

Kishan Vaghamashi

202312014

Question 1:

Code:

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("What do you want to choose\n 1. Stack\n 2. Queue");
        int choiceMain = sc.nextInt();

        if(choiceMain == 1){
            System.out.print("Set size of the stack:- ");
            Integer n = sc.nextInt();
            stack2 s=new stack2(n);

            while(true){
                System.out.println("1. Add element to stack.");
                System.out.println("2. Remove top element from stack.");
                System.out.println("3. Display the top element of the stack.");
                System.out.println("4. Display all the elements currently present in the stack.");
                System.out.println("5. Check if the stack is empty or not(underflow).");
                System.out.println("6. Check if the stack is full or not(overflow).");
                System.out.println("7. No. of elements currently present in the stack.");
                System.out.println("8. EXIT.");

                System.out.print("Enter your choice:- ");
                Integer choice = sc.nextInt();

                System.out.println("\n=====");
                switch (choice){
                    case 1:
                        System.out.print("Enter element:- ");
                        Integer num = sc.nextInt();
                        s.push(num);
                        break;

                    case 2:
                        s.pop();
                        break;

                    case 3:
                        System.out.println(s.peek());
```

```
        break;

    case 4:
        s.display();
        break;

    case 5:
        if(s.empty()){
            System.out.println("Stack is empty.");
        }else {
            System.out.println("Stack is not empty.");
        }
        break;

    case 6:
        if(s.full()){
            System.out.println("Stack is full.");
        }else {
            System.out.println("Stack is not full.");
        }
        break;

    case 7:
        System.out.println("Count => " + s.count());
        break;

    case 8:
        return;
    }
    System.out.println("\n=====n");
}
}else{
    System.out.print("Set size of the queue:- ");
    Integer n = sc.nextInt();
    queue q = new queue(n);

    while(true){
        System.out.println("1. Add element to queue.");
        System.out.println("2. Remove top element from queue.");
        System.out.println("3. Display the top element of the queue.");
        System.out.println("4. Display all the elements currently present in the queue.");
        System.out.println("5. Check if the queue is empty or not(underflow).");
        System.out.println("6. Check if the queue is full or not(overflow).");
        System.out.println("7. No. of elements currently present in the queue.");
```

```
System.out.println("8. EXIT.");

System.out.print("Enter your choice:- ");
Integer choice = sc.nextInt();

System.out.println("\n=====\\n");
switch (choice){
    case 1:
        System.out.print("Enter element:- ");
        Integer num = sc.nextInt();
        q.enqueue(num);
        break;

    case 2:
        q.dequeue();
        break;

    case 3:
        System.out.println(q.front());
        break;

    case 4:
        q.display();
        break;

    case 5:
        if(q.empty()){
            System.out.println("Queue is empty.");
        }else {
            System.out.println("Queue is not empty.");
        }
        break;

    case 6:
        if(q.full()){
            System.out.println("Queue is full.");
        }else {
            System.out.println("Queue is not full.");
        }
        break;

