```c
/*C-Program to implement LINKED LIST with functions insertion & deletion with specified
position
LAB 5 & 6
*/
#include<stdio.h>
#include<stdlib.h>
struct node
{
int info;
struct node *link;
};
typedef struct node *NODE;
NODE getnode(){
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL){
printf("mem full\n");
exit(0);
}
return x;
}
void freenode(NODE x){
free(x);
}
NODE insert_rear(NODE first,int item){
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
cur=first;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
return first;
}
NODE delete_rear(NODE first){
NODE cur,prev;
if(first==NULL){
printf("list is empty cannot delete\n");
return first;
}
```

Scanned with CamScanner

```
38 NODE delete_rear(NODE first){
39  NODE cur,prev;
40 if(first==NULL){
41  printf("list is empty cannot delete\n");
42  return first;
43 }
44 if(first->link==NULL){
45  printf("item deleted is %d\n",first->info);
46  free(first);
47  return NULL;
48 }
49  prev=NULL;
50  cur=first;
51 while(cur->link!=NULL){
52  prev=cur;
53  cur=cur->link;
54 }
55  printf("item deleted at rear-end is %d",cur->info);
56  free(cur);
57  prev->link=NULL;
58  return first;
59 }
60 NODE insert_pos(int item,int pos,NODE first){
61  NODE temp,cur,prev;
62  int count;
63  temp=getnode();
64  temp->info=item;
65  temp->link=NULL;
66 if(first==NULL&&pos==1){
67  return temp;
68 }
69 if(first==NULL){
70  printf("invalid position\n");
71  return first;
72 }
73 if(pos==1){
74  temp->link=first;
75  first=temp;
76  return temp;
77 }
78  count=1;
79  prev=NULL;
80  cur=first;
```

Scanned with CamScanner

```
77  }
78  count=1;
79  prev=NULL;
80  cur=first;
81  while(cur!=NULL&&count!=pos){
82  prev=cur;
83  cur=cur->link;
84  count++;
85  }
86  if(
87  count==pos){
88  prev->link=temp;
89  temp->link=cur;
90  return first;
91  }
92  printf("invalid position\n");
93  return first;
94  }
95  NODE delete_pos(int pos,NODE first){
96  NODE cur;
97  NODE prev;
98  int count,flag=0;
99  if(first==NULL || pos<0){
100 printf("invalid position\n");
101 return NULL;
102 }
103 if(pos==1){
104 cur=first;
105 first=first->link;
106 freenode(cur);
107 return first;
108 }
109 prev=NULL;
110 cur=first;
111 count=1;
112 while(cur!=NULL){
113 if(count==pos){
114 flag=1;
115 break;
116 }
117 count++;
118 prev=cur;
119 cur=cur->link;
```

Scanned with CamScanner

```
110   cur=first;
111   count=1;
112  while(cur!=NULL){
113  if(count==pos){
114   flag=1;
115   break;
116  }
117   count++;
118   prev=cur;
119   cur=cur->link;
120  }
121  if(flag==0){
122   printf("invalid position\n");
123   return first;
124  }
125   printf("item deleted at given position is %d\n",cur->info);
126   prev->link=cur->link;
127   freenode(cur);
128   return first;
129  }
130  void display(NODE first){
131   NODE temp;
132   if(first==NULL)
133   printf("list empty cannot display items\n");
134  for(temp=first;temp!=NULL;temp=temp->link){
135   printf("%d\n",temp->info);
136  }
137  }
138   void main()
139  {
140   int item,choice,key,pos;
141   int count=0;
142   NODE first=NULL;
143  for(;;){
144   printf("\n1:Insert_rear\n2:Delete_rear\n3:insert_info_position\n4:Delete_info_position\n5:Display_list\n6:Exit\n");
145   printf("enter the choice\n");
146   scanf("%d",&choice);
147  switch(choice){
148   case 1:printf("enter the item at rear-end\n");
149   scanf("%d",&item);
150   first=insert_rear(first,item);
151   break;
152   case 2:first=delete_rear(first);
```

Scanned with CamScanner

```
void display(NODE first){
 NODE temp;
 if(first==NULL)
 printf("list empty cannot display items\n");
for(temp=first;temp!=NULL;temp=temp->link){
 printf("%d\n",temp->info);
 }
 }
 void main()
{
 int item,choice,key,pos;
 int count=0;
 NODE first=NULL;
for(;;){
 printf("\n1:Insert_rear\n2:Delete_rear\n3:insert_info_position\n4:Delete_info_position\n5:Display_list\n6:Exit\n");
 printf("enter the choice\n");
 scanf("%d",&choice);
switch(choice){
 case 1:printf("enter the item at rear-end\n");
 scanf("%d",&item);
 first=insert_rear(first,item);
 break;
 case 2:first=delete_rear(first);
 break;
 case 3:printf("enter the item to be inserted at given position\n");
 scanf("%d",&item);
 printf("enter the position\n");
 scanf("%d",&pos);
 first=insert_pos(item,pos,first);
 break;
 case 4:printf("enter the position\n");
 scanf("%d",&pos);
 first=delete_pos(pos,first);
 break;
 case 5:display(first);
 break;
 default:exit(0);
 break;
 }
 }
 }
```

Scanned with CamScanner

```
"C:\web developement(html.css.js)\lab_5_6_ds.exe"

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
1
enter the item at rear-end
33

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
1
enter the item at rear-end
45

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
1
enter the item at rear-end
23

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
1
enter the item at rear-end
34

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
```

Scanned with CamScanner

```
"C:\web developement(html.css.js)\lab_5_6_ds.exe"
enter the item at rear-end
34

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
1
enter the item at rear-end
67

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
5
33
45
23
34
67

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
2
item deleted at rear-end is 67
1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
5
33
45
23
34
```

Scanned with CamScanner

```
"C:\web developement(html.css.js)\lab_5_6_ds.exe"

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
3
enter the item to be inserted at given position
66
enter the position
4

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
5
33
45
23
66
34

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
4
enter the position
2
item deleted at given position is 45

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
```

Scanned with CamScanner

```
"C:\web developement(html.css.js)\lab_5_6_ds.exe"
3
enter the item to be inserted at given position
66
enter the position
4

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
5
33
45
23
66
34

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
4
enter the position
2
item deleted at given position is 45

1:Insert_rear
2:Delete_rear
3:insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
enter the choice
6

Process returned 0 (0x0)   execution time : 206.475 s
Press any key to continue.
```

Scanned with CamScanner