

```

1  #include<stdio.h>
2  #include<malloc.h>
3  #include<stdlib.h>
4  struct node
5  {
6      int info;
7      struct node *link;
8  };
9  typedef struct node *NODE;
10 NODE getnode()
11 {
12     NODE x;
13     x=(NODE)malloc(sizeof(struct node));
14     if(x==NULL)
15     {
16         printf("Memory full\n");
17         exit(0);
18     }
19     return x;
20 }
21 void freenode(NODE x)
22 {
23     free(x);
24 }
25
26
27 NODE insert_rear(NODE first,int item)
28 {
29     NODE temp,cur;
30     temp=getnode();
31     temp->info=item;
32     temp->link=NULL;
33     if(first==NULL)
34         return temp;
35     cur=first;
36     while(cur->link!=NULL)
37         cur=cur->link;
38     cur->link=temp;

```

```

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

```

```

74 int count;
75 temp=getnode();
76 temp->info=item;
77 temp->link=NULL;
78 if(first==NULL&&pos==1)
79 {
80     return temp;
81 }
82 if(first==NULL)
83 {
84     printf("Invalid position\n");
85     return first;
86 }
87 if(pos==1)
88 {
89     temp->link=first;
90     first=temp;
91     return temp;
92 }
93 count=1;
94 prev=NULL;
95 cur=first;
96 while(cur!=NULL&&count!=pos)
97 {
98     prev=cur;
99     cur=cur->link;
100     count++;
101 }
102 if(count==pos)
103 {
104     prev->link=temp;
105     temp->link=cur;
106     return first;
107 }
108 printf("Invalid position\n");
109 return first;
110 }
111

```

```

111
112
113     NODE delete_pos(int pos, NODE first)
114     {
115         NODE cur;
116         NODE prev;
117         int count, flag=0;
118         if(first==NULL || pos<0)
119         {
120             printf("Invalid position\n");
121             return NULL;
122         }
123         if(pos==1)
124         {
125             cur=first;
126             first=first->link;
127             freenode(cur);
128             return first;
129         }
130         prev=NULL;
131         cur=first;
132         count=1;
133         while(cur!=NULL)
134         {
135             if(count==pos) {flag=1; break;}
136             count++;
137             prev=cur;
138             cur=cur->link;
139         }
140         if(flag==0)
141         {
142             printf("Invalid position\n");
143             return first;
144         }
145         printf("Item deleted at given position is %d\n", cur->info);
146         prev->link=cur->link;
147         freenode(cur);
148         return first;

```

```

152 void display(NODE first)
153 {
154     NODE temp;
155     if(first==NULL)
156         printf("List is empty cannot display items\n");
157     for(temp=first;temp!=NULL;temp=temp->link)
158     {
159         printf("%d\n",temp->info);
160     }
161 }
162
163
164 int main()
165 {
166     int item,choice,pos;
167     NODE first=NULL;
168     for(;;)
169     {
170         printf("\n 1:Insert_rear\n 2>Delete_rear\n");
171         printf(" 3:Insert_info_position\n 4>Delete_info_position\n 5:Display_list\n 6:Exit\n");
172         printf("Enter the choice : ");
173         scanf("%d",&choice);
174         switch(choice)
175         {
176             case 1:printf("Enter the item at rear-end:\n");
177                     scanf("%d",&item);
178                     first=insert_rear(first,item);
179                     break;
180             case 2:first=delete_rear(first);
181                     break;
182             case 3:printf("Enter the item to be inserted at given position:\n");
183                     scanf("%d",&item);
184                     printf("Enter the position:\n");
185                     scanf("%d",&pos);
186                     first=insert_pos(item,pos,first);
187                     break;
188             case 4:printf("Enter the position:\n");
189                     scanf("%d",&pos);

```

```

163 int main()
164 {
165     int item, choice, pos;
166     NODE first=NULL;
167     for(;;)
168     {
169         printf("\n 1:Insert_rear\n 2:Delete_rear\n");
170         printf(" 3:Insert_info_position\n 4:Delete_info_position\n 5:Display_list\n 6:Exit\n");
171         printf("Enter the choice : ");
172         scanf("%d",&choice);
173         switch(choice)
174         {
175             case 1:printf("Enter the item at rear-end:\n");
176                 scanf("%d",&item);
177                 first=insert_rear(first,item);
178                 break;
179             case 2:first=delete_rear(first);
180                 break;
181             case 3:printf("Enter the item to be inserted at given position:\n");
182                 scanf("%d",&item);
183                 printf("Enter the position:\n");
184                 scanf("%d",&pos);
185                 first=insert_pos(item,pos,first);
186                 break;
187             case 4:printf("Enter the position:\n");
188                 scanf("%d",&pos);
189                 first=delete_pos(pos,first);
190                 break;
191             case 5:display(first);
192                 break;
193             default:exit(0);
194                 break;
195         }
196     }
197     return 0;
198 }
199
200

```



```
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:
1
```

```
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:
2
```

```
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:
3
```

```
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:
12
Enter the position:
2
```

12

Enter the position:

2

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:

10

Enter the position:

3

1:Insert_rear

2>Delete_rear

3:Insert_info_position

4>Delete_info_position

5:Display_list

6:Exit

Enter the choice : 5

1

12

10

2

3

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:

3

Item deleted at given position is 10

1:Insert_rear

2>Delete_rear


```
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:

1

```
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:

5

Invalid position

```
1:Insert_rear
2:Delete_rear
3:Insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
```

Enter the choice : 5

12

2

3

```
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:

1

```
3:Insert_info_position
4:Delete_info_position
5:Display_list
6:Exit
```

Enter the choice : 5

12

2

3

```
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:

1

```
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 3

```
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:

3

Invalid position

```
1:Insert_rear
```

4:Delete_info_position
5:Display_list
6:Exit

Enter the choice : 4

Enter the position:

1

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 3

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:

3

Invalid position

1:Insert_rear
2:Delete_rear
3:Insert_info_position
4:Delete_info_position
5:Display_list
6:Exit

Enter the choice : 45

(program exited with code: 0)

Press any key to continue . . .