    case 7:
        System.out.println("Count => " + q.count());
        break;
```

Kishan Vaghamashi
202312014

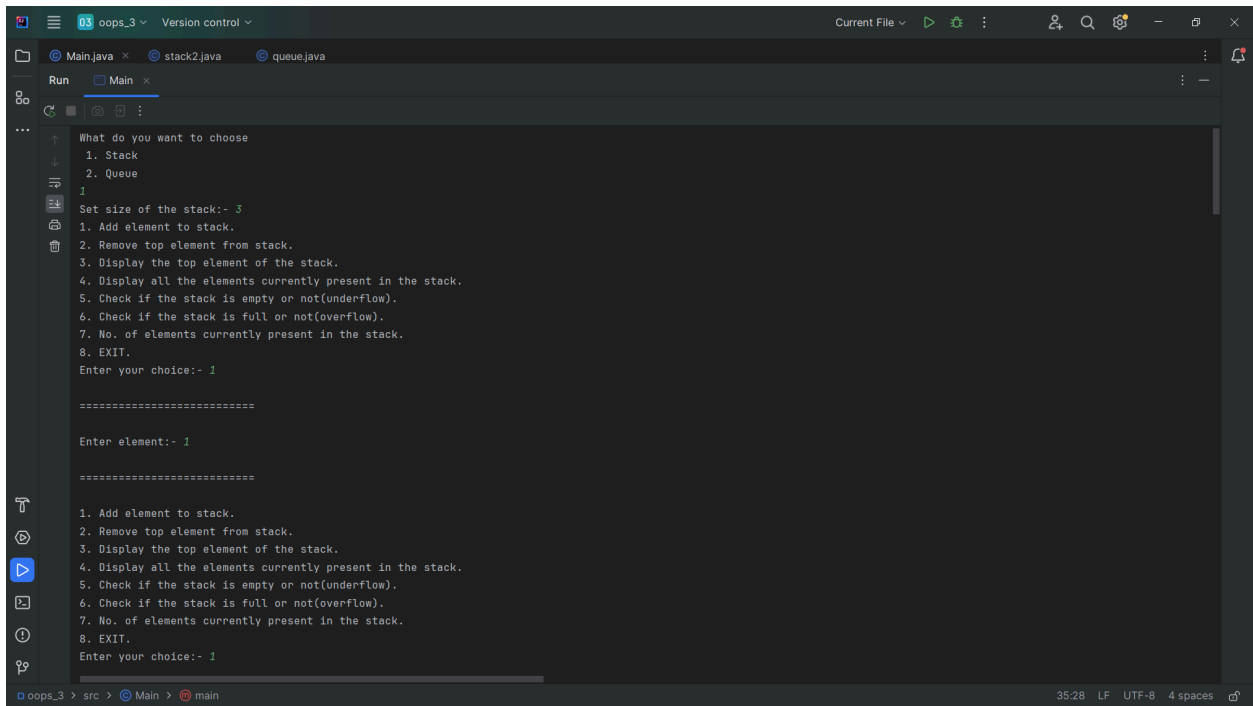
```
        case 8:
            return;
        }
        System.out.println("\n=====");
    }
}
}
```

Kishan Vaghamashi

202312014

Output:

Stack



```
oops_3  Version control
Main.java stack2.java queue.java
Run Main
What do you want to choose
1. Stack
2. Queue
1
Set size of the stack:- 3
1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 1

