```
struct node
  日 {
  ø fre
                  int info;
                  struct node *link;

    get

         10
  ø ins
              typedef struct node *NODE;
  NODE getnode()
Struc
        13
         14
                  NODE X;
B & not
         15
                  x=(NODE)malloc(sizeof(struct node));
   ii 🏺
         16
                  if (x==NULL)
        17
                      printf("Memory full\n");
Type
         19
                      exit(0);
  P NC
         20
         21
                  return x;
         22
             void freenode(NODE x)
         24
            ₽{
         25
                  free(x);
         26
         27
             NODE insert rear(int item, NODE first)
        28
         29
                  NODE temp, cur;
         30
                  temp=getnode();
        31
                  temp->info=item;
         32
                  temp->link=NULL;
         33
                  if (first==NULL)
         34
        35
                      return temp;
line: 68 / 167 col: 16
                 sel: 0
                              TAB mode: CRLF
                         INS
                                               encoding: UTF-8
                                                              filetype: C scope: delete rear
```

p/*WAP to Implement Singly Linked List with following operations

the list. c) Display the contents of the linked list.*/

a) a) Create a linked list. b) Deletion of first element, specified element and last element in

🗏 🤣 Func

ø del

ø del

#include <stdio.h>
#include <stdlib.h>

```
26
🗏 🔗 Func
             NODE insert rear(int item, NODE first)

ø del

        28
  29
                  NODE temp, cur;
        30
                  temp=getnode();

ø del

                  temp->info=item;
        31
  32
                  temp->link=NULL;
  free
        33
                  if (first==NULL)
        34

    get

        35
                      return temp;
  🤣 ins
        36
  37
                  cur=first;
Struc
                  while (cur->link!=NULL)
        39
B & not
                      cur=cur->link;
        40
   ii 🔊
        41
        42
                  cur->link=temp;
                  return first;
        43
Type
        44
  ₽ NC
        45
             NODE delete rear (NODE first)
        46 申{
        47
             NODE cur, prev;
             if (first==NULL)
        49
             printf("list is empty cannot delete\n");
        51
             return first;
        52
        53
             if (first->link==NULL)
        54
        55
             printf("item deleted is %d\n", first->info);
        56
             free (first);
        57
             return NULL;
        58
        59
             prev=NULL;
            |cur=first;
< >
line: 68 / 167 col: 16 sel: 0
                              TAB
                                   mode: CRLF
                                              encoding: UTF-8
                                                             filetype: C scope: delete_rear
                        INS
```

```
cur=cur->link;
  ø ins
        65
  printf("iten deleted at rear-end is %d", cur->info);
Struc
             free (cur);
             prev->link=NULL;
B 🦻 noc
             return first;
   🦆 jı
        70
   o li
        71
             NODE delete front (NODE first)
        72 早{
🗏 🦻 Type
        73
             NODE temp;
  P NC
        74
             if (first==NULL)
        75
             printf("list is empty cannot delete\n");
        77
             return first;
        78
        79
             temp=first;
             temp=temp->link;
             printf("item deleted at front-end is=%d\n", first->info);
        81
        82
             free (first);
        83
             return temp;
        84
        85
             NODE delete pos(int pos, NODE first)
        86
        87
                  NODE prev, cur;
        88
                  int count;
        89
                  if (first==NULL || nos<=0)</pre>
< >
line: 68 / 167 col: 16 sel: 0
                                              encoding: UTF-8
                                                            filetype: C scope: delete_rear
                        INS
                             TAB mode: CRLF
```

printf("item deleted is %d\n", first->info);

🗏 🔗 Func

ø del

ø del

get

57

58

62

free (first);

return NULL;

prev=NULL;

cur=first;

prev=cur;

