Assignment 13.

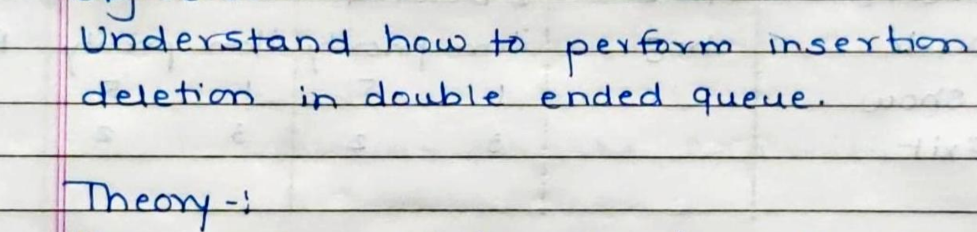
Class - SE IV　　　　　　　　ROll NO - 21430
Batch - F4　　　　　　　　　DOS - 3/12/2020.

Problem statement -:
A double ended queue (dequeue) is a linear list in which additions and deletions may be made at either end. obtain data representation mapping a dequeue into a one-dimensional array. Write C++ program to stimulate deque with functions to add and delete elements from either end of the dequeue.

Objective -:
Understand how to perform insertion and deletion in double ended queue.

Theory -:
Dequeue is a data structure in which elements may be added to or deleted from the front on the rear. Like an ordinary queue, double ended queue is a data structure it supports following operations.

Representation -:



DeQueue is represented in 2 ways :-

1. Input restricted Queue.
2. Output restricted Queue.

1. Operations on Dequeue :-

1. Empty() - Determine if queue is empty().

2. Full () - Determine if queue is full ().

3. enqueueF() - Insert an element at front end of the queue.

4. enqueueR() - Insert an element at rear end of the queue.

5. dequeueR() - Delete rear elements.

6. dequeueF() - Delete ~~rear~~ front elements

* Algorithm/Pseudocode :-

```
Dequeue as an ADT
#define Max 10
class dequeue
{
    int queue [size] ;
    int front, rear ;
    public:
        dequeue()
        {
            front = rear=-1;
        }
        insert_rear() ; //insert from rear end.
        insert_front() ;// insert from front end.
        delete_rear() // delete from rear end.
        delete_front();// delete from front end.
        display() // display dequeue.

}
```

* Procedure insert_rear(int x)

```
    if (front ==(rear+1) % Max )
            Display "Queue Overflow".
    else if (rear==-1)
         front=rear=0
         queue [rear] = x
    else
        rear= (rear+1) % Max ;
        queque [rear]=x ;

END Procedure.
```

```
Procedure insert_front (int x)

    If ( Front ==(rear +1) % Max )
        Display "Queue overflow";
    else if ( front == -1 )
        front = rear = 0
        queue [~~M~~rear] = x
    else
        front = ( front +Max -1 ) % Max
        queue[front] x ;

ENDProcedure

* Procedure delete_rear

    int x
    if( Front == -1 )
        Display "Queue Underflow"
    else if ( Front ==rear)
        x = queue [front]
        front = rear = -1
    else
        x = queue[rear]
        rear = ( rear + Max-1) % Max
    return x

END Procedure
```

* Procedure delete front

```
    int x
    if ( front == -1)
        Display 'Queue Underflow";
    else if ( Front == rear)
        x = queue [front]
        front = rear = -1
    else
        x = queue [ front]
        front = ( front +1) % Max
    return x

END Procedure
```

* Complexity-)