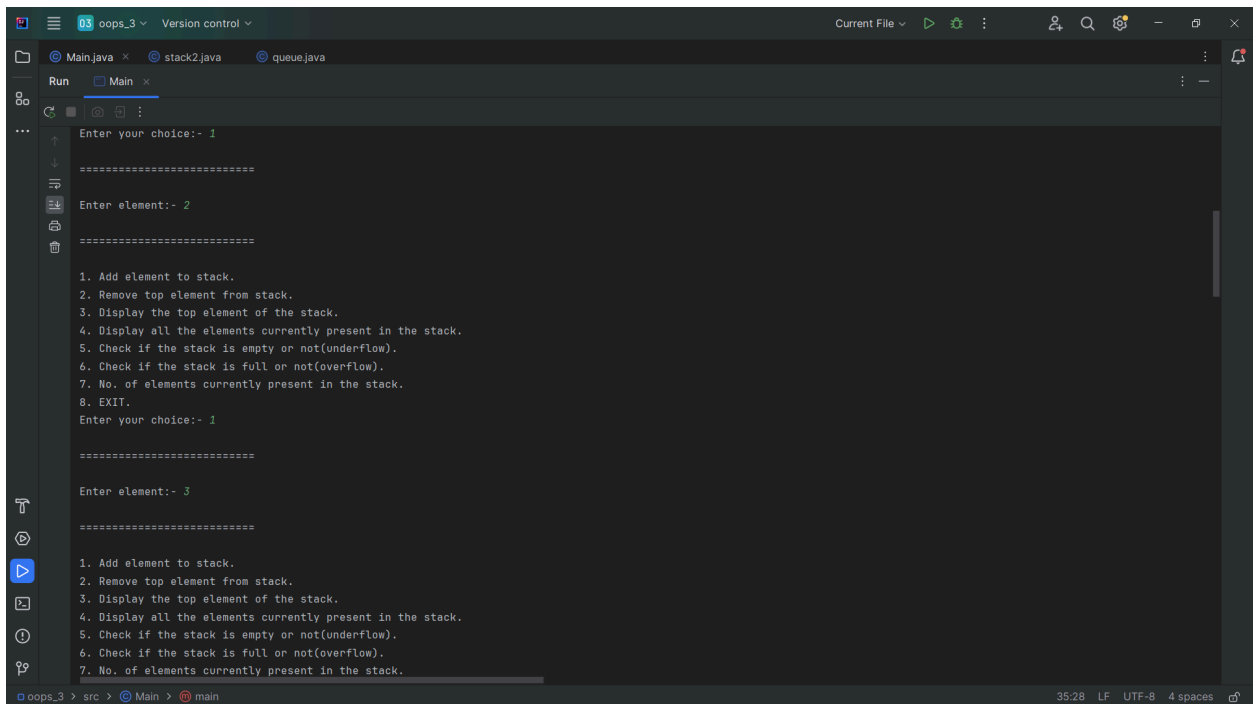
=====

Enter element:- 1

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 1

oops_3 > src > Main > main 35:28 LF UTF-8 4 spaces
```



```
oops_3  Version control
Main.java stack2.java queue.java
Run Main
Enter your choice:- 1

=====

Enter element:- 2

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 1

=====

Enter element:- 3

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.

oops_3 > src > Main > main 35:28 LF UTF-8 4 spaces
```

202312014

The screenshot shows an IDE with a Java file named `oops_3.java`. The code implements a stack with the following menu:

```

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:-

```

The execution output shows the following sequence of events:

- User enters choice 1 (Add element).
- User enters element 4.
- Stack is full (due to a hard-coded limit of 5).
- User enters choice 3 (Display top element).
- Output: 4.
- User enters choice 4 (Display all elements).
- Output: 1. Add element to stack. 2. Remove top element from stack. 3. Display the top element of the stack. 4. Display all the elements currently present in the stack.

The screenshot shows an IDE window titled 'oops_3' with a 'Version control' dropdown. The main editor displays a Java program for a stack. The program has a menu with 8 options: 4. Display all the elements currently present in the stack, 5. Check if the stack is empty or not(underflow), 6. Check if the stack is full or not(overflow), 7. No. of elements currently present in the stack, 8. EXIT. The user has entered choice 4, and the program has displayed the stack contents: 1, 2, 3. The user has entered choice 2, and the program has displayed the updated stack contents: 1. The IDE interface includes a top bar with 'oops_3' and 'Version control', a left sidebar with icons for Explorer, Run, and Source, and a bottom status bar showing 'oops_3 > src > Main > main' and '35:28 LF UTF-8 4 spaces'.

```

4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 4

=====

1
2
3

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 2

=====

=====

1. Add element to stack.

```

Kishan Vaghamashi
202312014

```
oops_3 Version control
Main.java stack2.java queue.java
Run Main
1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 4

=====

1
2

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not(underflow).
6. Check if the stack is full or not(overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 5

=====

Stack is not empty.
```

oops_3 > src > Main > main 35:28 LF UTF-8 4 spaces

Kishan Vaghamashi
202312014

```
oops_3 Version control
Main.java stack2.java queue.java
Run Main
1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not (underflow).
6. Check if the stack is full or not (overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 5

=====

Stack is not empty.

