SIDDHI SINGH
17BIT0028

ITE – 1004
DATA STRUCTURES AND ALGORITHM
EXERCISE - 6

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

struct node
{
    int info;
    struct node *ptr;
}*front,*rear,*temp,*front1;

int frontelement();
void enq(int data);
void deq();
void empty();
void display();
void create();
void queuesize();

int count = 0;

void main()
{
    int no, ch, e;

    printf("\n 1 - Enque");
    printf("\n 2 - Deque");
    printf("\n 3 - Front element");
    printf("\n 4 - Empty");
    printf("\n 5 - Exit");
    printf("\n 6 - Display");
    printf("\n 7 - Queue size");
    create();
    while (1)
    {
        printf("\n Enter choice : ");
        scanf("%d", &ch);
        switch (ch)
        {
        case 1:
            printf("Enter data : ");
            scanf("%d", &no);
            enq(no);
            break;
        case 2:
            deq();
            break;
        case 3:
```

17BIT0028

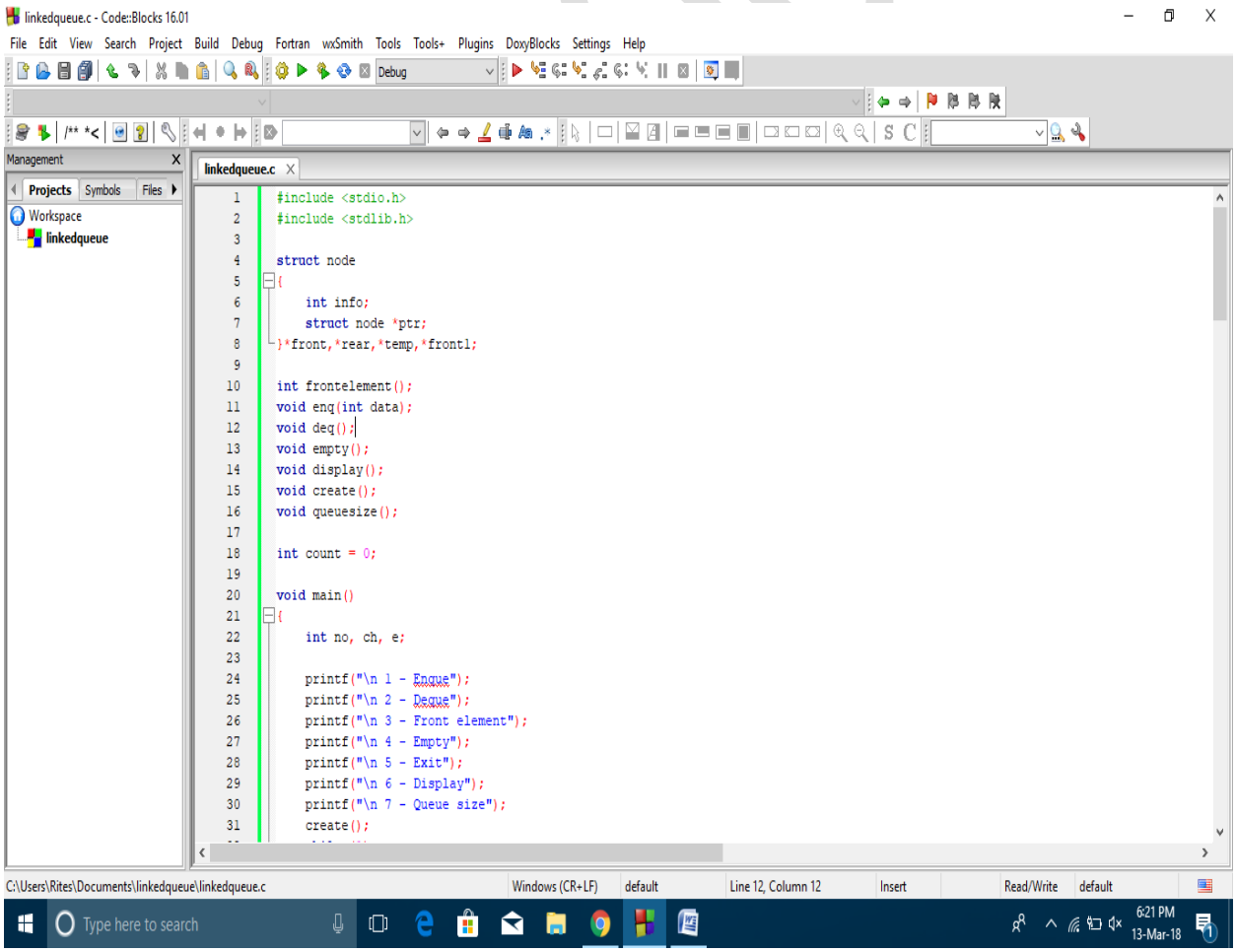
```
            e = frontelement();
            if (e != 0)
                printf("Front element : %d", e);
            else
                printf("\n No front element in Queue as queue is
empty");
            break;
        case 4:
            empty();
            break;
        case 5:
            exit(0);
        case 6:
            display();
            break;
        case 7:
            queuesize();
            break;
        default:
            printf("Wrong choice, Please enter correct choice  ");
            break;
        }
    }
}