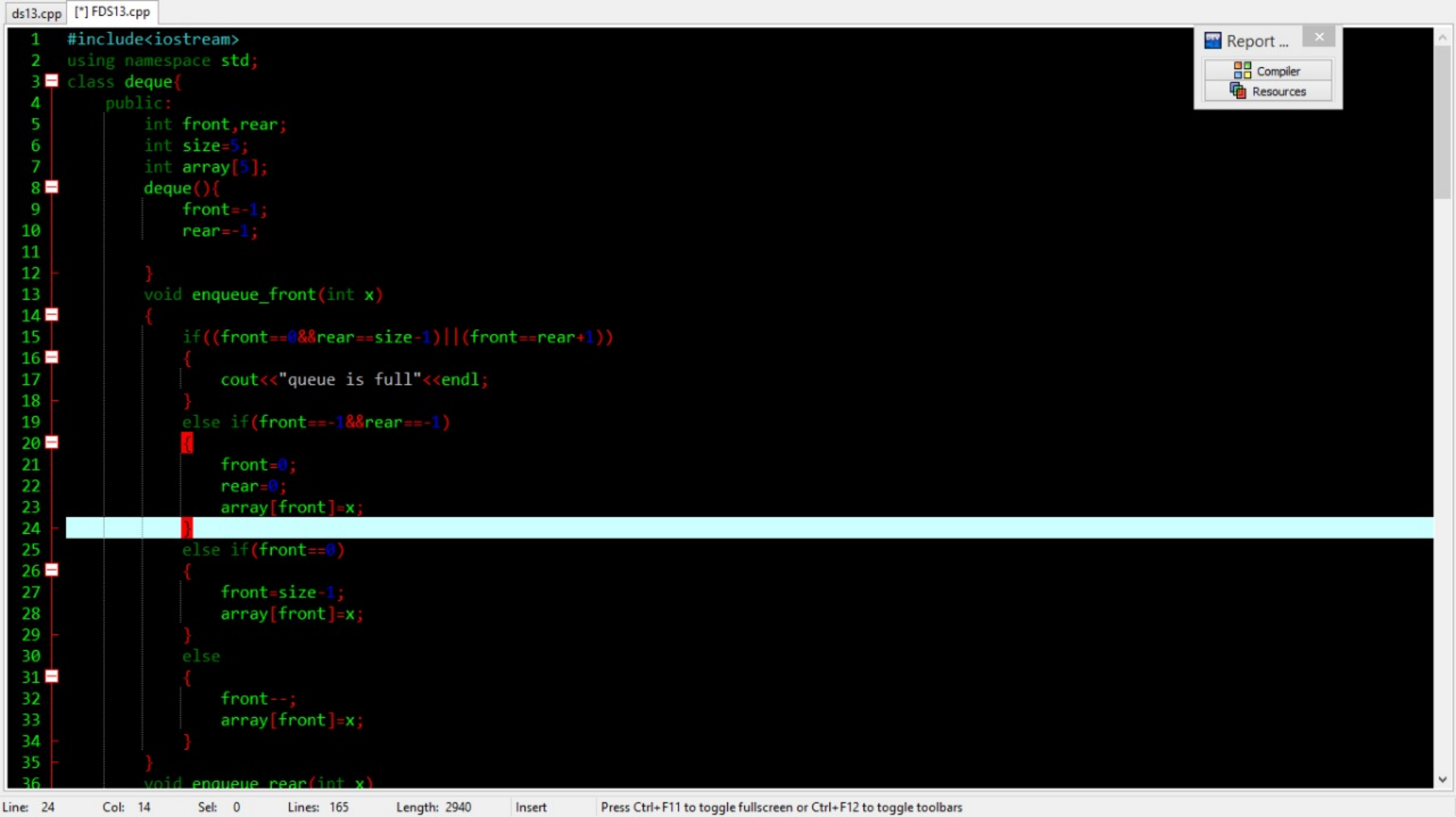
	Function	Time Complexity
1.	insert_front()	o(1)
2.	insert_rear()	O(1)
3.	delete_Front()	o(1)
4.	delete_rear()	o(1)

* Test Cases :-

NO.	Description	Input	Expected o/p	Actual o/p	Result
1.	1.Insert front	ch = 1			
	2.Insert rear	data : 5	10 5 15	10 5 15	
	3.Delete Front	ch = 1			
	4.Delete rear	data : 10			Pass.
	5.Display	ch = 2			
	6.Exit	data : 15			
	ch = 1				
2.	1.Insert front	ch = 3			
	2.Insert rear	10 deleted	5 15	5 15	
	3.Delete front	ch = 5			Pass.
	4.Delet rear				
	5.Display				
	6.Exit				