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not (underflow).
6. Check if the stack is full or not (overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 6

=====

Stack is not full.
```

oops_3 > src > Main > main 35:28 LF UTF-8 4 spaces

```
oops_3 Version control
Main.java stack2.java queue.java
Run Main
1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not (underflow).
6. Check if the stack is full or not (overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 7

=====

Count => 2

=====

1. Add element to stack.
2. Remove top element from stack.
3. Display the top element of the stack.
4. Display all the elements currently present in the stack.
5. Check if the stack is empty or not (underflow).
6. Check if the stack is full or not (overflow).
7. No. of elements currently present in the stack.
8. EXIT.
Enter your choice:- 8

=====

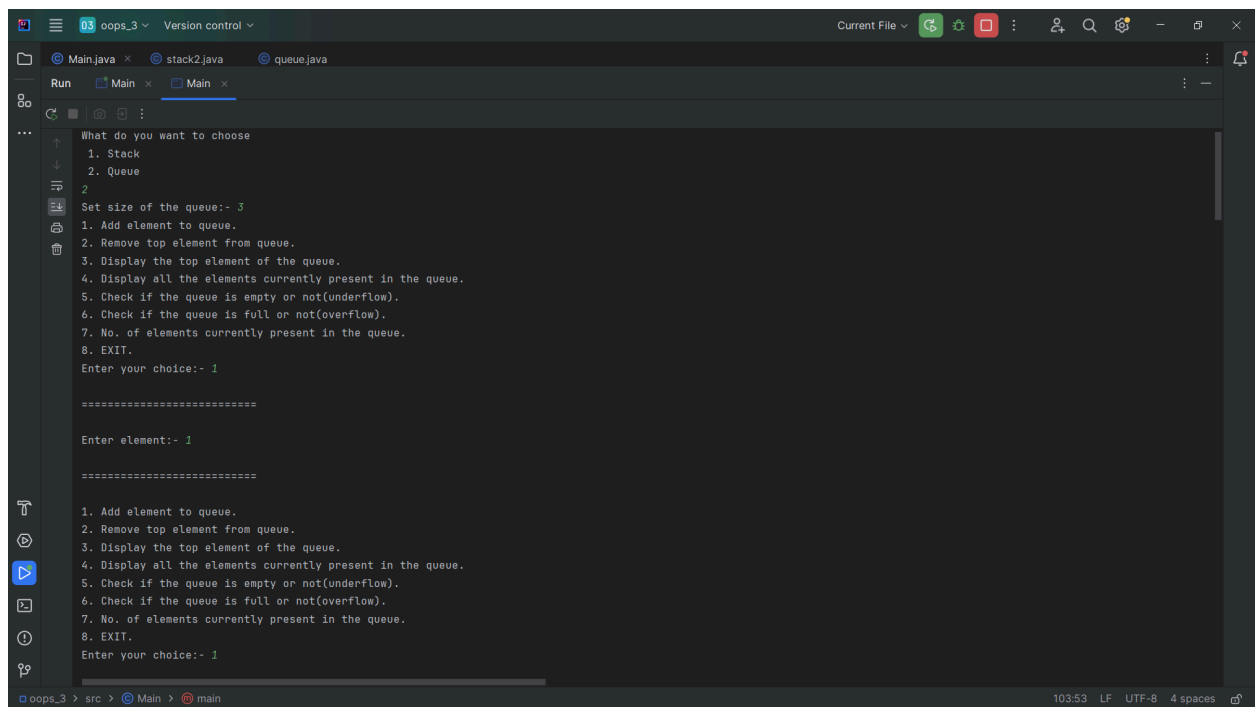
Process finished with exit code 0
```

oops_3 > src > Main > main 35:28 LF UTF-8 4 spaces

Kishan Vaghamashi

202312014

Queue:



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
What do you want to choose
1. Stack
2. Queue
2
Set size of the queue:- 3
1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 1

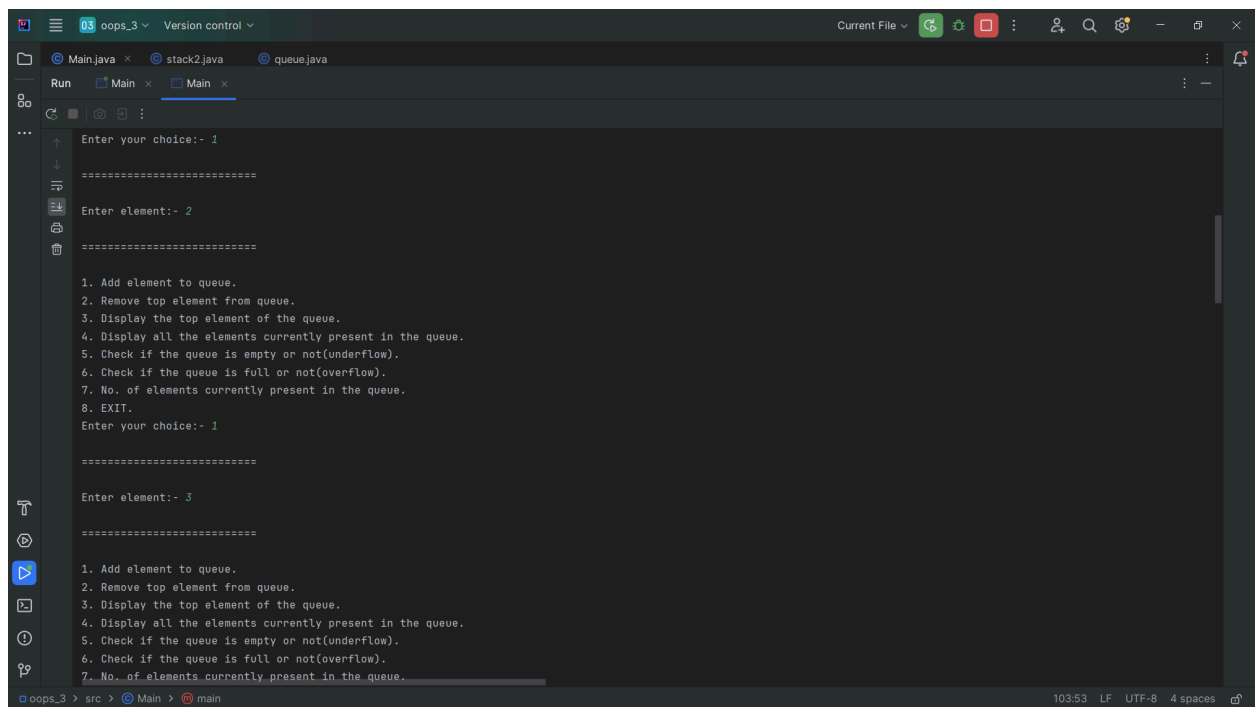
=====

Enter element:- 1

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 1

oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces
```



