9/12/2024

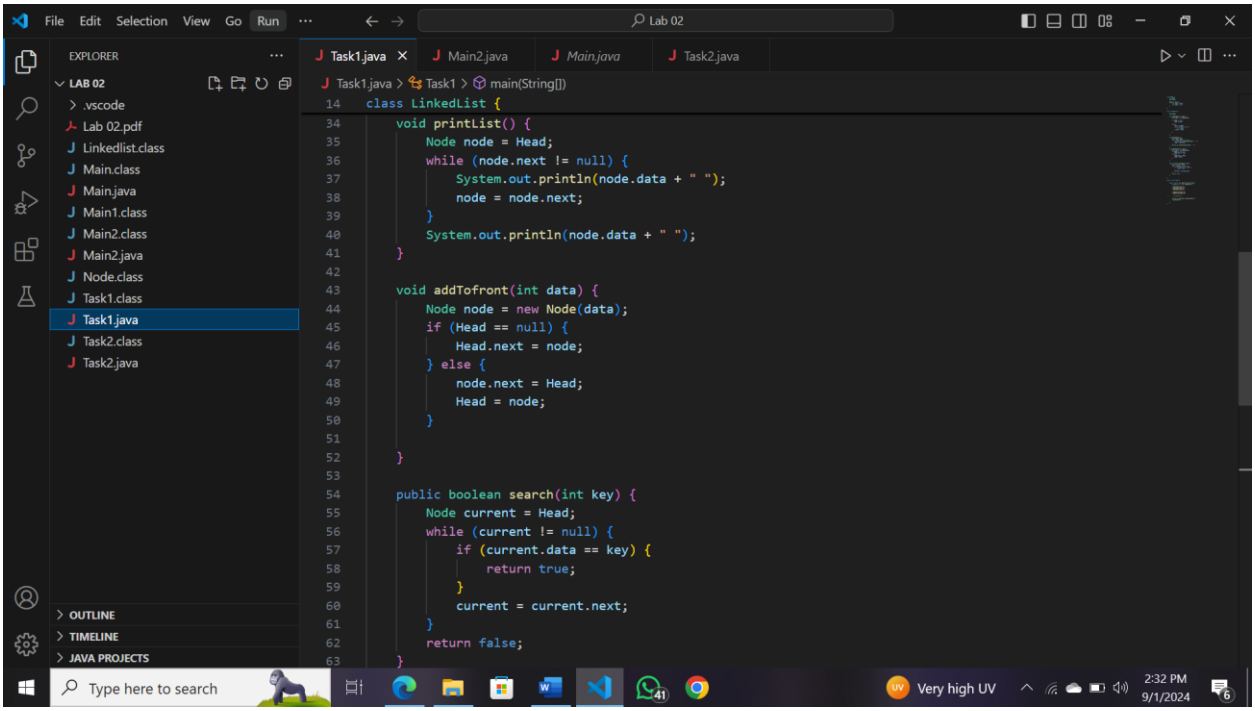
LAB O4 Exercise

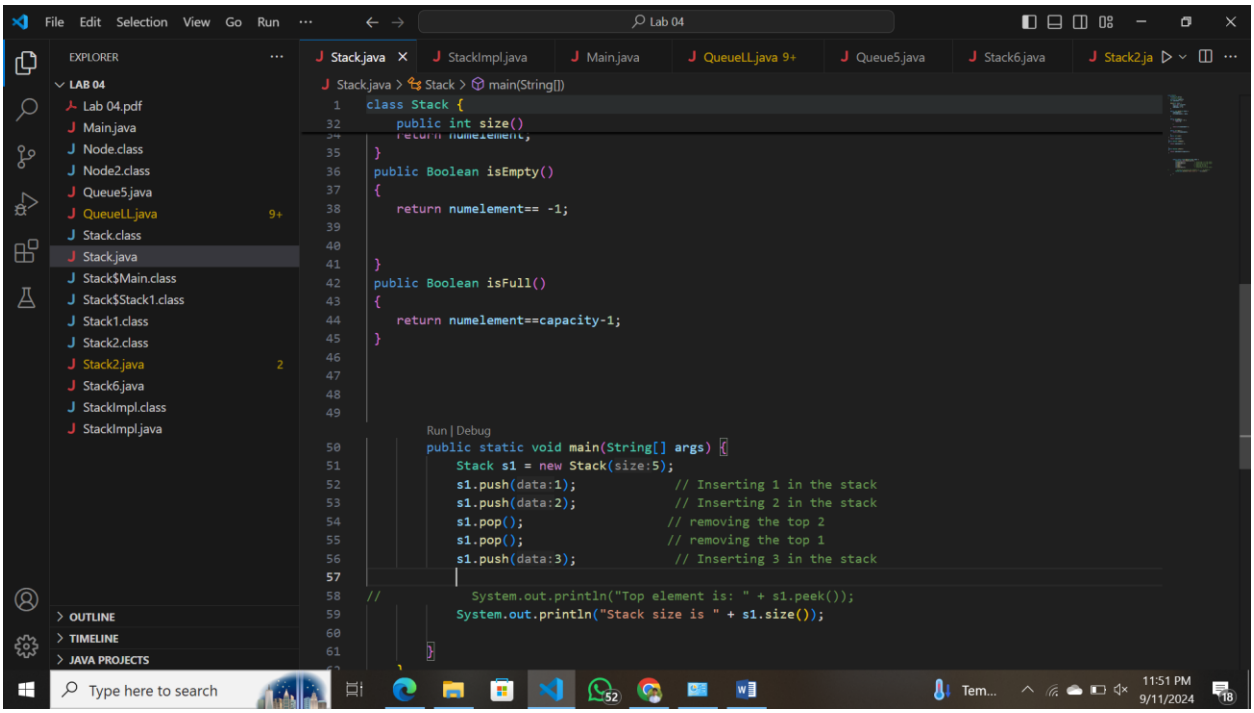