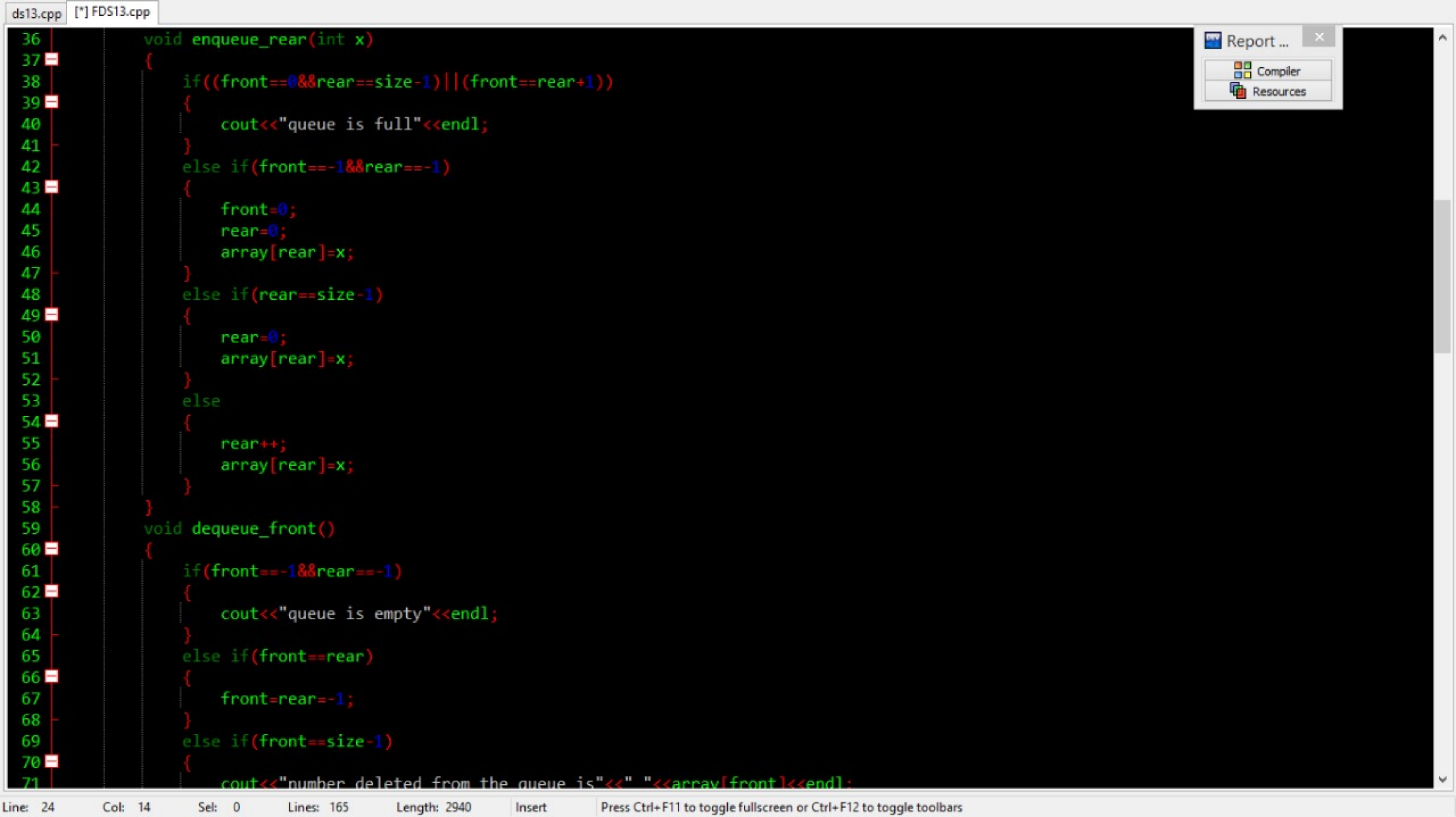
Conclusion

Understand the Concept of how to insert and delete elements in double ended queue.