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
Enter your choice:- 1

=====

Enter element:- 2

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 1

=====

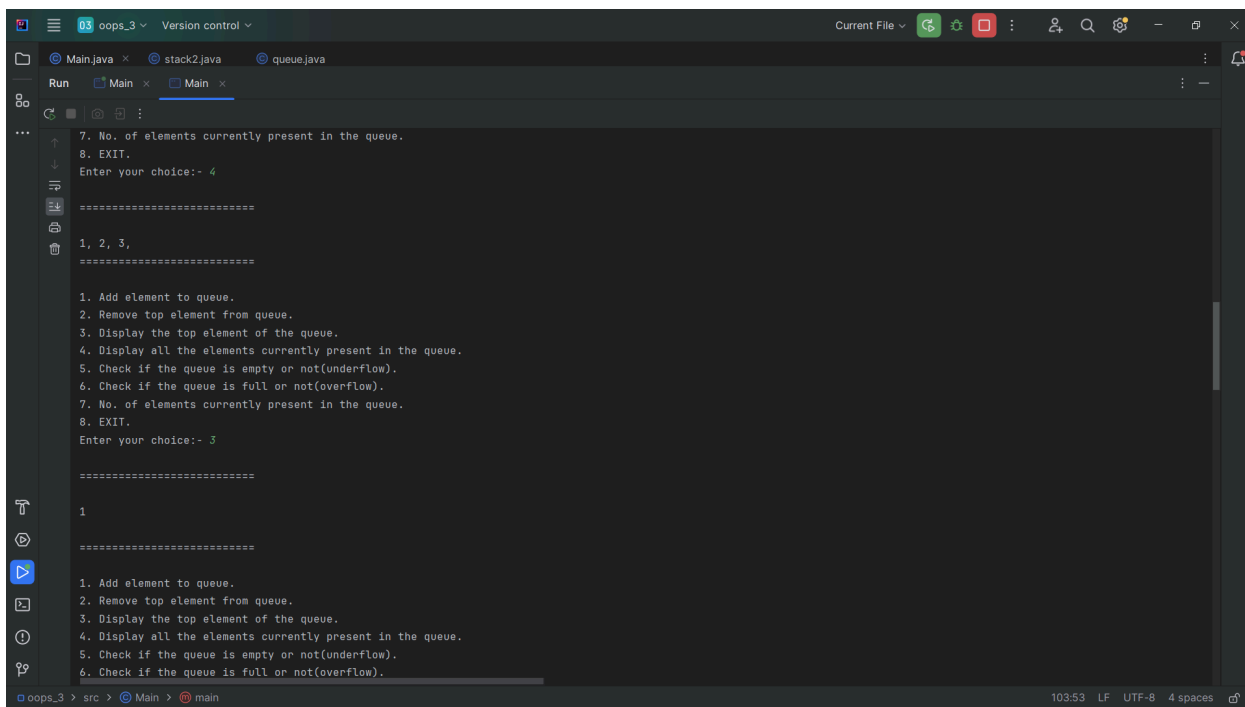
Enter element:- 3

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.

oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces
```

Kishan Vaghamashi
202312014



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 4

=====

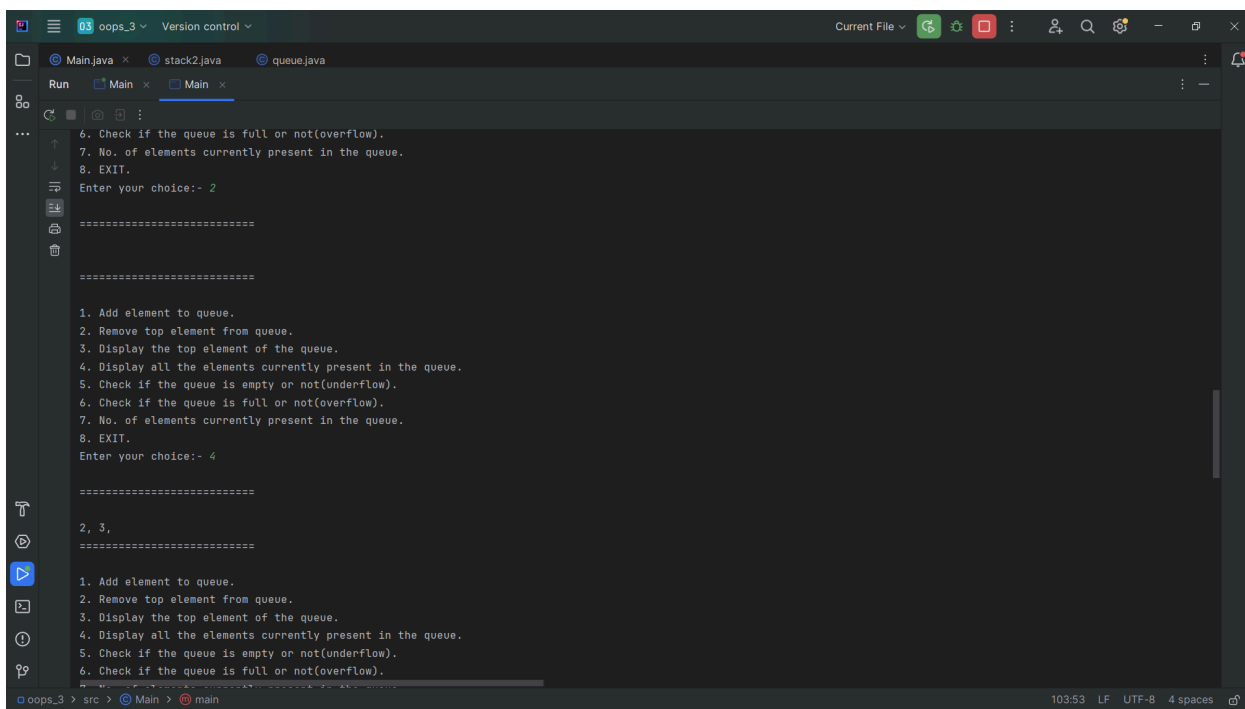
1, 2, 3,
=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 3

=====

1
=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces
```



