Ajith M S
1BM19CS010

LAB-6

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void create();
void display();
void insertpos(int);
void insert_beg();
void delete();
void delpos(int);
void del_beg();
struct node
{
  int rem, wsh;
  char name[20];
struct node *next;
};
struct node *head=NULL;
int count = 0;
int main(int argc, char *argv)
{
  int choice, ele, a;
do
{
printf("1. Insert at end 2. Insert at begin 3. Insert at any
position 4. Delete at the end 5. Delete at
beginning 6. Delete at any position, 7. Display);
```

```
printf ("\n Enter your choice : ");
scanf ("%d", &choice);
switch (choice)
{
case 1 : create (); break;
case 2 : insert_beg ();
    break
case 3 : printf ("Enter the position to be inserted\n");
    scanf ("%d", &ele);
    insert_pos (ele);
    break;
case 4 : delete (); break;
case 5 : del_beg () break;
case 6 : printf ("Enter the position");
scanf ("%d", &a);
    delpos (a);
    break;
case 7 : display ()
    break;
case 8 : exit (0);
}
} while (choice != 8);
}
```

```
void create()
{ struct node *newnode, *temp;
  int rno1, usn1
  char name1[20];
printf ("Enter name, USN, rno of student:");
scanf ("%s %d %d", name1, &usn1, &rno1);
newnode = (struct node*) malloc (size of (struct node

strcpy(newnode->name, name1);
newnode->usn = usn1;
newnode->rno = rno1;
if (head == NULL);
  {
    newnode->next = NULL;
  head = newnode;
   printf ("Node created\n");
  }
else
{
temp = head;
   while (temp->next != NULL)
{
     temp = temp->next;
   }
temp -> next = newnode;
newnode -> next = NULL;
   printf ("Node created\n");
   count++;
}
```

```
void display()
{
  struct node *ptr=NULL;
 ptr=head;
  if (ptr==NULL)
{
  printf ("No this to print");
  }
else
{
  while (ptr!=NULL)
  {
  printf ("%-s", ptr->name);
  printf ("%-d", ptr->usn);
  printf ("%-d", ptr->sem);
   printf ("\n");
    ptr=ptr->next;
     }
  }
}

void insert beg()
{
  struct node *newnode;
 int sem1, usn1;
  char name1[20];
printf ("Enter name usn sem of student: ");
scanf ("%s %d %d", name1, &usn1, &sem1);
newnode=(struct node*) malloc (size of (struct node));
```

```
strcpy (newnode->name, name);
newnode->usn = usn1;
newnode->sem = sem1
newnode->next = head;
head = newnode;
count++;
}

void delete()
{ struct node *temp=NULL;
  int sem1, usn1;
  char name1[20];
  if (head ==NULL)
  printf ("Linked list is empty");
  else
  {
   temp =head;
   while (temp->next->next !=NULL)
   {
    temp = temp->next;
   }
strcpy (name1, temp->next->name);
sem1 = temp->next->sem;
usn1 = temp->next->usn;
printf ("student info deleted = %d %s %d", name1, usn1, sem1
temp->next=NULL;
   count--;
  }
}
```

```
void delpos(int p)
{
  struct node *temp=NULL;
  int rem, num;
    char name[20];
if (head==NULL)
printf("Linked list empty");
else if (count+1<p)
printf("the position exceeds the no of nodes");
else if (p==1)
 {
    strcpy(name, head->name);
    rem=head->rem;
    num=head->num
   printf("the stored info deleted=%s %d %d", name, rem, num);
   temp=head;
   head=temp->next;
   free(temp);
   count--;
 }
 else
{
   int i;
   struct node *temp, *ptr
   temp=head;
    for(i=2; i<p; i++)
    {
    temp=temp->next;
    }
```

```
strcpy(name1, temp->next->name);
sem1 = temp->next->sem;
usn1 = temp->next->usn;
printf("the student info deleted= %d %d %s %s %s", name1, usn1, s
ptr = temp->next;
temp->next = temp->ref->next;
free(ptr);
count--;
}
}
```