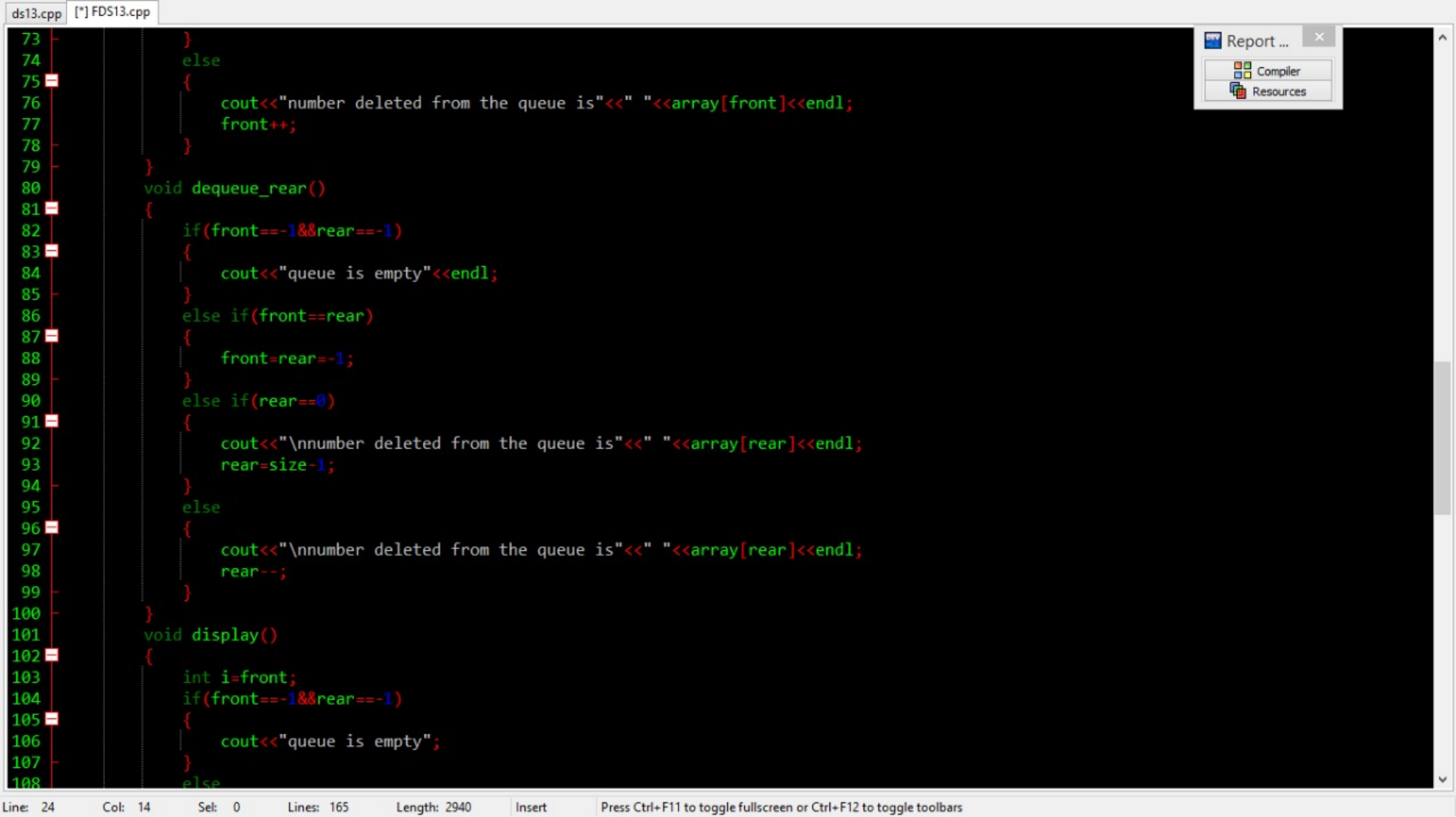
```
#include<iostream>
using namespace std;
class deque{
    public:
        int front,rear;
        int size=5;
        int array[5];
        deque(){
            front=-1;
            rear=-1;

        }
        void enqueue_front(int x)
        {
            if((front==0&&rear==size-1)||(front==rear+1))
            {
                cout<<"queue is full"<<endl;
            }
            else if(front==-1&&rear==-1)
            {
                front=0;
                rear=0;
                array[front]=x;
            }
            else if(front==0)
            {
                front=size-1;
                array[front]=x;
            }
            else
            {
                front--;
                array[front]=x;
            }
        }
        void enqueue_rear(int x)
```

Line: 24 Col: 14 Sel: 0 Lines: 165 Length: 2940 Insert Press Ctrl+F11 to toggle fullscreen or Ctrl+F12 to toggle toolbars