while(cur->link!=NULL)

```
84
🗏 🔗 Func
             NODE delete pos(int pos, NODE first)

ø del

        86

ø del

        87
                  NODE prev, cur;
        88
                  int count;
  if (first==NULL || pos<=0)</pre>
  free
        91
                      printf("Invalid position\n");
                      return NULL;

    get

  ø ins
                  if (pos==1)
  95
Struc
                      cur=first;
                      first=first->link;
🗏 🦻 noc
                      freenode (cur);
                      return first;
    o li
       100
                  prev=NULL;
       101
Type
                  cur=first;
       102
  ₽ NC
       103
                  count=1;
       104
                  while (cur!=NULL)
       105
       106
                      if (count==pos)
       107 自
       108
                           break;
       109
       110
                      prev=cur;
       111
                      cur=cur->link;count++;
       112
       113
                  if (count!=pos)
       114
       115
                      printf("Invalid position\n");
       116
                      return first;
       117
       118
                  prev->link=cur->link;
line: 68 / 167 col: 16
                 sel: 0
                         INS
                              TAB
                                   mode: CRLF
                                               encoding: UTF-8
                                                              filetype: C scope: delete_rear
```

```
■ Punc 113
                      printf("Invalid position\n");
       115

  del

       116
                      return first;
       117

ø del

                 prev->link=cur->link;
       119
                 freenode (cur);
  free
       120
                 return first;
       121

    get

             void display(NODE first)
       123 早{

    ma

       124
                 NODE temp;
Struc 125
                 if (first==NULL)
       126
B P noc
       127
                      printf("Linked is empty cannot display items\n");
       128
    oli
       129
                 printf("The contents of the linked list are:\n");
       130
                 for (temp=first;temp!=NULL;temp=temp->link)
3 Type
       131
  ₽ NC
       132
                      printf("%d\n", temp->info);
       133
       134
       135
       136
             int main()
       137
       138
             int item, choice, pos;
       139
             NODE first=NULL;
       140
             for(;;)
       141
       142
             printf("\n 1:Insert rear\n 2:Delete front\n 3:Delete rear\n4:Delete at specified position 5:Display list\n6:Exit\n");
             printf("enter the choice\n");
       143
       144
             scanf ("%d", &choice);
       145
             switch (choice)
       146
       147
            case 1:printf("enter the item at rear-end\n");
< >
                                              encoding: UTF-8
                                                            filetype: C
line: 68 / 167 col: 16 sel: 0
                        INS
                             TAB mode: CRLF
                                                                      scope: delete rear
```

if (count!=pos)

```
134
🛮 🔗 Func
       135

ø del

       136
             int main()

    del

       137
       138
             int item, choice, pos;

ø del

             NODE first=NULL;
             for(;;)
       140
  free
      141
            申{
             printf("\n 1:Insert rear\n 2:Delete front\n 3:Delete rear\n4:Delete at specified position 5:Display list\n6:Exit\n");

    get

             printf("enter the choice\n");
  ø ins
             scanf ("%d", &choice);
       144

    ma

       145
             switch (choice)
case 1:printf("enter the item at rear-end\n");
       147
B P no
             scanf("%d", &item);
       149
             first=insert rear(item, first);
   ₀li 150
             break;
3 № Type 151
             case 2:first=delete front(first);
             break;
       152
  ₱ NC
       153
             case 3:first=delete rear(first);
       154
             break;
             case 4:printf("Enter the position:\n");
       155
       156
                                 scanf ("%d", &pos);
       157
                                 first=delete pos(pos, first);
       158
                                 break;
       159
             break;
       160
             case 5:display(first);
       161
             break;
       162
             default:exit(0);
       163
             break;
       164
       165
       166
       167
< >
                                  mode: CRLF
line: 68 / 167 col: 16
                 sel: 0
                        INS
                             TAB
                                              encoding: UTF-8
                                                            filetype: C
                                                                     scope: delete rear
```

```
1:Insert_rear
2:Delete front
3:Delete rear
4:Delete at specified position 5:Display list
6:Exit
enter the choice
enter the item at rear-end
1:Insert_rear
2:Delete front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
enter the item at rear-end
1:Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
enter the item at rear-end
1: Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
enter the item at rear-end
```

```
enter the choice
enter the item at rear-end
1: Insert_rear
2:Delete front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
enter the item at rear-end
1:Insert rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
The contents of the linked list are:
1: Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
item deleted at front-end is=10
1:Insert_rear
2:Delete_front
3:Delete rear
```

```
4:Delete at specified position 5:Display list
6:Exit
enter the choice
item deleted at front-end is=10
1:Insert rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display list
6:Exit
enter the choice
The contents of the linked list are:
1: Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
iten deleted at rear-end is 50
1: Insert rear
2:Delete_front
3:Delete rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
The contents of the linked list are:
1: Insert_rear
2:Delete front
```

```
1: Insert rear
2:Delete_front
3:Delete rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
Enter the position:
1:Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
The contents of the linked list are:
1: Insert_rear
2:Delete_front
3:Delete rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
item deleted at front-end is=20
1:Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
```

```
1: Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
item deleted at front-end is=20
1:Insert_rear
2:Delete_front
3:Delete rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
item deleted is 40
1:Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
list is empty cannot delete
1:Insert_rear
2:Delete_front
3:Delete_rear
4:Delete at specified position 5:Display_list
6:Exit
enter the choice
(program exited with code: 0)
```

Press any key to continue . . . _