Faheem Akbar

CMS ID: 023-23-0365

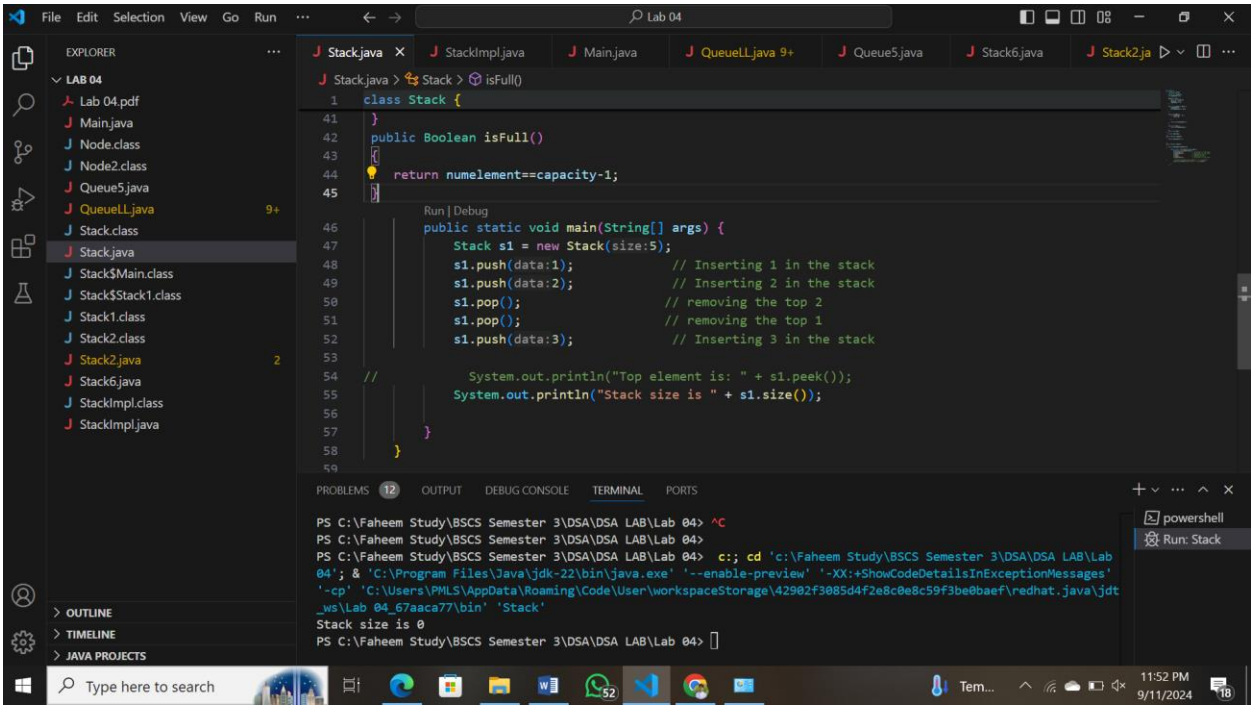
DSA LAB 04 || SUBMITTED TO MA'AM MARINA GUL

