*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 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;
```

*Untitled - Notepad

File Edit Format View Help

```
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);
}}

NODE insert_pos( int item, int pos, NODE first)
{
NODE temp;
NODE prev,cur;
int count;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL && pos==1)
{
return temp;
}
if(first==NULL)
{
printf("invalid position \n");
return first;
}
if(pos==1)
{
temp->link=first;
return temp;
}
count=1;
prev=NULL;
cur=first;
while(cur!=NULL && count!=pos)
{
prev=cur;
cur=cur->link;
count++;
}
if(count==pos)
{
prev->link=temp;
temp->link=cur;
return first;
}
```

*Untitled - Notepad

File Edit Format View Help

```
{
prev=cur;
cur=cur->link;
count++;
}
if(count==pos)
{
prev->link=temp;
temp->link=cur;
return first;
}
printf("invalid position \n");
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert_front\n 2:Insert at specified position \n 3:Insert_rear\n 4:Display_list\n 6: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:printf("enter the item to be inserted:\n");
       scanf("%d",&item);
       printf("enter the position at which item to be inserted:\n");
       scanf("%d",&pos);
       first=insert_pos(item,pos,first);
break;
case 3:printf("enter the item at rear-end\n");
scanf("%d",&item);
first=insert_rear(first,item);
break;
case 4:display(first);
break;
default:exit(0);
break;
}
}
getch();
return 0;
}
```

```
 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
2

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
5

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
7
enter the position at which item to be inserted:
2
```

```
enter the item at front-end
5

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
7
enter the position at which item to be inserted:
2

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
4
5
7
2

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
```

Date __/__/__
Page _____
SPLASH

LAB PROGRAM-5

```
#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 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;
}
void display(NODE first)
{
NODE temp;
if(first == NULL)
printf("list empty cannot display\n");
for(temp = first; temp != NULL; temp = temp->link)
{
printf("%d\n", temp->info);
}}
NODE insert_pos(int item, int pos, NODE first)
{
NODE temp;
```

Date __/__/__
Page
SPLASH

```
NODE *prev, *cur;
int count;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
if (first == NULL && pos == 1)
{
return temp;
}
printf("invalid position\n");
return first;
}
if (pos == 1)
{
temp -> link = first;
return temp;
}
count = 1;
prev = NULL;
cur = first;
while (cur != NULL && count != pos)
{
prev = cur;
cur = cur -> link;
count++;
}
if (count == pos)
{
prev -> link = temp;
temp -> link = cur;
return first;
}
```

Page

SPLASH

```
{
prev = cur;
    cur = cur -> link;
    count++;
}
 if (count == pos)
 {
 prev -> link = temp;
 temp -> link = cur;
 return first;
 }
 int main()
 {  int item, choice, pos;
    NODE first = NULL;
    system ("cls");
    for(;;)
{
printf("\n1.Insert front\n 2:Insert at specific position\n3:
 Insert rear\n 4:Display list\n6.Exit\n");
    printf(" enter the choice\n");
     scanf ("%d" choice);
    switch(choice)
   {
 Case1 : printf ("enter the item at front end \n");
   scanf ("%d", &item);
    first = insert_front (first, item);
 break;
 Case 2;  printf ("enter the item to be inserted: \n");
          scanf ("%d", &item);
          printf ("enter the position at which the item
           to be inserted \n");
         scanf ("%d", &pos);
```

```
break;
case3: printf("enter the item at rear-end\n");
 scanf("%d", &item);
 first = insert_rear(first, item);
break;
case 4: display(first);
break;
default: exit(0);
break;
}
}
getch();
return 0;
}.
```