/* Create an empty queue */
void create()
{
    front = rear = NULL;
}

/* Returns queue size */
void queuesize()
{
    printf("\n Queue size : %d", count);
}

/* Enqueing the queue */
void enq(int data)
{
    if (rear == NULL)
    {
        rear = (struct node *)malloc(1*sizeof(struct node));
        rear->ptr = NULL;
        rear->info = data;
        front = rear;
    }
    else
    {
        temp=(struct node *)malloc(1*sizeof(struct node));
        rear->ptr = temp;
        temp->info = data;
```

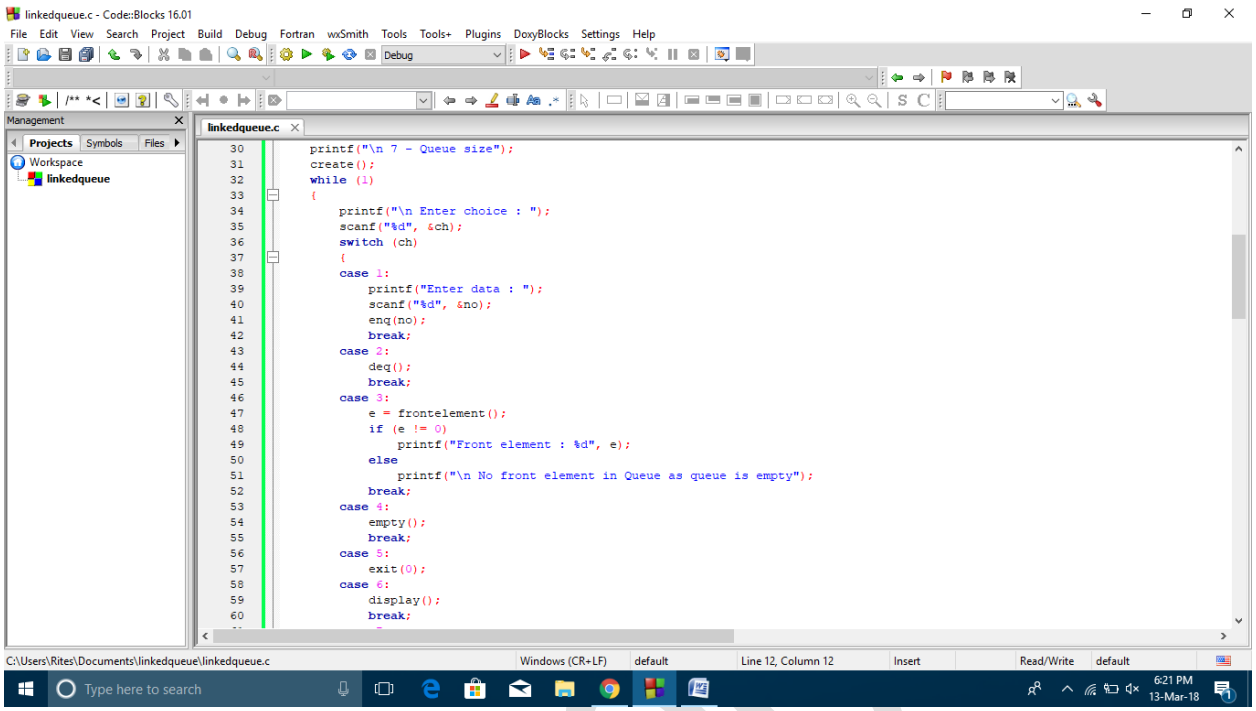
1BIT0028

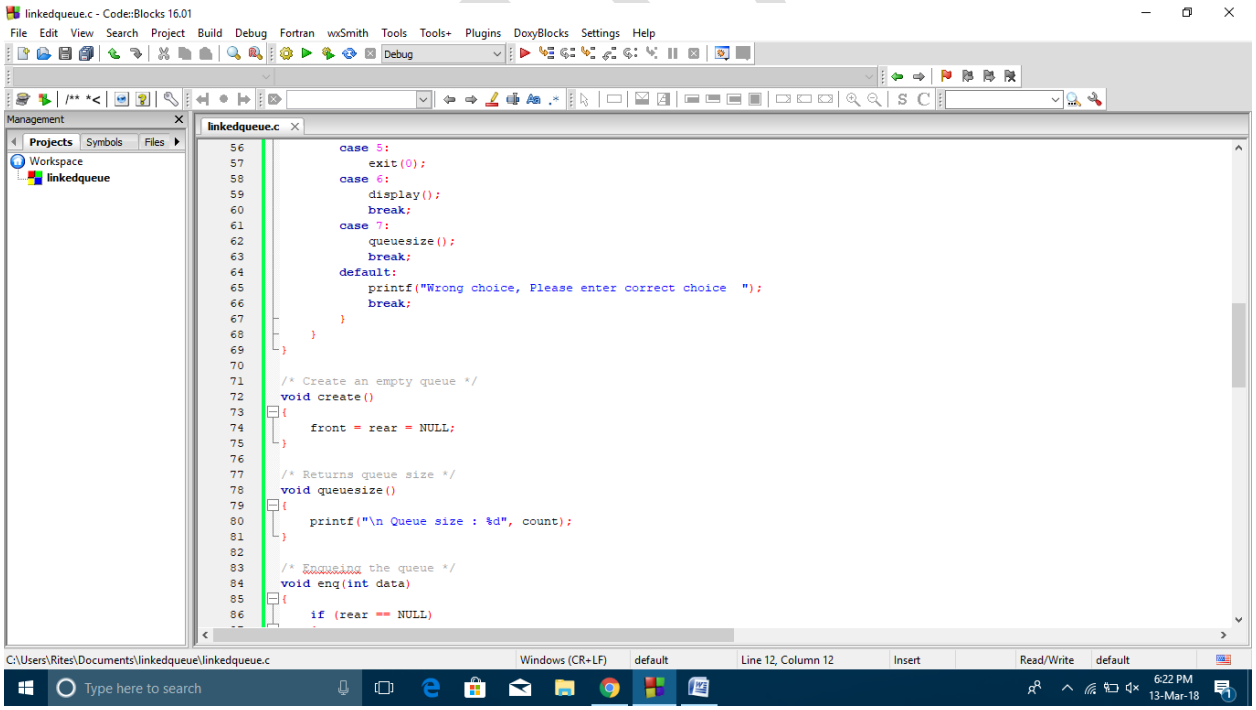
```
        temp->ptr = NULL;