1. Stack using array: Understand provided code and implement all required methods in Stack. Stack Code is given below:

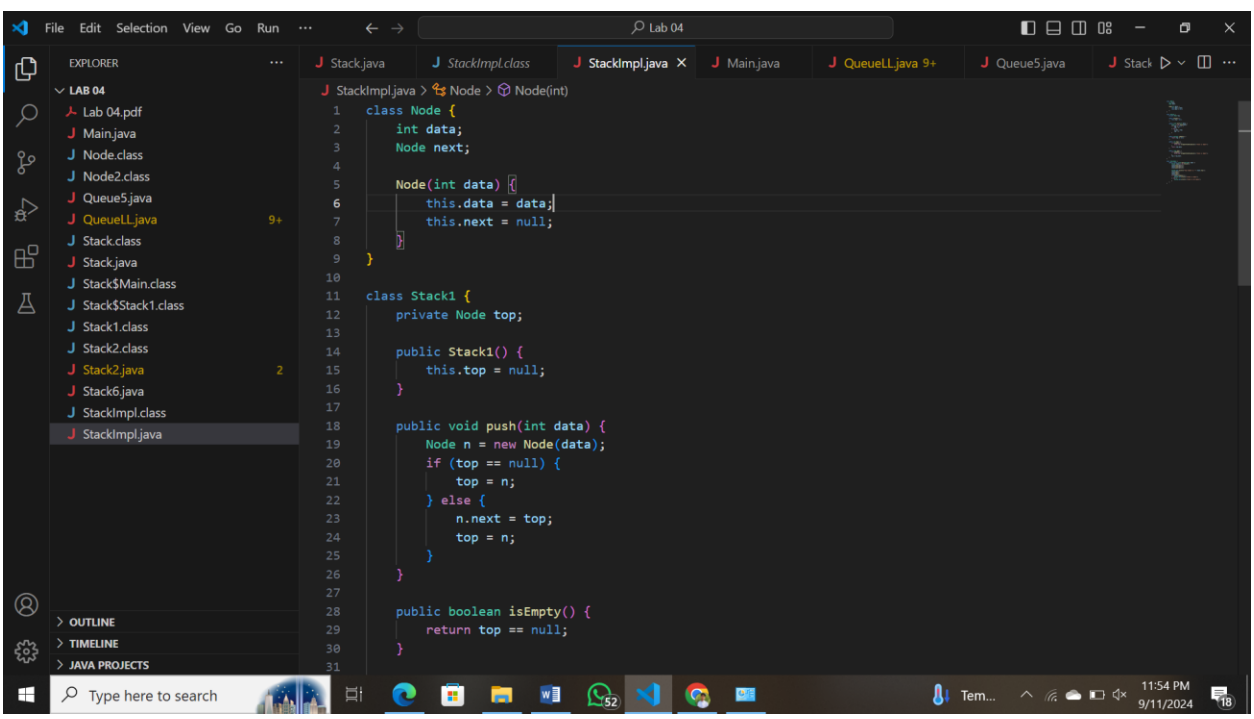




OUTPUT 1:



