**Name:** Safyan Anwar

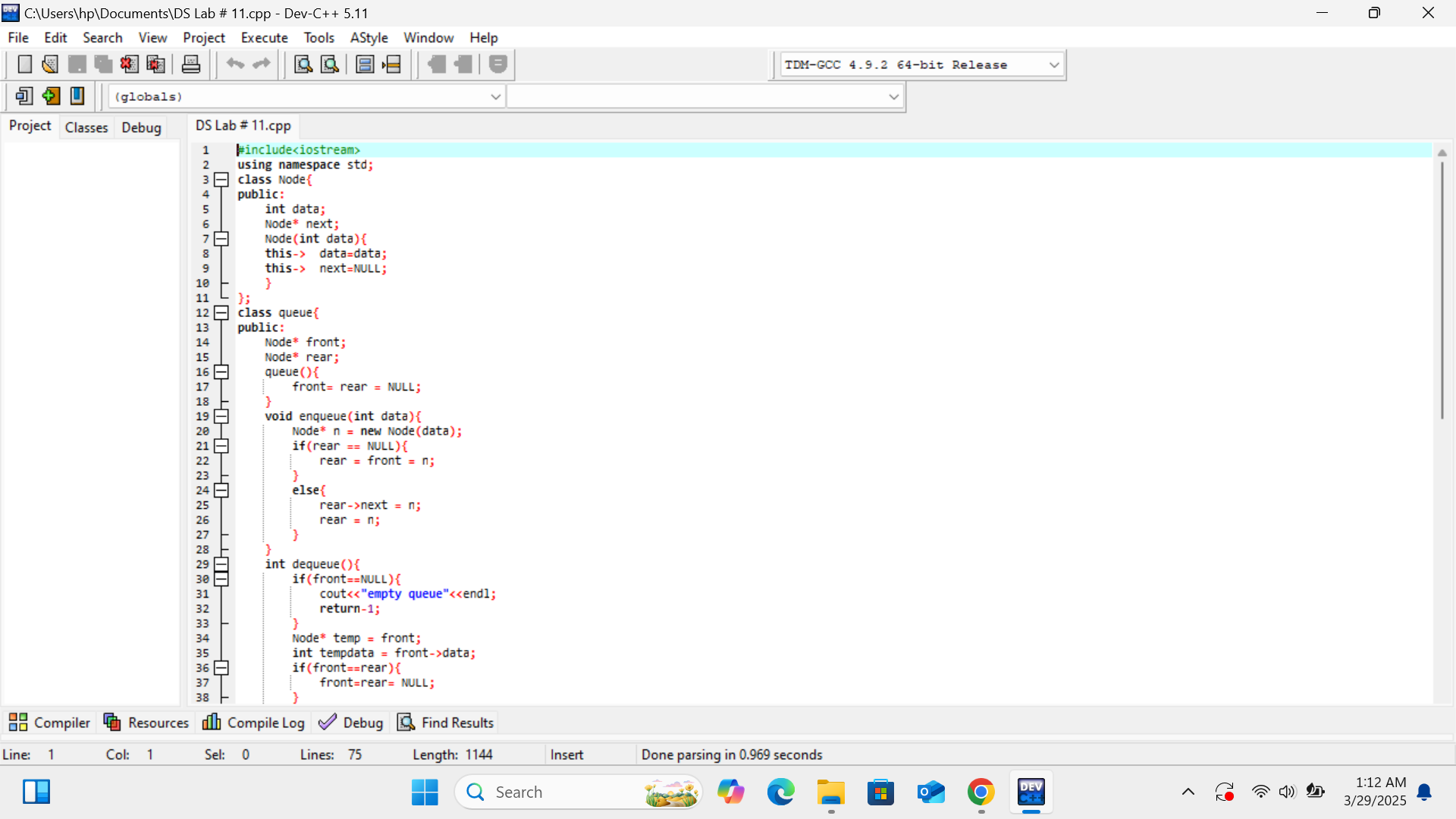
**Roll #:** 018

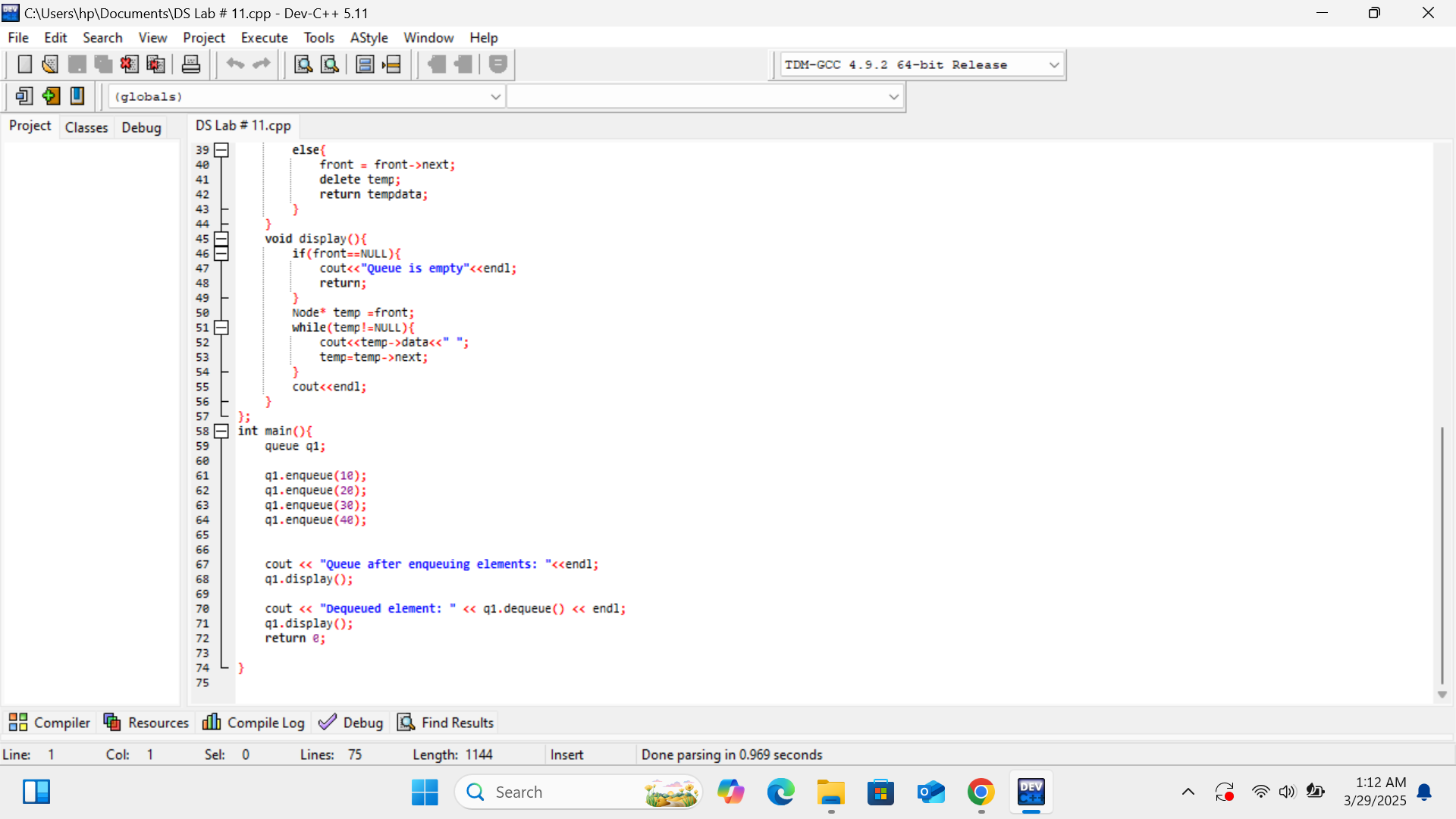
**Section:** 3A-BSSE

**Submitted to:** Sir. Rashik

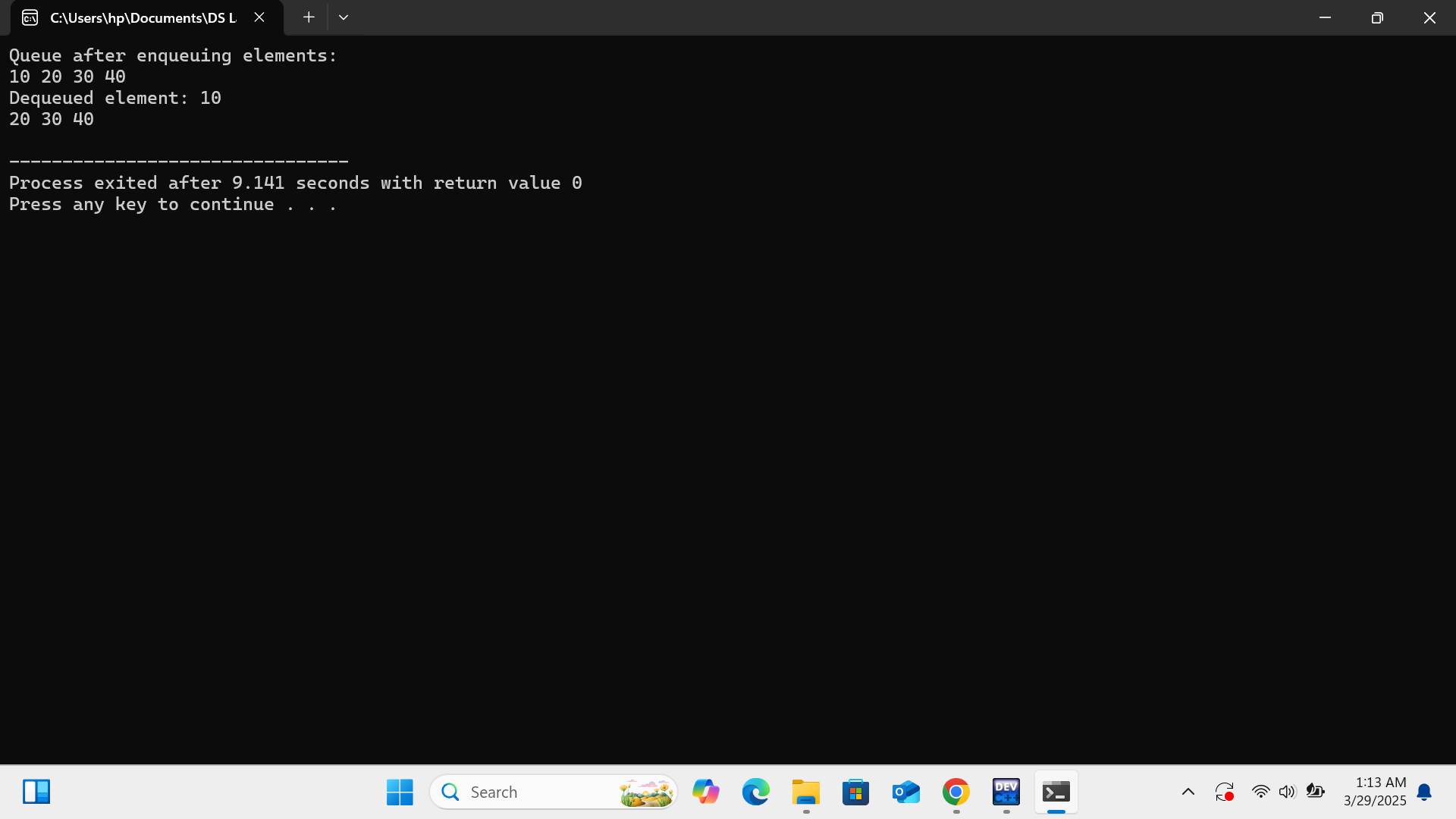
**LAB TASK:**

**Code: ‘**implemented by linked list method:**’:-**

****

****

**Output:**

****

**Explanation:**

1. **Node Class:**
   * Contains two members:
     + **data:** Stores the value of the node.
     + **next:** A pointer to the next node in the queue.
2. **Queue Class:**
   * Contains two pointers:
     + **front**: Points to the first node in the queue.
     + **rear:** Points to the last node in the queue.