        rear = temp;
    }
    count++;
}

/* Displaying the queue elements */
void display()
{
    front1 = front;

    if ((front1 == NULL) && (rear == NULL))
    {
        printf("Queue is empty");
        return;
    }
    while (front1 != rear)
    {
        printf("%d ", front1->info);
        front1 = front1->ptr;
    }
    if (front1 == rear)
        printf("%d", front1->info);
}

/* Dequeing the queue */
void deq()
{
    front1 = front;

    if (front1 == NULL)
    {
        printf("\n Error: Trying to display elements from empty
queue");
        return;
    }
    else
        if (front1->ptr != NULL)
        {
            front1 = front1->ptr;
            printf("\n Dequed value : %d", front->info);
            free(front);
            front = front1;
        }
        else
        {
            printf("\n Dequed value : %d", front->info);
            free(front);
            front = NULL;
            rear = NULL;
        }
```

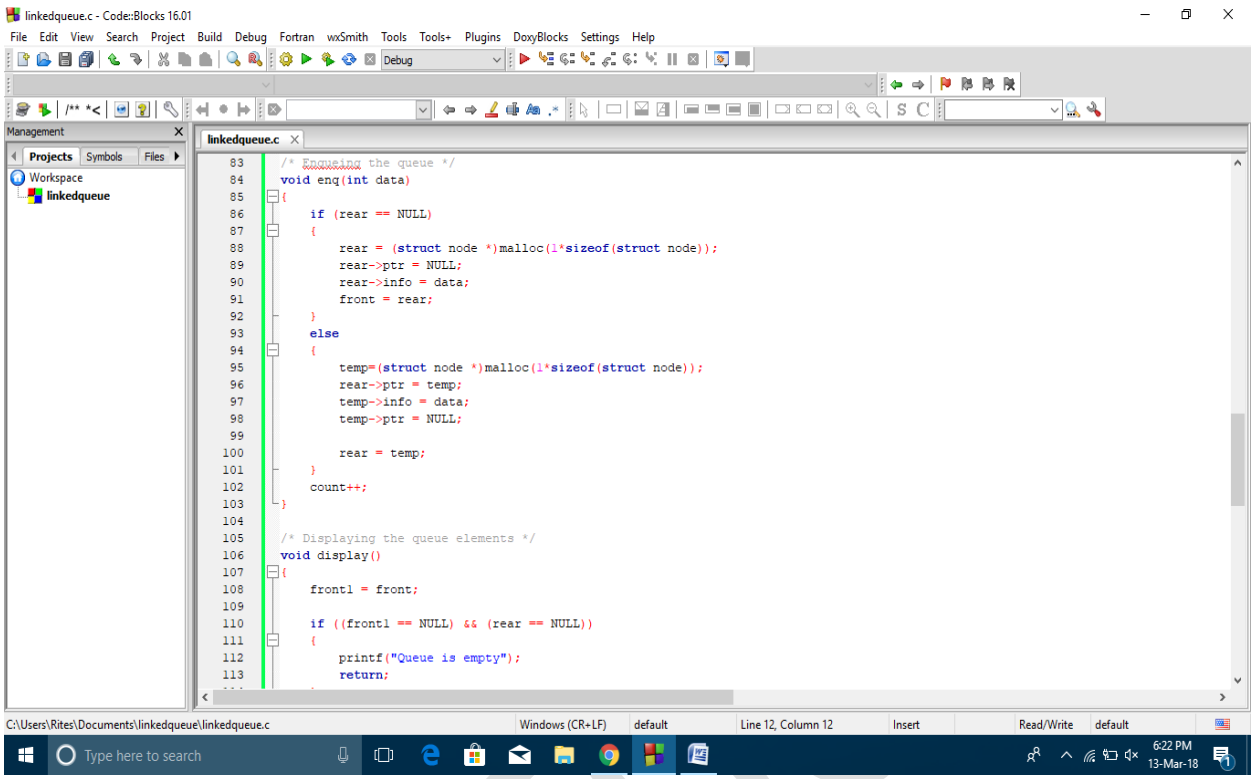
17BIT0028

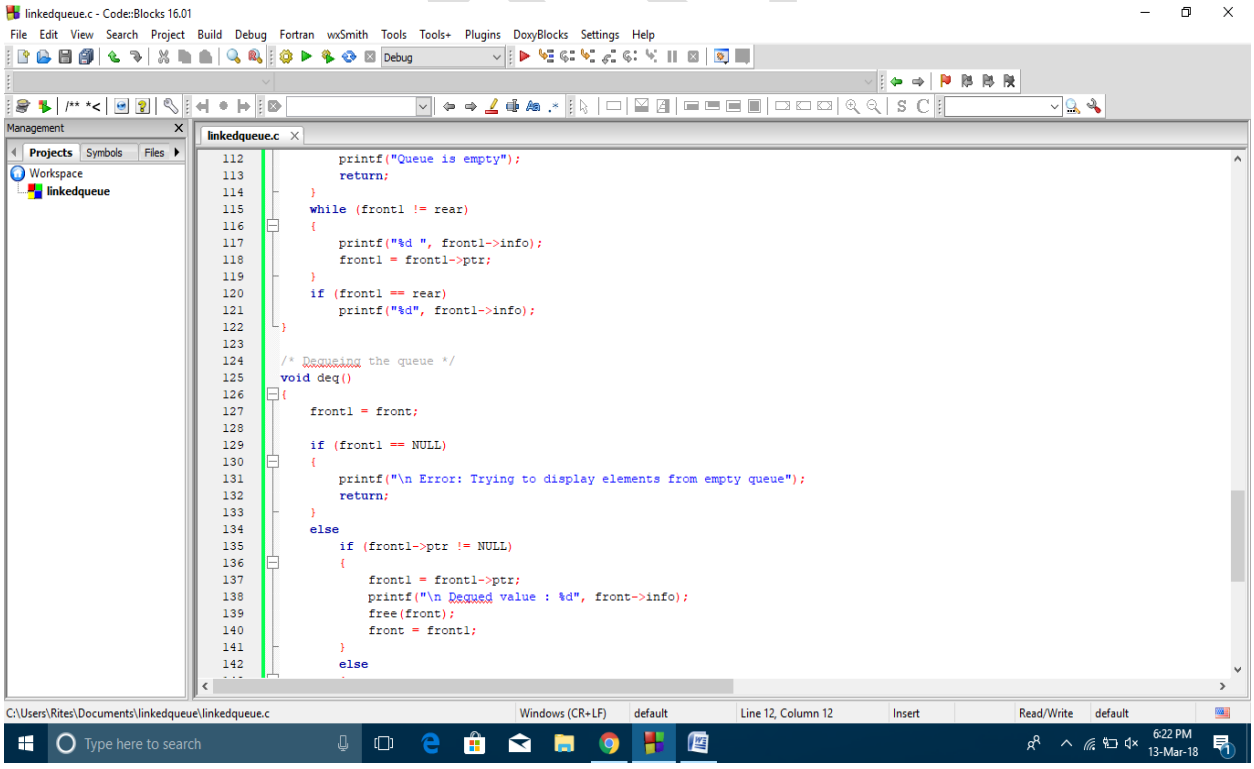
```
        count--;
}