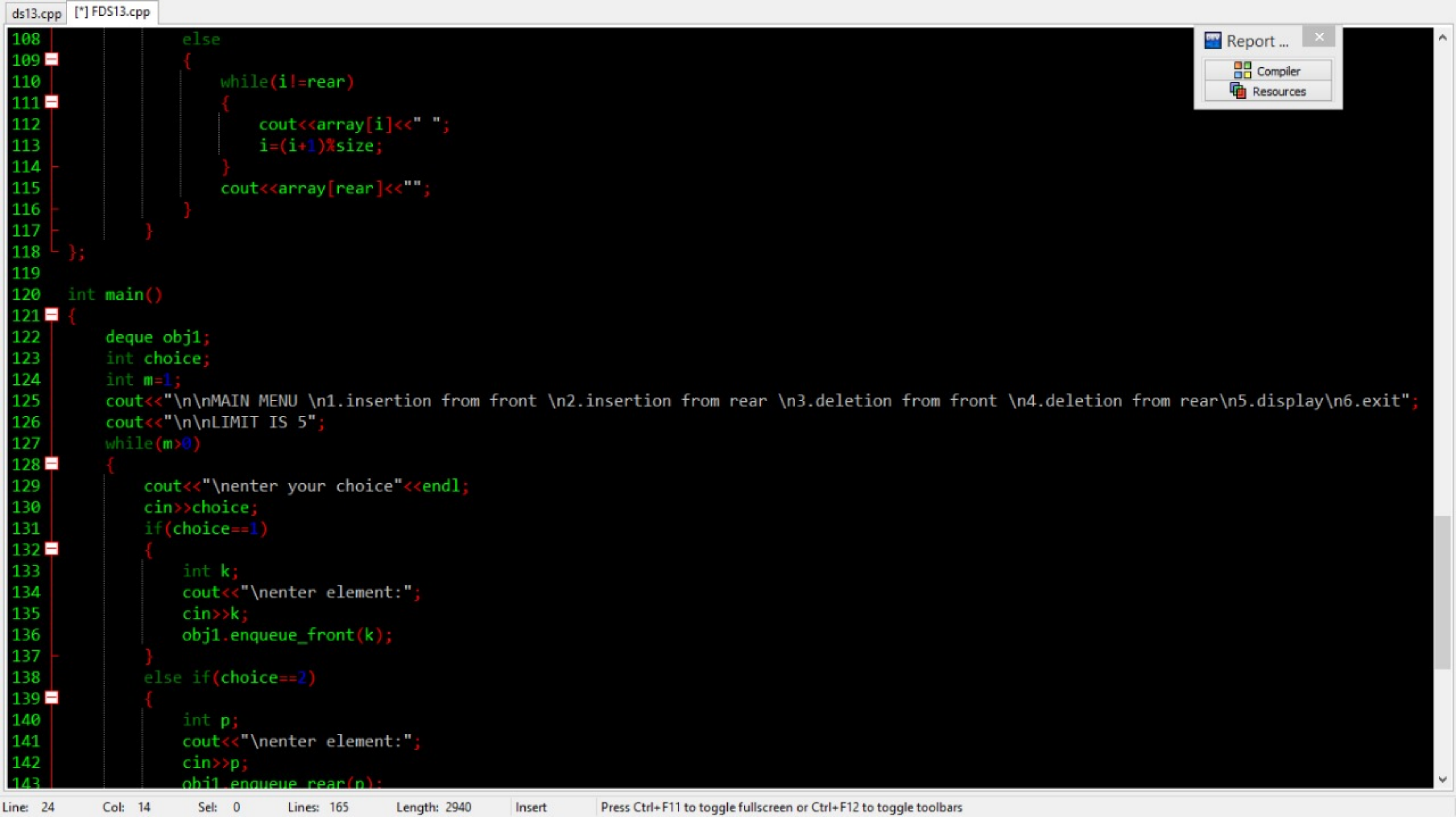
```cpp
        void enqueue_rear(int x)
        {
            if((front==0&&rear==size-1)||(front==rear+1))
            {
                cout<<"queue is full"<<endl;
            }
            else if(front==-1&&rear==-1)
            {
                front=0;
                rear=0;
                array[rear]=x;
            }
            else if(rear==size-1)
            {
                rear=0;
                array[rear]=x;
            }
            else
            {
                rear++;
                array[rear]=x;
            }
        }
        void dequeue_front()
        {
            if(front==-1&&rear==-1)
            {
                cout<<"queue is empty"<<endl;
            }
            else if(front==rear)
            {
                front=rear=-1;
            }
            else if(front==size-1)
            {
                cout<<"number deleted from the queue is"<<" "<<array[front]<<endl;
```

Line: 24 Col: 14 Sel: 0 Lines: 165 Length: 2940 Insert Press Ctrl+F11 to toggle fullscreen or Ctrl+F12 to toggle toolbars