2. Stack using Linked list: Understand provided code and implement all required methods in Stack. Stack Code is given below:



```java
class Stack1 {
    public boolean isEmpty() {
        return top == null;
    }

    public int top() {
        if (isEmpty()) {
            throw new IllegalStateException(s:"Stack is empty");
        }
        return top.data;
    }

    public void pop() {
        if (isEmpty()) {
            throw new IllegalStateException(s:"Stack is empty");
        }
        top = top.next;
    }
}

class StackImpl {
    Run | Debug
    public static void main(String[] args) {
        Stack1 stack = new Stack1();
        stack.push(data:1);
        stack.push(data:2);
        stack.push(data:3);

        System.out.println("Top element is " + stack.top());
        stack.pop();
        stack.pop();
```

OUTPUT 2:

```java
class StackImpl {
    public static void main(String[] args) {
        Stack1 stack = new Stack1();
        stack.push(data:1);
        stack.push(data:2);
        stack.push(data:3);

        System.out.println("Top element is " + stack.top());
        stack.pop();
        stack.pop();
        stack.pop();
        if (stack.isEmpty()) {
            System.out.print(s:"Stack is empty");
        } else {
            System.out.print(s:"Stack is not empty");
        }
    }
}
```

```
udy\BSCS Semester 3\DSA\DSA LAB\Lab 04>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview'
'-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\PMLS\AppData\Roaming\Code\User\workspaceStorage\
42902f3085d4f2e8c0e8c59f3be0baef\redhat.java\jdt_ws\Lab 04_67aaca77\bin' 'StackImpl'
Top element is 3
Stack is empty
PS C:\Faheem Study\BSCS Semester 3\DSA\DSA LAB\Lab 04>
```

3: Queue using array: Understand provided code and implement all required methods in Queue. Queue Code is given below:

```java
class Queue {
    private int arr[];
    private int front;
    private int rear;
    private int capacity;
    private int count;

    // Constructor to initialize queue
    Queue(int size) {
        arr = new int[size];
        capacity = size;
        front = 0;
        rear = 0;
        count = 0;
    }

    // Utility function to remove front element from the queue and check for Queue
    // Underflow
    public void dequeue() {
        if (isEmpty()) {
            System.out.println(x:"Queue Underflow");
            return;
        }
        front = (front + 1) % capacity;
        count--;
    }

    // Utility function to add an item to the queue and check for queue overflow
    public void enqueue(int item) {
        if (isFull()) {
            System.out.println(x:"Queue Overflow");
```

```java
class Queue {
    // Utility function to add an item to the queue and check for queue overflow
    public void enqueue(int item) {
        if (isFull()) {
            System.out.println(x:"Queue Overflow");
            return;
        }
        arr[rear] = item;
        rear = (rear + 1) % capacity;
        count++;
    }

    // Utility function to return front element in the queue and check for Queue
    // Underflow
    public int peek() {
        if (isEmpty()) {
            System.out.println(x:"Queue Underflow");
            return -1; // Return an invalid value
        }
        return arr[front];
    }

    // Utility function to return the size of the queue
    public int size() {
        return count;
    }

    // Utility function to check if the queue is empty or not
    public Boolean isEmpty() {
        return (count == 0);
```

```java
class Queue {
    }

    // Utility function to check if the queue is full or not
    public Boolean isFull() {
        return (count == capacity);
    }
}

class Main {
    // main function
    public static void main(String[] args) {
        // create a queue of capacity 5
        Queue q = new Queue(size:5);
        q.enqueue(item:1);
        q.enqueue(item:2);
        q.enqueue(item:3);
        System.out.println("Front element is: " + q.peek());
        q.dequeue();
        System.out.println("Front element is: " + q.peek());
        System.out.println("Queue size is " + q.size());
        q.dequeue();
        q.dequeue();
        if (q.isEmpty())
            System.out.println(x:"Queue Is Empty");
        else
            System.out.println(x:"Queue Is Not Empty");
    }
}
```

