

LAB-8:--

PROGRAM

```
#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;
```

```
cur->link=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;
```

```
}
```

```
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);
```

```
    }
```

```
}
```

```
int main()
```

```
{
```

```
    int item,choice,pos;
```

```
    NODE first=NULL;
```

```
for(;;)

{

printf("\n 1:Insert_rear\n 2:Delete_front\n 3:Display_list\n 4:Exit\n");

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_front(first);

        break;

case 3:display(first);

        break;

default:exit(0);
```

break;

}

}

}

SCREENSHOT:--

The screenshot displays a Replit IDE interface. The browser address bar shows the URL `repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c`. The file explorer on the left lists `main.c`. The main editor window shows the following C code:

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 struct node
4 {
5     int info;
6     struct node *link;
7 };
8 typedef struct node *NODE;
9 NODE getnode()
10 {
11     NODE x;
12     x=(NODE)malloc(sizeof(struct node));
13     if(x==NULL)
14     {
15         printf("mem full\n");
16         exit(0);
17     }
18     return x;
19 }
20 void freenode(NODE x)
21 {
22     free(x);
23 }
```

The terminal window on the right shows the program's execution. It prompts the user to "enter the choice" and displays the menu options: 1:Insert\_rear, 2:Delete\_front, 3:Display\_list, 4:Exit. The user has entered '1' twice, and the program has responded with "enter the item at rear-end" each time. The terminal output is as follows:

```
1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
enter the item at rear-end
12

1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
enter the item at rear-end
23

1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
```

The Windows taskbar at the bottom indicates the system time as 20:55 on 06-12-2020.

Replit - DeepskyblueRedundantTec...  
repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c

Files  
main.c

```
main.c
23 }
24 NODE insert_rear(NODE first,int item)
25 {
26     NODE temp,cur;
27     temp=getnode();
28     temp->info=item;
29     temp->link=NULL;
30     if(first==NULL)
31         return temp;
32     cur=first;
33     while(cur->link!=NULL)
34         cur=cur->link;
35     cur->link=temp;
36     return first;
37 }
38
39 NODE delete_front(NODE first)
40 {
41     NODE temp;
42     if(first==NULL)
43     {
44         printf("list is empty cannot delete\n");
```

enter the item at rear-end  
24  
1:Insert\_rear  
2:Delete\_front  
3:Display\_list  
4:Exit  
enter the choice  
1  
enter the item at rear-end  
25  
1:Insert\_rear  
2:Delete\_front  
3:Display\_list  
4:Exit  
enter the choice  
3  
12  
23  
24  
25  
1:Insert\_rear  
2:Delete\_front

Type here to search

Replit - DeepskyblueRedundantTec...  
repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c

Files  
main.c

```
main.c
45 return first;
46 }
47 temp=first;
48 temp=temp->link;
49 printf("item deleted at front-end is=%d\n",
first->info);
50 free(first);
51 return temp;
52 }
53 void display(NODE first)
54 {
55     NODE temp;
56     if(first==NULL)
57         printf("list empty cannot display items\n");
58     for(temp=first;temp!=NULL;temp=temp->link)
59     {
60         printf("%d\n",temp->info);
61     }
62 }
63 int main()
64 {
65     int item,choice,pos;
66     NODE first=NULL;
```

3:Display\_list  
4:Exit  
enter the choice  
2  
item deleted at front-end is=12  
1:Insert\_rear  
2:Delete\_front  
3:Display\_list  
4:Exit  
enter the choice  
2  
item deleted at front-end is=23  
1:Insert\_rear  
2:Delete\_front  
3:Display\_list  
4:Exit  
enter the choice  
2  
item deleted at front-end is=24  
1:Insert\_rear  
2:Delete\_front  
3:Display\_list

Type here to search

```
main.c
66 NUDE t1st=NULL;
67
68 for(;;)
69 {
70 printf("\n 1:Insert_rear\n 2:Delete_front\n
71 3:Display_list\n 4:Exit\n");
72 printf("enter the choice\n");
73 scanf("%d",&choice);
74 switch(choice)
75 {
76 case 1:printf("enter the item at rear-end\n");
77 scanf("%d",&item);
78 first=insert_rear(first,item);
79 break;
80 case 2:first=delete_front(first);
81 break;
82 case 3:display(first);
83 break;
84 default:exit(0);
85 break;
86 }
87 }
```

```
2
item deleted at front-end is=24

1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
2
item deleted at front-end is=25

1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
3
list empty cannot display items

1:Insert_rear
2:Delete_front
3:Display_list
4:Exit
enter the choice
```

WRITTEN:---

IMPLEMENTING STACKS:--

program:--

```
#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_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("stack 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;

}

void display(NODE first)

{

    NODE temp;

    if(first==NULL)

        printf("stack empty cannot display items\n");
```

```
for(temp=first;temp!=NULL;temp=temp->link)

{

    printf("%d\n",temp->info);

}

}

int main()