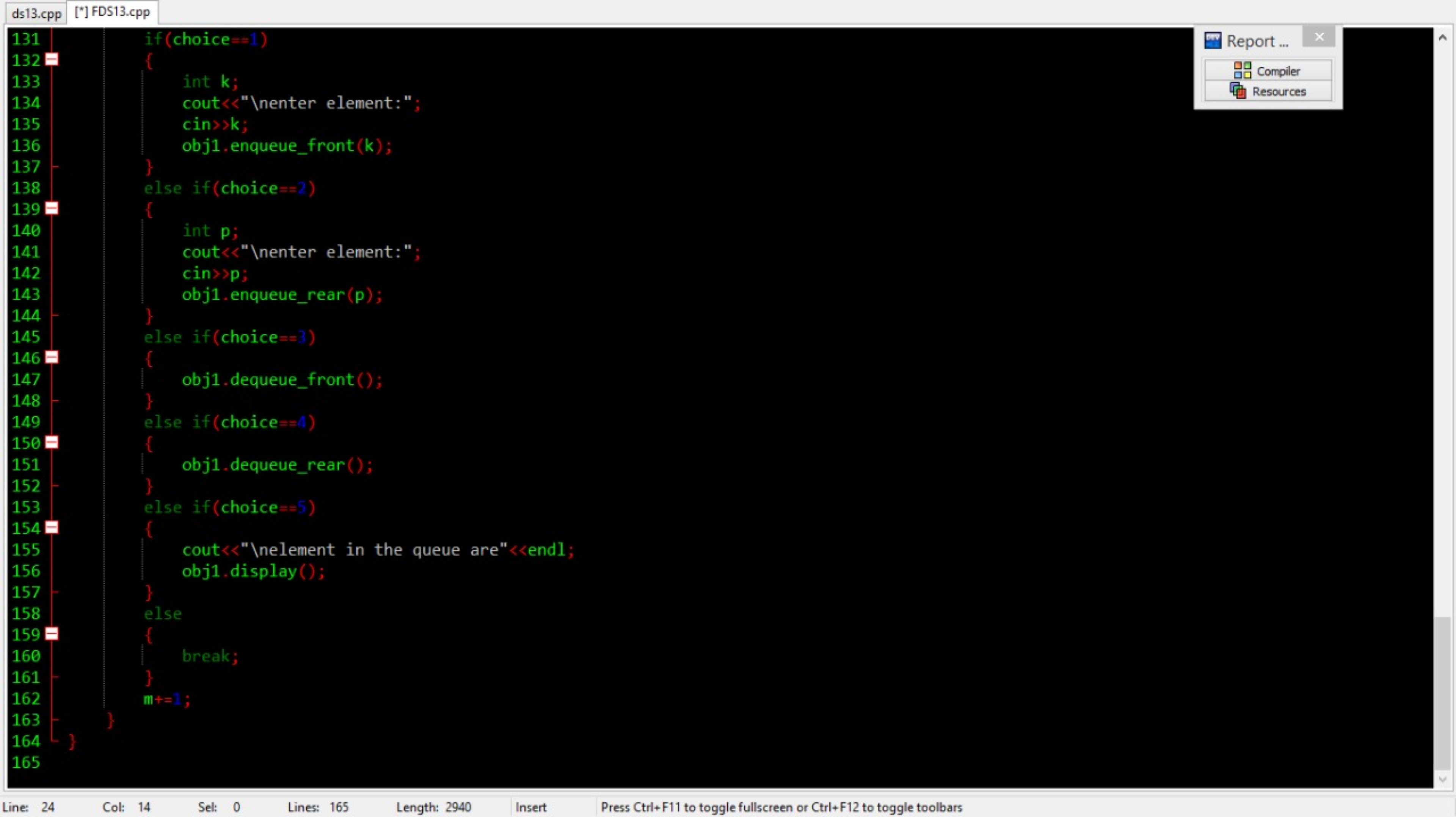
```
            }
            else
            {
                cout<<"number deleted from the queue is"<<" "<<array[front]<<endl;
                front++;
            }
        }
        void dequeue_rear()
        {
            if(front==-1&&rear==-1)
            {
                cout<<"queue is empty"<<endl;
            }
            else if(front==rear)
            {
                front=rear=-1;
            }
            else if(rear==0)
            {
                cout<<"\nnumber deleted from the queue is"<<" "<<array[rear]<<endl;
                rear=size-1;
            }
            else
            {
                cout<<"\nnumber deleted from the queue is"<<" "<<array[rear]<<endl;
                rear--;
            }
        }
        void display()
        {
            int i=front;
            if(front==-1&&rear==-1)
            {
                cout<<"queue is empty";
            }
            else
```

Line: 24 Col: 14 Sel: 0 Lines: 165 Length: 2940 Insert Press Ctrl+F11 to toggle fullscreen or Ctrl+F12 to toggle toolbars



