6.
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
void Insertend()
void Insertanypos(int p)
void Insertbegining()
void delete();
void delpos(int)
void del_beg();
void display();
struct node
  {
    int sem;
    char usn[20];
    char name[20];
    struct node * next;
  };
struct node *head=NULL;
int count=0;
int main()
 { int choice, ele, a;
   do {
   printf("\n1. Insert at the end \n2. Insert at the begining
        \n3. Insert at a position\n4. Delete at the
        end \n5. Delete at the begining\n6. Delete
        at a position.\n7. Display\n8. Exit);
 scanf("%d", &choice);
 switch(choice)
   { case 1: Insertend(); break;
     case 2: Insertbegining;
     break;
     case 3: printf("Enter the position");
```

```
scanf("%d", &ele)
insertpos(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 delete()
{
struct node *temp=NULL;
int sem1;
char usn1[20], 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);
    strcpy(usn1, temp->next->usn);
    sem1=temp->next->sem;
    printf("Student deleted %s is %s %d",
      name1, usn1, sem1);
    temp->next=NULL;
      count--;
      }}
```

```
void del_beg ()
{ struct node * temp = NULL;
  int sem1;
  char usn1[20], name1[20];
  if (head == NULL)
    printf("List is empty");
  else {
      strcpy (name1, head -> name);
      strcpy (usn1, head -> usn);
      sem1 = head -> sem;
      printf(" Student deleted %s %s %d,
      name1, usn1, sem1);
      temp = head;
      head = temp -> next;
      free (temp)
      count --;
  }}
void delpos (int p)
  {
    struct node *temp=NULL
    int sem1,
    char usn1[20], name1[20];
    if (head == NULL)
      printf("Linked list is empty.");
    else if (count+1<p)
    printf(" position not possible");
    else if (p == 1)
      { strcpy (name1, head -> name);
      sem1 = head -> sem
    strcpy(usn1 -> head -> usn);
        printf("Student deleted %s %s %d", name
                usn, sem);
```

```
temp = head;
head = temp -> next;
free(temp);
count--;
}
else
{ int i
  struct node *temp, *ptr;
  temp = head;
  for(i = 2; i < pi++)
  { temp = temp -> next;
  }
  strcpy(name1, temp -> next -> name);
  strcpy(usn1, temp -> next -> usn);
  sem1 = temp -> next -> sem;
  printf("student deleted %s %s %d",
  name1, usn1, sem1);
  ptr = temp -> next;
  temp -> next = temp -> next -> next;
  free(ptr);
  count--;
}
}
```

```
void Insertbegining ()
{ struct node * newnode;
  int s;
  char a[50], b[50];
  printf("Enter your name:");
  scanf("%s", a);
  print(" Enter your usn:");
  scanf("%s", b);
  print(" Enter your semester");
  scanf("%d", &s);
  newnode = (struct node *) malloc (sizeof(struct
    node));
  newnode -> sem = s;
   strcpy (newnode -> name, a);
   strcpy (newnode -> usn, b);
  newnode -> next = head;
   head = newnode;
    C++;
    printf("Node Created\n");
}
```

```
void Insertanyppos(int p)
{
    struct node *newnode;
    int s;
    char a[30], b[30];
    printf("Enter your name:");
    scanf("%s", a);
    printf("Enter your usn:");
    scanf("%s", b);
    printf("Enter your semester:");
    scanf("%d", &s);
    newnode = (struct node*) malloc (sizeof(struct
                                     node));
    newnode->sem = s;
    strcpy(newnode->name, a);
    strcpy(newnode->usn, b);
    if (p == 1)
    {
        printf("Node of linked list is inserted in
        position one\n");
        newnode->next = head;
        head = newnode;
        c++;
    }
    else if (head == NULL && p>1)
    {
        printf("List is empty");
        return;
    }
    else if (p > (c+1))
    {
        printf("Not possible as number of nodes
        present is insufficient\n");
        return;
    }
```

```
else
  {
    struct node * temp1;
    struct node * temp2;
    int count = 1;
    temp1 = head;
    while (count < (p-1))
      {  temp1 = temp1 -> next;
         count ++;
      }
    temp2 = temp1 -> next;
    temp1 -> next = newnode;
    newnode -> next = temp2;
    c++;
    printf (" Node inserted at %d position in
    linked list\n", p);
  }
}
void Insatend()
  {
    struct node * newnode;
    struct node * temp;
    int us;
    char n[30], u[30]
    printf ("Enter your name:");
    scanf ("%s", n);
    printf ("Enter your usnster:");
    scanf ("%d, &s");
    printf ("Enter your usn:");
    scanf ("%s" u);
    newnode = (struct node*) malloc (sizeof (
      struct node));
    newnode -> usen = us;
```

```
strcpy(newnode->name, u);
strcpy(newnode->usn, v);
if(head==NULL)
  { newnode->next = NULL;
    head = newnode;
    printf("First node created\n");
      c++;
  }
else
  { temp = head;
    while(temp->next!=NULL)
      {
        temp = temp->next;
      }
    temp->next = newnode;
    newnode->next = NULL;
    c++;
    printf("Node created");
  }
}
void Display()
 { struct node *ptr;
   ptr = head;
   int i=1;
   if(ptr==NULL)
     { printf("Linked list is empty");
     }
   else
     { while(ptr!=NULL)
       { printf("Node %d", i);
         printf("Name %s \n usn %s, Sem %d",
         ptr->name, ptr->usn, ptr->sem);
```

```
i++;
ptr = ptr -> next; }}}
```