OUTPUT 3:

```java
class Main {
    public static void main(String[] args) {
        // create a queue of capacity 5
        Queue q = new Queue(size:5);
        q.enqueue(item:1);
        q.enqueue(item:2);
        q.enqueue(item:3);
        System.out.println("Front element is: " + q.peek());
        q.dequeue();
        System.out.println("Front element is: " + q.peek());
        System.out.println("Queue size is " + q.size());
        q.dequeue();
        q.dequeue();
        if (q.isEmpty())
            System.out.println(x:"Queue Is Empty");
        else
            System.out.println(x:"Queue Is Not Empty");
    }
}
```

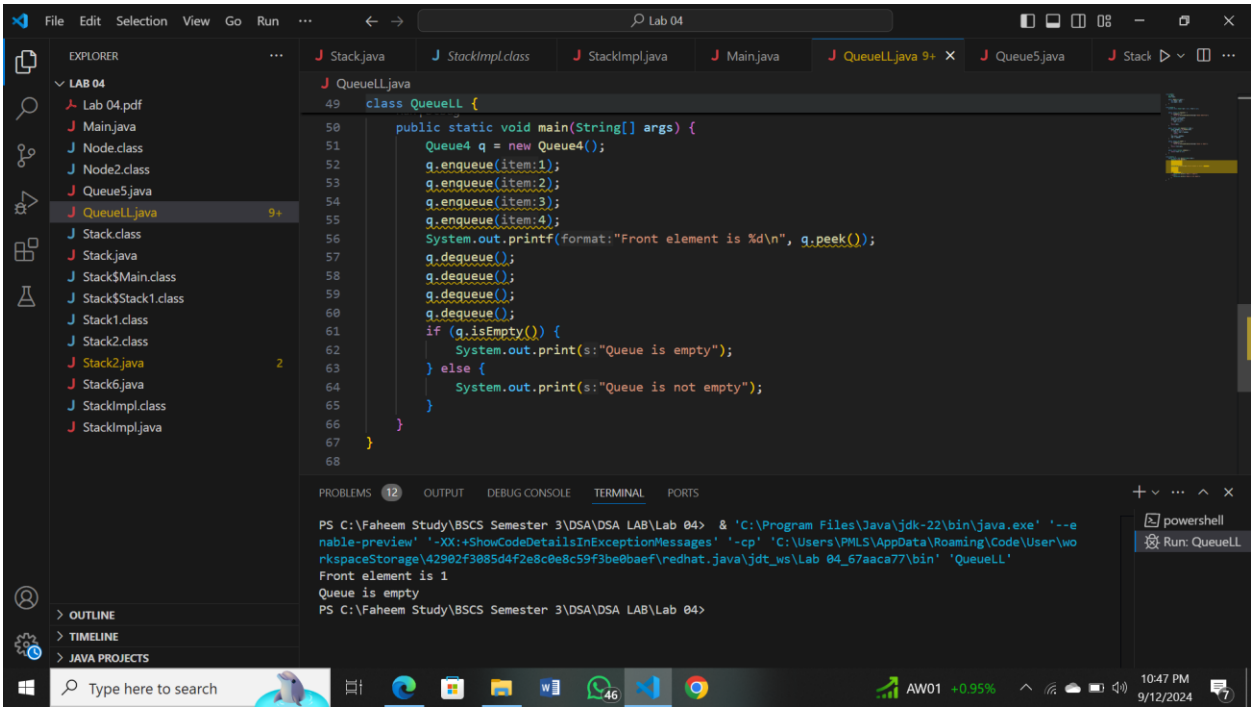
```
udy\BSCS Semester 3\DSA\DSA LAB\Lab 04>  & 'C:\Program Files\Java\jdk-22\bin\java.exe' '--enable-preview'
'-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\PMLS\AppData\Roaming\Code\User\workspaceStorage\
42902f3085d4f2e8c0e8c59f3be0baef\redhat.java\jdt_ws\Lab 04_67aaca77\bin' 'Main'
Front element is: 1
Front element is: 2
Queue size is 2
Queue Is Empty
PS C:\Faheem Study\BSCS Semester 3\DSA\DSA LAB\Lab 04>
```

4. Queue using Linked list: Understand provided code and implement all required methods in Queue. Queue Code is given below:

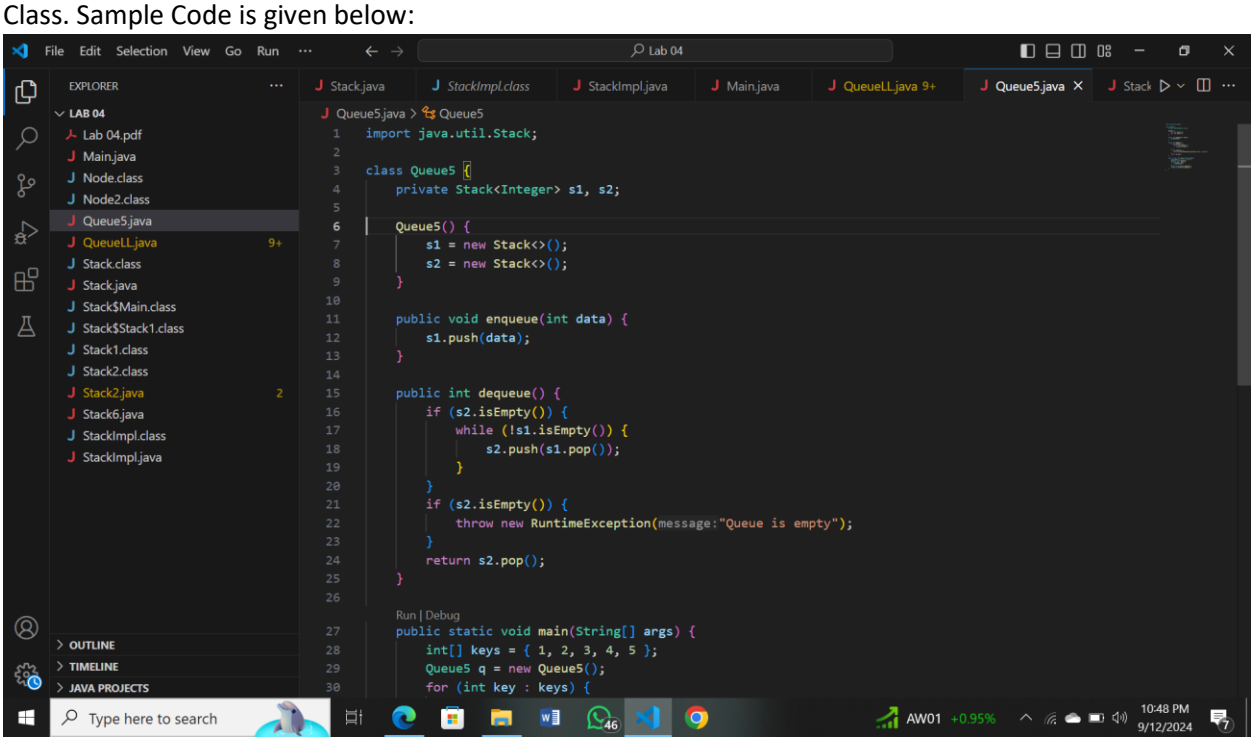
```java
class Node4 {
    int data;
    Node4 next;

    public Node4(int data) {
        this.data = data;
        this.next = null;
    }
}

class Queue4 {
    private static Node4 rear = null, front = null;


    public static int dequeue() {
        if (isEmpty()) {
            throw new RuntimeException(message:"Queue Underflow");
        }
        int data = front.data;
        front = front.next;
        if (front == null) {
            rear = null;
        }
        return data;
    }

    public static void enqueue(int item) {
        Node4 newNode = new Node4(item);
        if (rear == null) {
            front = rear = newNode;
            return;
```

```java
class Queue4 {
    public static void enqueue(int item) {
        if (rear == null) {
        }
        rear.next = newNode;
        rear = newNode;
    }

    public static int peek() {
        if (isEmpty()) {
            throw new RuntimeException(message:"Queue is empty");
        }
        return front.data;
    }

    public static boolean isEmpty() {
        return front == null;
    }
}

class QueueLL {
    Run | Debug
    public static void main(String[] args) {
        Queue4 q = new Queue4();
        q.enqueue(item:1);
        q.enqueue(item:2);
        q.enqueue(item:3);
        q.enqueue(item:4);
        System.out.printf(format:"Front element is %d\n", q.peek());
        q.dequeue();
        q.dequeue();
```