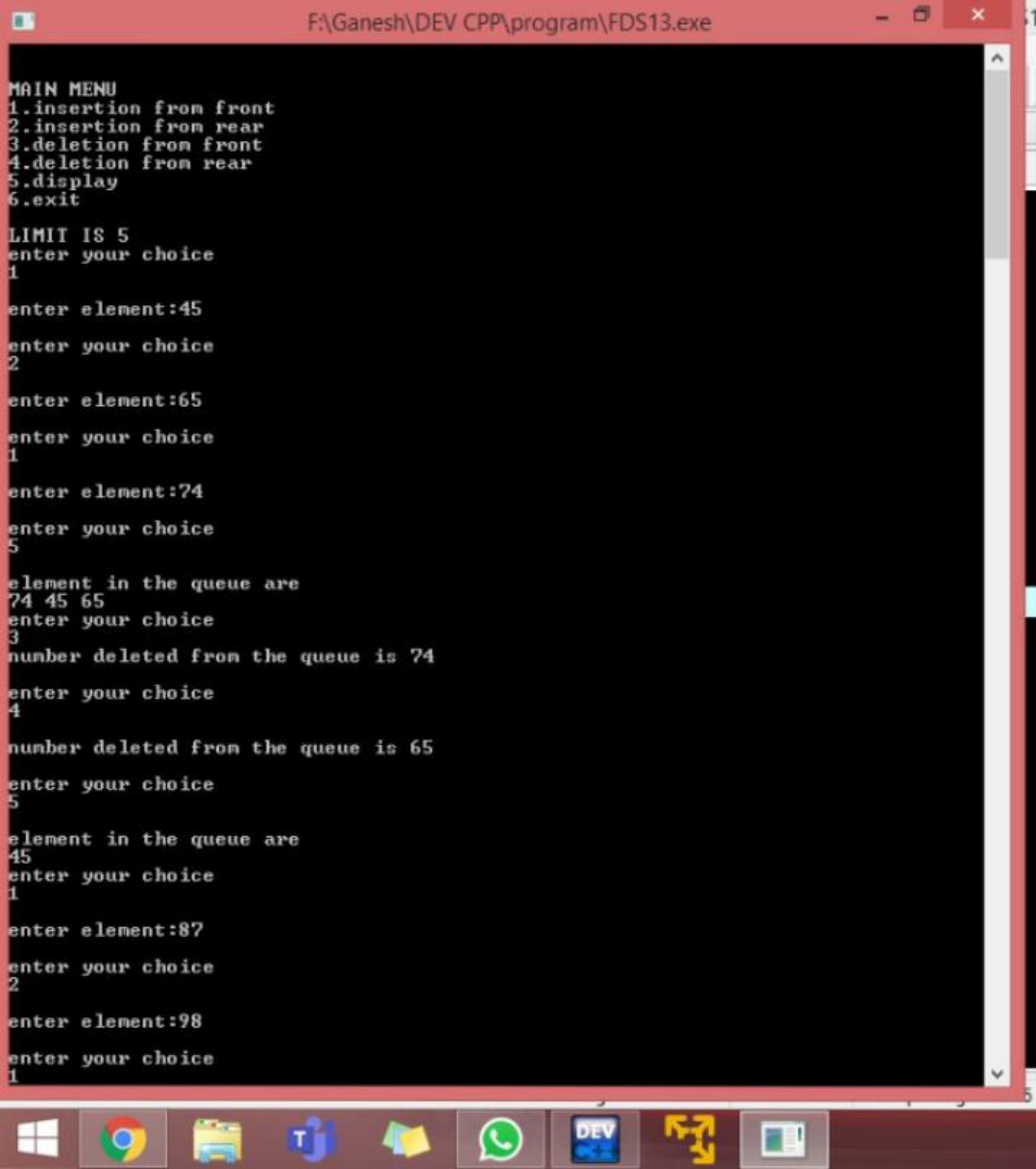
```cpp
            else
            {
                while(i!=rear)
                {
                    cout<<array[i]<<" ";
                    i=(i+1)%size;
                }
                cout<<array[rear]<<"";
            }
        }
};

int main()
{
    deque obj1;
    int choice;
    int m=1;
    cout<<"\n\nMAIN MENU \n1.insertion from front \n2.insertion from rear \n3.deletion from front \n4.deletion from rear\n5.display\n6.exit";
    cout<<"\n\nLIMIT IS 5";
    while(m>0)
    {
        cout<<"\nenter your choice"<<endl;
        cin>>choice;
        if(choice==1)
        {
            int k;
            cout<<"\nenter element:";
            cin>>k;
            obj1.enqueue_front(k);
        }
        else if(choice==2)
        {
            int p;
            cout<<"\nenter element:";
            cin>>p;
            obj1.enqueue_rear(p);
```

Line: 24 Col: 14 Sel: 0 Lines: 165 Length: 2940 Insert Press Ctrl+F11 to toggle fullscreen or Ctrl+F12 to toggle toolbars

