s",name);
                printf("Enter SRN : ");
                scanf("%s",srn);
                printf("Enter CGPA : ");
                scanf("%f",&cgpa);
                SortedInsert(&head,srn,name,cgpa);
                break;
            case 2:
                printf("Enter SRN to be searched : ");
                scanf("%s",srn);
                NODE *temp = search(head,srn);
                display(temp);
                break;
            case 3:
                exit(0);

        }

    }while (choice<3 && choice>0);
    return 0;
}
```

odll.c

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include"odll.h"

void initList(NODE **head){
    (*head)=NULL;
}

NODE* newNode(char *srn,char *name,float cgpa){
    NODE *temp= malloc(sizeof(NODE));
    temp->s=malloc(sizeof(STUDENT));
    temp->next=NULL;
    temp->prev=NULL;
    temp->s->cgpa=cgpa;
    strcpy(temp->s->name,name);
    strcpy(temp->s->srn,srn);
    return temp;
}

int compareSRN(char *s1,char*s2){
    int k=strcmp(s1,s2);
    return k;
}

void SortedInsert(NODE **head,char *srn,char *name,float cgpa){
    NODE *node= newNode(srn,name,cgpa);

    if((*head)==NULL){

        (*head)= node;
    }
    else if (compareSRN((*head)->s->srn,node->s->srn)<0){
        // insert at head
        node->next=(*head);
        (*head)->prev= node;
        (*head)=node;
    }
    else{

        NODE *temp= (*head);
        while(temp!=NULL && compareSRN(temp->next->s->srn,node->s->srn)>0){
            temp=temp->next;
        }
        node->next=temp->next;
        node->prev= temp;
        temp->next=node;
        node->next->prev=node;
    }

}
```

```
NODE *search(NODE*head,char *srn){
    NODE *temp=head;
    while(temp !=NULL && compareSRN(srn,temp->s->srn)!=0){
        temp=temp->next;
    }
    return temp;
}

void display(NODE *p){
    printf("Name : %s",p->s->name);
    printf("CGPA : %f",p->s->cgpa);
}
```

odll.h

```
typedef struct student
{
    char srn[14];
    char name[30];
    float cgpa;
} STUDENT;

typedef struct node
{
    STUDENT *s;
    struct node *prev, *next;
} NODE;

void initList(NODE **head);
NODE* newNode(char *srn,char *name,float cgpa);
void SortedInsert(NODE **head,char *srn,char *name,float cgpa);
NODE *search(NODE *head,char *srn);
void display(NODE *p);
```

OUTPUT :

```
D:\PES clg\DSAinC\Lab\Week2\ques1>a.exe
1. Insert in DLL
2. Search in DLL
3. Exit
Enter Choice : 1
Enter Name : Shreya
Enter SRN : 123
Enter CGPA : 9.5
1. Insert in DLL
2. Search in DLL
3. Exit
Enter Choice : 1
Enter Name : Shreeja
Enter SRN : 234
Enter CGPA : 9.3
1. Insert in DLL
2. Search in DLL
3. Exit
Enter Choice : 1
Enter Name : Adrija
Enter SRN : 345
Enter CGPA : 9.3
1. Insert in DLL
2. Search in DLL
3. Exit
Enter Choice : 2
Enter SRN to be searched : 234
Name : ShreejaCGPA : 9.3000001. Insert in DLL
2. Search in DLL
3. Exit
Enter Choice : 3

D:\PES clg\DSAinC\Lab\Week2\ques1>
```

merge.c

```c
#include <stdio.h>
#include <stdlib.h>
struct Node
{
    int data;
    struct Node *next, *prev;
};
void create(struct Node **p, int n);
void merge(struct Node **p, struct Node **q, struct Node **r);
void merge(struct Node **p, struct Node **q, struct Node **r)
{
    struct Node *pres1, *pres2;
    pres1 = *p;
    pres2 = *q;
    while (pres1 != NULL)
    {
        create(r, pres1->data);
        pres1 = pres1->next;
    }
    while (pres2 != NULL)
    {
        create(r, pres2->data);
        pres2 = pres2->next;
    }
}
```

```c
void create(struct Node **p, int n)
{
    struct Node *temp, *pres;
    temp = (struct Node *)malloc(sizeof(struct Node));
    pres = *p;
    temp->data = n;
    temp->prev = NULL;
    temp->next = NULL;
    if (*p == NULL)
    {
        *p = temp;
    }
    else
    {
        while (pres->next != NULL && n > pres->data)
        {
            pres = pres->next;
        }
        if (n <= pres->data)
        {
            if (pres->prev == NULL)
            {
                temp->next = pres;
                pres->prev = temp;
                *p = temp;
            }
            else
            {
                pres->prev->next = temp;
                temp->prev = pres->prev;
                pres->prev = temp;
                temp->next = pres;
            }
        }
        else
        {
            pres->next = temp;
            temp->prev = pres;
        }
    }
}
void display(struct Node *p)
{
    if (p == NULL)
        printf("List is empty\n");
    else
    {
        while (p != NULL)
        {
            printf("%d\t", p->data);
            p = p->next;
        }
    }
}
```

```c
int main()
{
    struct Node *pl1, *pl2, *pl3;
    int x, ch;
    pl1 = NULL;
    pl2 = NULL;
    pl3 = NULL;
    do
    {
        printf("\n0: exit\n1: Enter value to list 1\n2 :Enter value to list 2\n3: merge\n4
: display list1\n5 : display list2\n6 : display list3\n");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1:
            printf("Enter value of x : ");
            scanf("%d", &x);
            create(&pl1, x);
            break;
        case 2:
            printf("Enter value of x : ");
            scanf("%d", &x);
            create(&pl2, x);
            break;
        case 3:
            merge(&pl1, &pl2, &pl3);
            break;
        case 4:
            display(pl1);
            break;
        case 5:
            display(pl2);
            break;
        case 6:
            display(pl3);
            break;
        }
    } while (ch != 0);
    return 0;
}
```

OUTPUT:

```
D:\PES clg\DSAinC\Lab\Week2\ques2>a.exe

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
1
Enter value of x : 6

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
1
Enter value of x : 4

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
1
Enter value of x : 2

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
```

```
5 : display list2
6 : display list3
4
2       4       6
0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
2
Enter value of x : 9

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
2
Enter value of x : 6

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
2
Enter value of x : 10

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
```

```
3: merge
4 : display list1
5 : display list2
6 : display list3
2
Enter value of x : 8

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
5
6       8       9       10
0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
3

0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
4 : display list1
5 : display list2
6 : display list3
6
2       4       6       6       8       9       10
0: exit
1: Enter value to list 1
2 :Enter value to list 2
3: merge
```