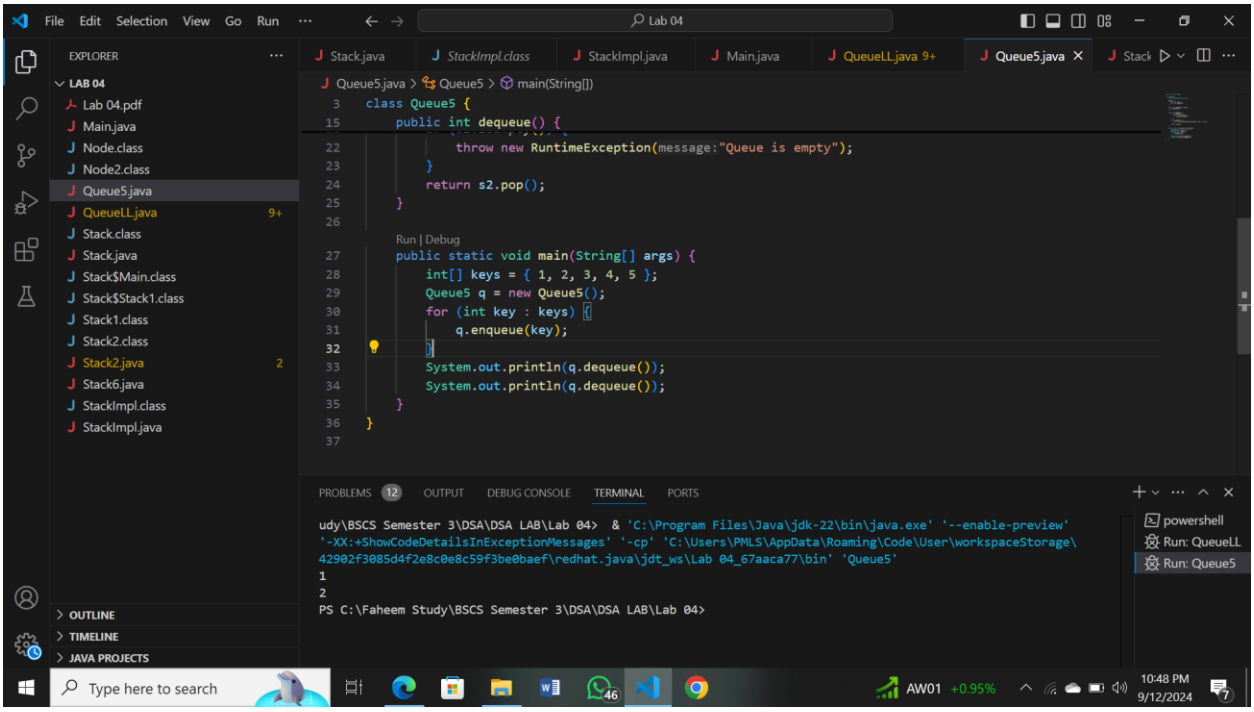
OUTPUT 4:



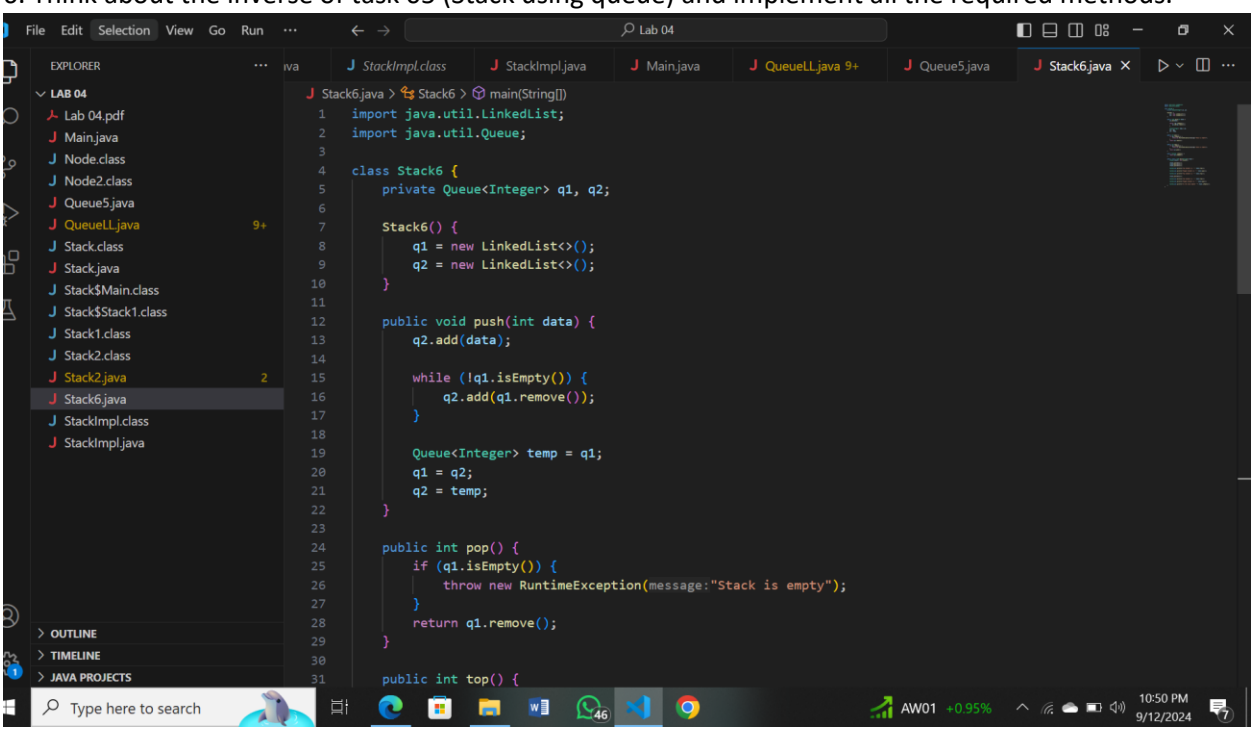
5. Queue using two Stacks: Understand provided code and implement all required methods in Queue Class. Sample Code is given below:



OUTPUT:



6. Think about the inverse of task 05 (Stack using queue) and implement all the required methods.



```java
class Stack6 {

    public int top() {
        if (q1.isEmpty()) {
            throw new RuntimeException(message:"Stack is empty");
        }
        return q1.peek();
    }

    public boolean isEmpty() {
        return q1.isEmpty();
    }

    Run | Debug
    public static void main(String[] args) {
        Stack6 stack = new Stack6();

        stack.push(data:1);
        stack.push(data:2);
        stack.push(data:3);

        System.out.println("Top element is: " + stack.top());

        System.out.println("Popped element is: " + stack.pop());

        System.out.println("Top element is: " + stack.top());

        stack.push(data:4);

        System.out.println("Top element is: " + stack.top());
```

OUTPUT 6:

```java
class Stack6 {
    public static void main(String[] args) {
        stack.push(data:3);

        System.out.println("Top element is: " + stack.top());

        System.out.println("Popped element is: " + stack.pop());

        System.out.println("Top element is: " + stack.top());

        stack.push(data:4);

        System.out.println("Top element is: " + stack.top());

        System.out.println("Popped element is: " + stack.pop());

        System.out.println("Is the stack empty? " + stack.isEmpty());
    }
}
```

```
'-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\PMLS\AppData\Roaming\Code\User\workspaceStorage\
42902f3085d4f2e8c0e8c59f3be0baef\redhat.java\jdt_ws\Lab 04_67aaca77\bin' 'Stack6'
Top element is: 3
Popped element is: 3
Top element is: 2
Top element is: 4
Popped element is: 4
Is the stack empty? false
PS C:\Faheem Study\BSCS Semester 3\DSA\DSA LAB\Lab 04>
```