*Untitled - Notepad

File Edit Format View Help

```
#include<stdio.h>
#include<malloc.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;
}
int freenode(NODE x)
{
free(x);
return 0;
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
```

*Untitled - Notepad

File Edit Format View Help

```
}
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;
}
if(first->link==NULL)
{
printf("item deleted is %d\n",first->info);
free(first);
return NULL;
}
prev=NULL;
cur=first;
while(cur->link!=NULL)
{
prev=cur;
cur=cur->link;
}
printf("iten deleted at rear-end is %d",cur->info);
free(cur);
prev->link=NULL;

return first;
}
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);
```

*Untitled - Notepad

File Edit Format View Help

```
printf("%d\n",temp->info);
}
}
NODE delete_pos(int pos, NODE first)
{
NODE cur;
NODE prev;
int count;
if(first==NULL || pos<=0)
{
printf("invalid position \n");
return NULL;
}
if (pos==1)
{
cur=first;
first=first->link;
freenode(cur);
return first;
}
prev=NULL;
cur=first;
count=1;
while(cur!=NULL)
{
if(count==pos) break;
prev=cur;
cur=cur->link;
count++;
}
if(count!=pos)
{
printf("invalid position \n");
return first;
}
if(count!=pos)
{
printf("invalid position specified \n");
return first;
}
prev->link=cur->link;
freenode(cur);
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
```

*Untitled - Notepad

File Edit Format View Help

```
}
if(count!=pos)
{
printf("invalid position specified \n");
return first;
}
prev->link=cur->link;
freenode(cur);
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert_front\n 2:Delete_front\n 3:Insert_rear\n 4:Delete_rear\n 5.Delete at specified position \n 6:Display_list\n 7:Exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter the item at front-end\n");
scanf("%d",&item);
first=insert_front(first,item);
break;
case 2:first=delete_front(first);
break;
case 3:printf("enter the item at rear-end\n");
scanf("%d",&item);
first=insert_rear(first,item);
break;
case 4:first=delete_rear(first);
break;
case 5:printf("enter the position of the item to be deleted: \n");
       scanf("%d",&pos);
       first=delete_pos(pos,first);
    break;

case 6:display(first);
break;
default:exit(0);
break;
}
}
getch();
return 0;
}
```

```
 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
1
enter the item at front-end
9

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
1
enter the item at front-end
6

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
```

```
6

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
2
item deleted at front-end is=6

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
4
item deleted is 9

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 7:Exit
enter the choice
```

Date / / Page SPLASH

LAB PROGRAM-6

```
#include <stdio.h>
#include <malloc.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;
}
int freenode(NODE x)
{
free(x);
return 0;
}
NODE insert_front(NODE first, int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
```

```
return temp;
    temp -> link = first;
    first = temp;
    return first;
}
NODE delete_first(NODE first)
{
NODE temp;
if (first == NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp = first;
temp = temp -> link;
printf("item deleted at front end is = %d\n",
                first->info);
free(first);
    return temp;
}
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;
```

Date __/__/__
Page
SPLASH

```
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;
}
if(first -> link == NULL)
{
printf("The item deleted is %d\n", first->info);
free(first);
return NULL;
}
prev = NULL;
cur = first;
while (cur -> link != NULL)
{
prev = cur;
cur = cur -> link;
}
printf("The deleted at rear = node is %d", cur->info);
free(cur);
prev -> link = NULL;
return first;
}
void display(NODE first)
```

Date / /
Page
SPLASH

```
{
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);
  }
}
NODE cur;
NODE prev;
int count;
if (first == NULL || pos <= 0)
{
printf("invalid position \n");
return NULL;
}
if (pos == 1)
{
cur = first;
first = first->link;
freenode(cur);
return first;
}
prev = NULL;
cur = first;
count = 1;
while (cur != NULL)
{
if (count == pos) break;
prev = cur;
cur = cur->link;
```

Date __/__/__
Page
SPLASH

```
count++;
}
prev->link = cur->link;
freenode(cur);
return first;
}
int main()
{
int item, choice, pos;
NODE first = NULL;
Sqslow("CLS");
for(;;)
{
printf("\n1: Insert front\n2: Delete front\n
3. Insert rear\n4. Delete rear\n5. Delete at Specified
position\n6: Display list\n7: Exit\n");
printf("enter the choice\n");
scanf("%d", &choice);
switch(choice)
{
case 1: printf("enter the item at front-end\n");
scanf("%d", &item);
first = insert_front(first, item);
break;
case 2: first = delete_front(first);
break;
case 3: printf("enter the item at rear-end\n");
scanf("%d", &item);
first = insert_rear(first, item);
break;
case 5: printf("enter the position of the item to be deleted:\n");
```

```
        scanf("%d", &pos);
        first = delete_pos(pos, first);
        break;
case 6: display(first);
    break;
    default: exit(0);
    break;
    }
}
    getch();
    return 0;
}
```