/* Returns the front element of queue */
int frontelement()
{
    if ((front != NULL) && (rear != NULL))
        return(front->info);
    else
        return 0;
}

/* Display if queue is empty or not */
void empty()
{
     if ((front == NULL) && (rear == NULL))
        printf("\n Queue empty");
    else
       printf("Queue not empty");
}
```

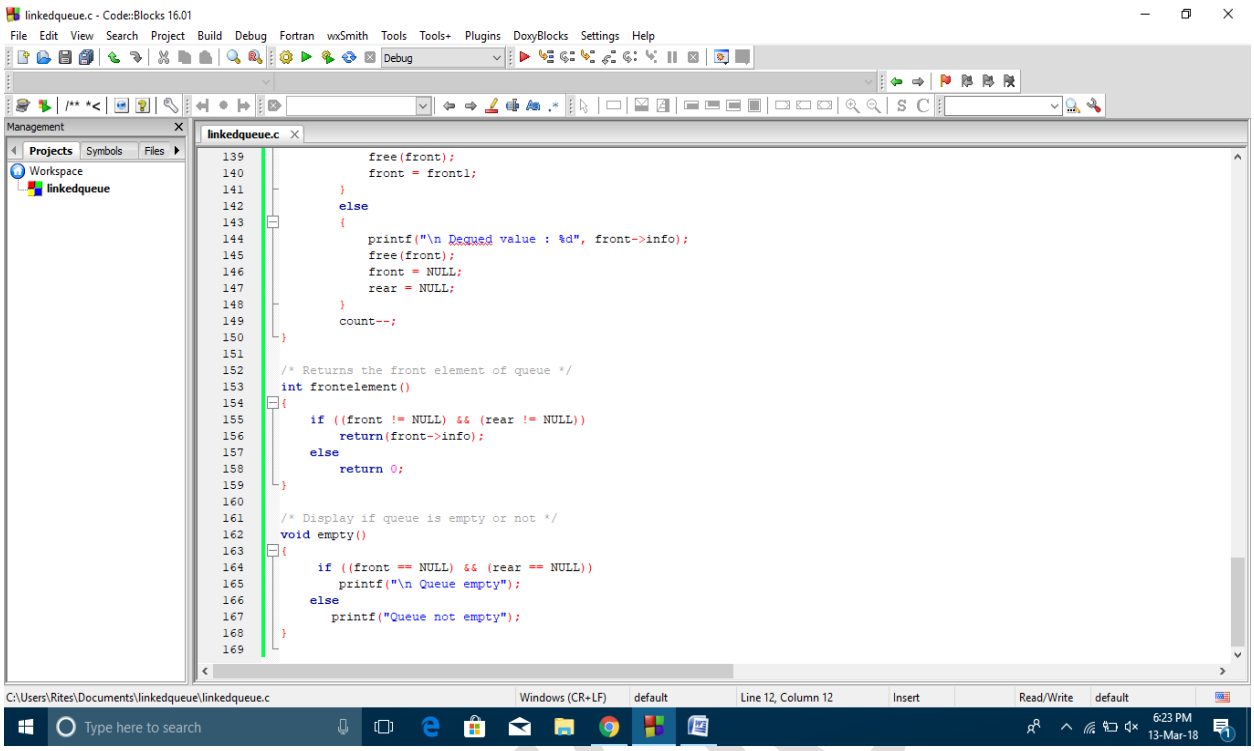



linkedqueue.c - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Management
Projects Symbols Files
Workspace
linkedqueue
linkedqueue.c
printf("\n 7 - Queue size");
create();
while (1)
{
printf("\n Enter choice : ");
scanf("%d", &ch);
switch (ch)
{
case 1:
printf("Enter data : ");
scanf("%d", &no);
enq(no);
break;
case 2:
deq();
break;
case 3:
e = frontelement();
if (e != 0)
printf("Front element : %d", e);
else
printf("\n No front element in Queue as queue is empty");
break;
case 4:
empty();
break;
case 5:
exit(0);
case 6:
display();
break;
C:\Users\Rites\Documents\linkedqueue\linkedqueue.c
Windows (CR+LF)
default
Line 12, Column 12
Insert
Read/Write
default
Type here to search
6:21 PM
13-Mar-18


linkedqueue.c - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Management
Projects Symbols Files
Workspace
linkedqueue
linkedqueue.c
case 5:
exit(0);
case 6:
display();
break;
case 7:
queuesize();
break;
default:
printf("Wrong choice, Please enter correct choice ");
break;
}
}
}
/* Create an empty queue */
void create()
{
front = rear = NULL;
}
/* Returns queue size */
void queuesize()
{
printf("\n Queue size : %d", count);
}
/* Enqueuing the queue */
void enq(int data)
{
if (rear == NULL)
C:\Users\Rites\Documents\linkedqueue\linkedqueue.c
Windows (CR+LF)
default
Line 12, Column 12
Insert
Read/Write
default
Type here to search
6:22 PM
13-Mar-18


linkedqueue.c - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Management
Projects Symbols Files
Workspace
linkedqueue
linkedqueue.c
