File   Edit   View   Search   Project   Build   Debug   Fortran   wxSmith   Tools   Tools+   Plugins   DoxyBlocks   Settings   Help

`<global>`   `main() : int`

main.c

```c
#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;
```

Logs & others
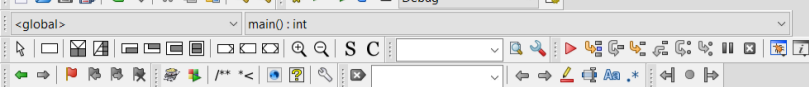
C:\Users\91957\Desktop\cc\WAP to implement stack and queues\main.c

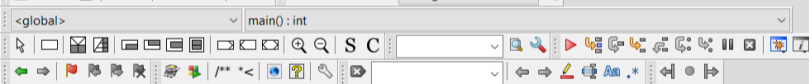C/C++   Windows (CR+LF)   WINDOWS-1252   Line 123, Col 2, Pos 2004   Insert   Read/Write   default

```c
        cur->link=temp;
        return first;
    }

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

    NODE delete_front(NODE first)
    {
    NODE temp;
    if(first==NULL)
    {
    printf("list is empty cannot delete\n");
```

```c
73      printf("list is empty cannot delete\n");
74      return first;
75
76      }
77      temp=first;
78      temp=temp->link;
79      printf("item deleted at front-end is=%d\n",first->info);
80      free(first);
81      return temp;
82      }
83
84      void display(NODE first)
85      {
86      NODE temp;
87      if(first==NULL)
88      printf("list empty cannot display items\n");
89      for(temp=first;temp!=NULL;temp=temp->link)
90      {
91      printf("%d\n",temp->info);
92      }
93      }
94
95      int main()
96      {
97      int item,choice;
98      NODE first=NULL;
99          printf("STACK-insert rear and delete rear\nQUEUE-insert rear and delete front\n");
100     printf("\n 1:Insert_rear\n 2:Delete_rear\n 3:Delete_front\n 4:Display_list\n 5:Exit\n");
101     do
102     {
103     printf("enter the choice\n");
104     scanf("%d",&choice);
105     switch(choice)
106     {
107     case 1:printf("enter the item at rear-end\n");
108     scanf("%d",&item);
```

```c
        for(temp=first;temp!=NULL;temp=temp->link)
        {
        printf("%d\n",temp->info);
        }
        }

        int main()
        {
        int item,choice;
        NODE first=NULL;
            printf("STACK-insert rear and delete rear\nQUEUE-insert rear and delete front\n");
        printf("\n 1:Insert_rear\n 2:Delete_rear\n 3:Delete_front\n 4:Display_list\n 5:Exit\n");
        do
        {
        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:first=delete_front(first);
        break;
        case 4:display(first);
        break;
        case 5:break;
        default:printf("invalid choice");
        break;
        }
        }while(choice!=5);
        return 0;
        }
```

```
STACK-insert rear and delete rear
QUEUE-insert rear and delete front

 1:Insert_rear
 2:Delete_rear
 3:Delete_front
 4:Display_list
 5:Exit
enter the choice
1
enter the item at rear-end
10
enter the choice
1
enter the item at rear-end
20
enter the choice
1
enter the item at rear-end
30
enter the choice
4
10
20
30
enter the choice
3
item deleted at front-end is=10
enter the choice
2
item deleted at rear-end is 30
enter the choice
```

```
item deleted is 20
enter the choice
4
list empty cannot display items
enter the choice
```