{

    int item,choice,pos;

    NODE first=NULL;

    for(;;)

    {

        printf("\n 1:Insert_front\n 2:Delete_front\n 3:Display_list\n 4: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:display(first);
```

```
    break;
```

```
default:exit(0);
```

```
    break;
```

```
}
```

```
}
```

```
}
```

screenshots:--

Replit - DeepskyblueRedundantTec x +

repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c

Apps Gmail YouTube Maps News Translate SEM HALL TICKET

PremaPrema / DeepskyblueRedunda... Stop Upgrade Share +

Files main.c

```
1 #include<stdio.h>
2 #include<stdlib.h>
3 struct node
4 {
5     int info;
6     struct node *link;
7 };
8 typedef struct node *NODE;
9 NODE getnode()
10 {
11     NODE x;
12     x=(NODE)malloc(sizeof(struct node));
13     if(x==NULL)
14     {
15         printf("mem full\n");
16         exit(0);
17     }
18     return x;
19 }
20 void freenode(NODE x)
21 {
22     free(x);
23 }
```

```
1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
enter the item at front-end
12

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
enter the item at front-end
13

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
1
enter the item at front-end
```

Type here to search

21:01 06-12-2020

Replit - DeepskyblueRedundantTec x +

repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c

Apps Gmail YouTube Maps News Translate SEM HALL TICKET

PremaPrema / DeepskyblueRedunda... Stop Upgrade Share +

Files main.c

```
45 temp=temp->link;
46 printf("item deleted at front-end is=%d\n",
47 first->info);
48 free(first);
49 return temp;
50 }
51 void display(NODE first)
52 {
53     NODE temp;
54     if(first==NULL)
55     printf("stack empty cannot display items\n");
56     for(temp=first;temp!=NULL;temp=temp->link)
57     {
58         printf("%d\n",temp->info);
59     }
60 }
61 int main()
62 {
63     int item,choice,pos;
64     NODE first=NULL;
65     for(;;)
```

```
enter the choice
2
item deleted at front-end is=13

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
2
item deleted at front-end is=12

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
2
stack is empty cannot delete

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
```

Type here to search

21:02 06-12-2020

The screenshot shows a Replit IDE interface. The browser address bar displays `repl.it/@PremaPrema/DeepskyblueRedundantTechnician#main.c`. The Replit header includes the username `PremaPrema`, a `Stop` button, and links for `Upgrade` and `Share`.

The file explorer on the left shows a file named `main.c`. The main editor displays the following C code:

```
64
65 for(;;)
66 {
67     printf("\n 1:Insert_front\n 2:Delete_front\n
68     3:Display_list\n 4:Exit\n");
69     printf("enter the choice\n");
70     scanf("%d",&choice);
71     switch(choice)
72     {
73     case 1:printf("enter the item at front-end\n");
74             scanf("%d",&item);
75             first=insert_front(first,item);
76             break;
77     case 2:first=delete_front(first);
78             break;
79     case 3:display(first);
80             break;
81     default:exit(0);
82     }
83 }
84 }
```

The terminal on the right shows the program's output:

```
2
item deleted at front-end is=12

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
2
stack is empty cannot delete

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
3
stack empty cannot display items

1:Insert_front
2:Delete_front
3:Display_list
4:Exit
enter the choice
```

The Windows taskbar at the bottom shows the search bar and various application icons. The system clock indicates the time is 21:02 on 06-12-2020.

WRITTEN:

## Lab 8

### ① stack implementation.

```
#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-at-end(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;
}
```



DATE \_\_\_\_\_  
PAGE \_\_\_\_\_

```
NODE delete-front(NODE first)
```

```
{
```

```
    NODE temp;
```

```
    if (first == NULL)
```

```
        printf("LE ");
```

```
        return first;
```

```
    }
```

```
    temp = first
```

```
    temp = temp->link;
```

```
    printf("Item deleted @ front end is - %d\n", first->data);
```

```
    free(first);
```

```
    return temp;
```

```
}
```

```
void display(NODE first)
```

```
{ NODE temp;
```

```
    if (first == NULL)
```

```
        printf("LE ");
```

```
        for (temp = first; temp != NULL; temp = temp->link)
```

```
            printf("%d ", temp->data);
```

```
}
```

```
int main()
```

```
{ int item, choice, pos;
```

```
    NODE first = NULL;
```

```
    for(;;)
```

```
{
```

```
    printf("\n 1. insert rear\n 2. Delete-front\n 3. display\n 4. exit\n 5. exit\n");
```

```
    printf("enter the choice: ");
```

```
    switch(choice)
```

```
{
```

Date \_\_\_\_\_  
Page \_\_\_\_\_

```
case 1: pf ("enter the item at rear end")
        sf ("%d", &item);
        front = insert_rear(front, item);
        break;
case 2: front = delete_front(front);
        break;
case 3: display(front);
        break;
default: exit(0);
        break;
}
}
}
```