/* Enqueuing the queue */
void enq(int data)
{
if (rear == NULL)
{
rear = (struct node *)malloc(1*sizeof(struct node));
rear->ptr = NULL;
rear->info = data;
front = rear;
}
else
{
temp=(struct node *)malloc(1*sizeof(struct node));
rear->ptr = temp;
temp->info = data;
temp->ptr = NULL;
rear = temp;
}
count++;
}
/* Displaying the queue elements */
void display()
{
front1 = front;
if ((front1 == NULL) && (rear == NULL))
{
printf("Queue is empty");
return;
C:\Users\Rites\Documents\linkedqueue\linkedqueue.c
Windows (CR+LF)
default
Line 12, Column 12
Insert
Read/Write
default
Type here to search
6:22 PM
13-Mar-18


linkedqueue.c - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
Management
Projects Symbols Files
Workspace
linkedqueue
linkedqueue.c
printf("Queue is empty");
return;
}
while (front1 != rear)
{
printf("%d ", front1->info);
front1 = front1->ptr;
}
if (front1 == rear)
printf("%d", front1->info);
}
/* Dequeuing the queue */
void deq()
{
front1 = front;
if (front1 == NULL)
{
printf("\n Error: Trying to display elements from empty queue");
return;
}
else
if (front1->ptr != NULL)
{
front1 = front1->ptr;
printf("\n Dequed value : %d", front->info);
free(front);
front = front1;
}
else
C:\Users\Rites\Documents\linkedqueue\linkedqueue.c
Windows (CR+LF)
default
Line 12, Column 12
Insert
Read/Write
default
Type here to search
6:22 PM
13-Mar-18



OUTPUT

```
1 - Enque
2 - Deque
3 - Front element
4 - Empty
5 - Exit
6 - Display
7 - Queue size
 Enter choice : 1
Enter data : 12

 Enter choice : 1
Enter data : 28

 Enter choice : 1
Enter data : 23

 Enter choice : 1
Enter data : 43

 Enter choice : 1
Enter data : 67
```

```
 Enter choice : 1
Enter data : 67

 Enter choice : 2

 Dequed value : 12
 Enter choice : 3
Front element : 28
 Enter choice : 6
28 23 43 67
 Enter choice : 7

 Queue size : 4
 Enter choice : 4
Queue not empty
 Enter choice : 5
```

17BIT002