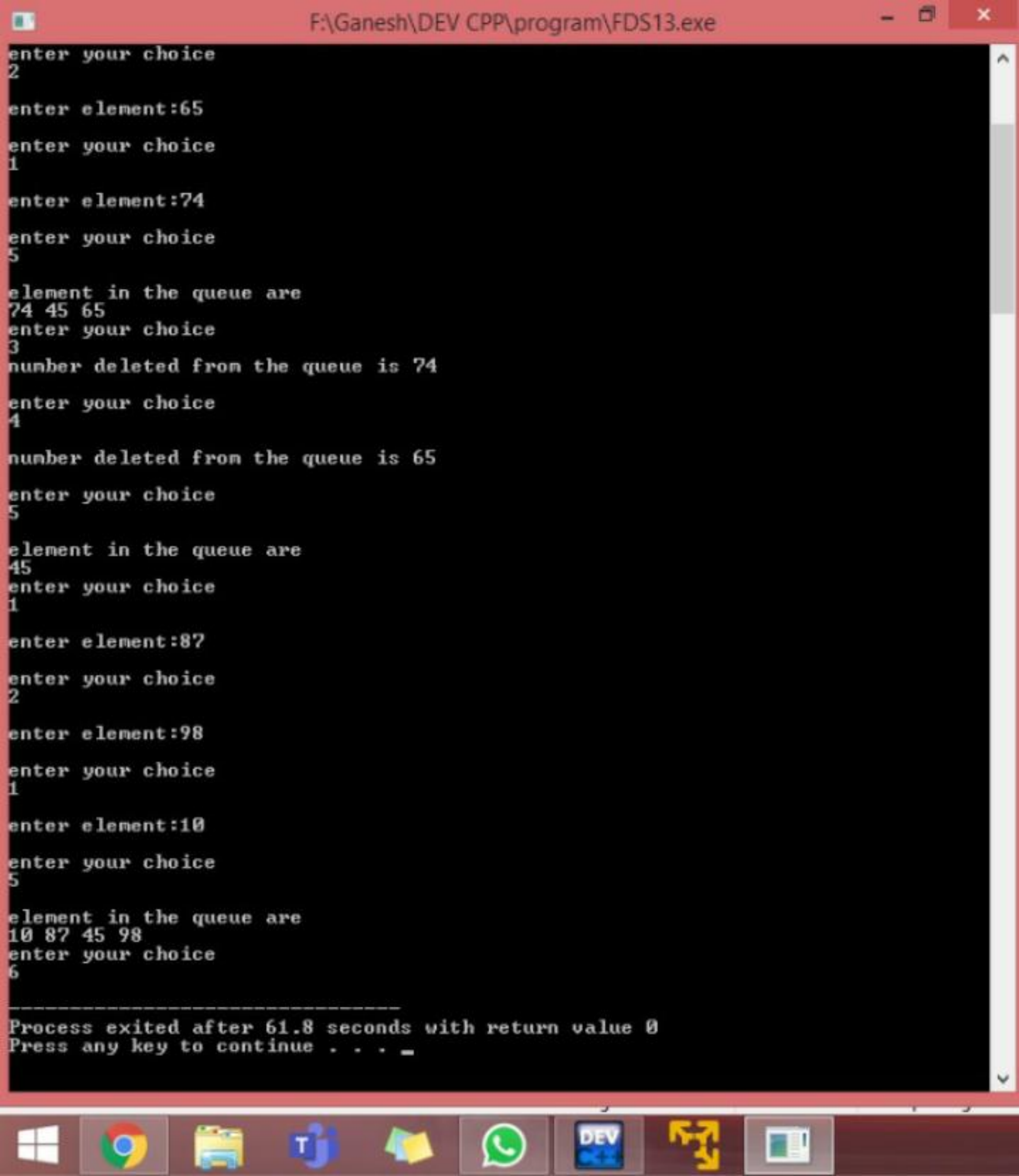


```cpp
            if(choice==1)
            {
                int k;
                cout<<"\nenter element:";
                cin>>k;
                obj1.enqueue_front(k);
            }
            else if(choice==2)
            {
                int p;
                cout<<"\nenter element:";
                cin>>p;
                obj1.enqueue_rear(p);
            }
            else if(choice==3)
            {
                obj1.dequeue_front();
            }
            else if(choice==4)
            {
                obj1.dequeue_rear();
            }
            else if(choice==5)
            {
                cout<<"\nelement in the queue are"<<endl;
                obj1.display();
            }
            else
            {
                break;
            }
            m+=1;
        }
    }

```

Line: 24   Col: 14   Sel: 0   Lines: 165   Length: 2940   Insert   Press Ctrl+F11 to toggle fullscreen or Ctrl+F12 to toggle toolbars



```
MAIN MENU
1.insertion from front
2.insertion from rear
3.deletion from front
4.deletion from rear
5.display
6.exit

LIMIT IS 5
enter your choice
1

enter element:45

enter your choice
2

enter element:65

enter your choice
1

enter element:74

enter your choice
5

element in the queue are
74 45 65
enter your choice
3
number deleted from the queue is 74

enter your choice
4

number deleted from the queue is 65

enter your choice
5

element in the queue are
45
enter your choice
1

enter element:87

enter your choice
2

enter element:98

enter your choice
1
```



```
enter your choice
2

enter element:65

enter your choice
1

enter element:74

enter your choice
5

element in the queue are
74 45 65
enter your choice
3
number deleted from the queue is 74

enter your choice
4

number deleted from the queue is 65

enter your choice
5

element in the queue are
45
enter your choice
1

enter element:87

enter your choice
2

enter element:98

enter your choice
1

enter element:10

enter your choice
5

element in the queue are
10 87 45 98
enter your choice
6

--------------------------------
Process exited after 61.8 seconds with return value 0
Press any key to continue . . . _
```