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 2

=====

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 4

=====

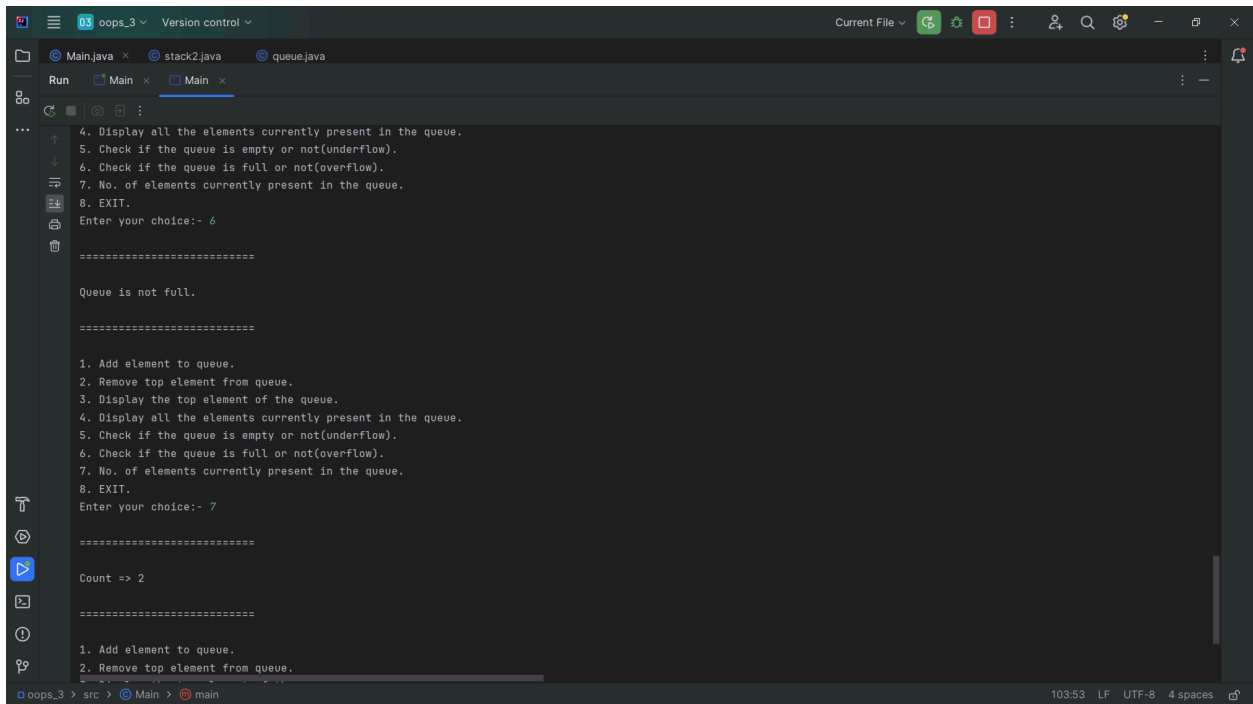
2, 3,
=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces
```

Kishan Vaghamashi
202312014