### **Methods**

1. **Constructor:**
   * Initializes front and rear to NULL.
2. **enqueue:**
   * Adds a new element to the end of the queue.
   * If the queue is empty both front and rear point to the new node.
   * Otherwise, the new node is added after rear, and rear is updated to point to the new node.
3. **dequeue:**
   * Removes and returns the element at the front of the queue.
   * If the queue is empty, it prints a message and returns -1.
   * If there is only one element, both front and rear are set to NULL.
   * Otherwise, front is updated to point to the next node, and the old front node is deleted.
4. **display:**
   * Prints all elements in the queue from front to rear.
   * If the queue is empty, it prints a message indicating that.

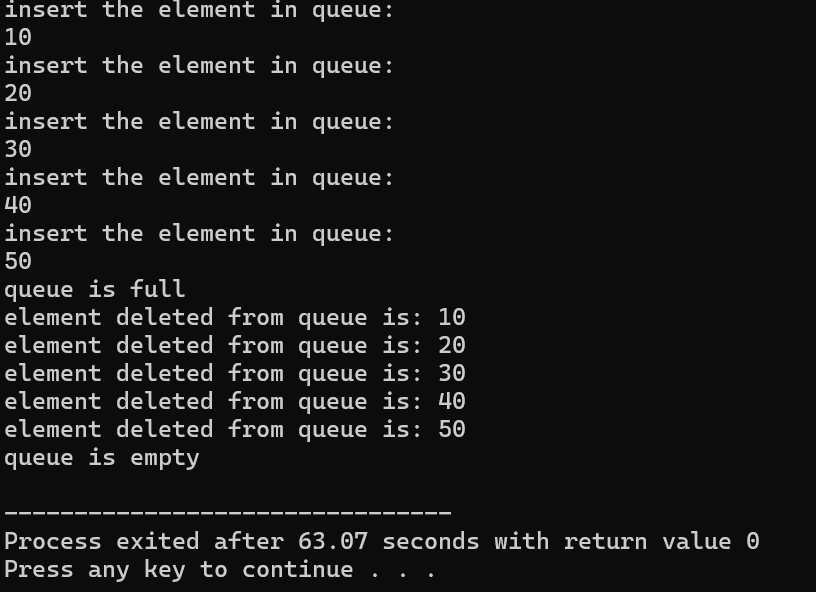
### **Main Function:**

* Creates an instance of the queue class.
* Enqueues four integers like a; (10, 20, 30, 40).
* Displays the queue after enque.

**Code: ‘**implemented code by array**’:-**

****

**Output:**

****

**Explanation:**

1. **Global Variables:**
   * **int queue[5]:** An array to store up to 5 elements.
   * **n=5:** Maximum size of the queue.
   * **front=-1 & rear=-1:** Indicated to track the front and rear of the queue.

### **Functions:**

1. **insert:**
   * Adds an element to the rear of the queue.
   * Checks if the queue is full; if not, it prompts the user for input, increments rear, and stores the value in the queue.
   * If the queue was empty, it sets front to 0.
2. **delete queue:**
   * Removes an element from the front of the queue.
   * Checks if the queue is empty; if not, it prints the removed element.
   * If the queue becomes empty after deletion, it resets front and rear to -1

### **Main Function:**

* Calls insert six times to add elements.
* Calls delete queue six times to remove elements.