```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 4
=====
2, 3,
=====
1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 5
=====
Queue is not empty.
=====
1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces
```

Kishan Vaghamashi
202312014



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 6

=====

Queue is not full.

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 7

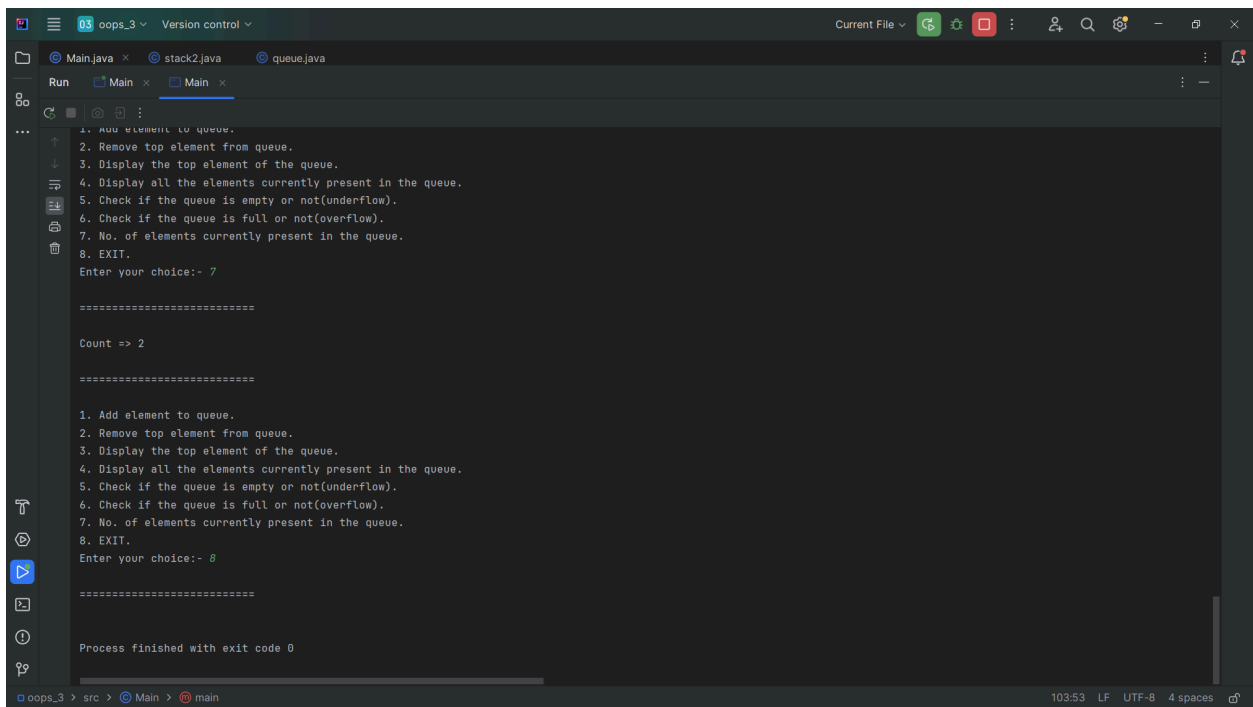
=====

Count => 2

=====

1. Add element to queue.
2. Remove top element from queue.
```

oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces



```
oops_3 Version control
Main.java stack2.java queue.java
Run Main Main
1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 7

=====

Count => 2

=====

1. Add element to queue.
2. Remove top element from queue.
3. Display the top element of the queue.
4. Display all the elements currently present in the queue.
5. Check if the queue is empty or not(underflow).
6. Check if the queue is full or not(overflow).
7. No. of elements currently present in the queue.
8. EXIT.
Enter your choice:- 8

=====

Process finished with exit code 0
```

oops_3 > src > Main > main 103:53 LF UTF-8 4 spaces