

## Lab Assignment: 3.

### 3. Circular Queue:

- **Source Code:**

```
#include <iostream>

#define MAX_SIZE 5

class CircularQueue { private:

    int front, rear;

    int queue[MAX_SIZE];

public:

    CircularQueue() {
front = -1;    rear = -1;

    }

    // Function to check if the queue is empty    bool
isEmpty() {

        return (front == -1 && rear == -1);

    }

    // Function to check if the queue is full    bool
isFull() {

        return (front == (rear + 1) % MAX_SIZE);

    }

    // Function to enqueue (insert) an element    void
enqueue(int data) {    if (isFull()) {

        std::cout << "Queue is full. Cannot enqueue." << std::endl;        return;

    } else if (isEmpty()) {        front =
rear = 0;

    } else {

        rear = (rear + 1) % MAX_SIZE;

    }

}
```

```

        queue[rear] = data;
    }

    // Function to dequeue (remove) an element    void
    dequeue() {    if (isEmpty()) {
        std::cout << "Queue is empty. Cannot dequeue." << std::endl;    return;
    } else if (front == rear) {    front =
rear = -1;
    } else {
        front = (front + 1) % MAX_SIZE;
    }
}

// Function to display the elements in the queue    void
display() {    if (isEmpty()) {
        std::cout << "Queue is empty." << std::endl;    return;
    }
    int i = front;    while (i != rear) {
std::cout << queue[i] << " ";
        i = (i + 1) % MAX_SIZE;
    }
    std::cout << queue[rear] << std::endl;
}
};

int main() {    CircularQueue
queue;

    queue.enqueue(1);    queue.enqueue(2);
queue.enqueue(3);

    queue.display();

    queue.dequeue();    queue.display();

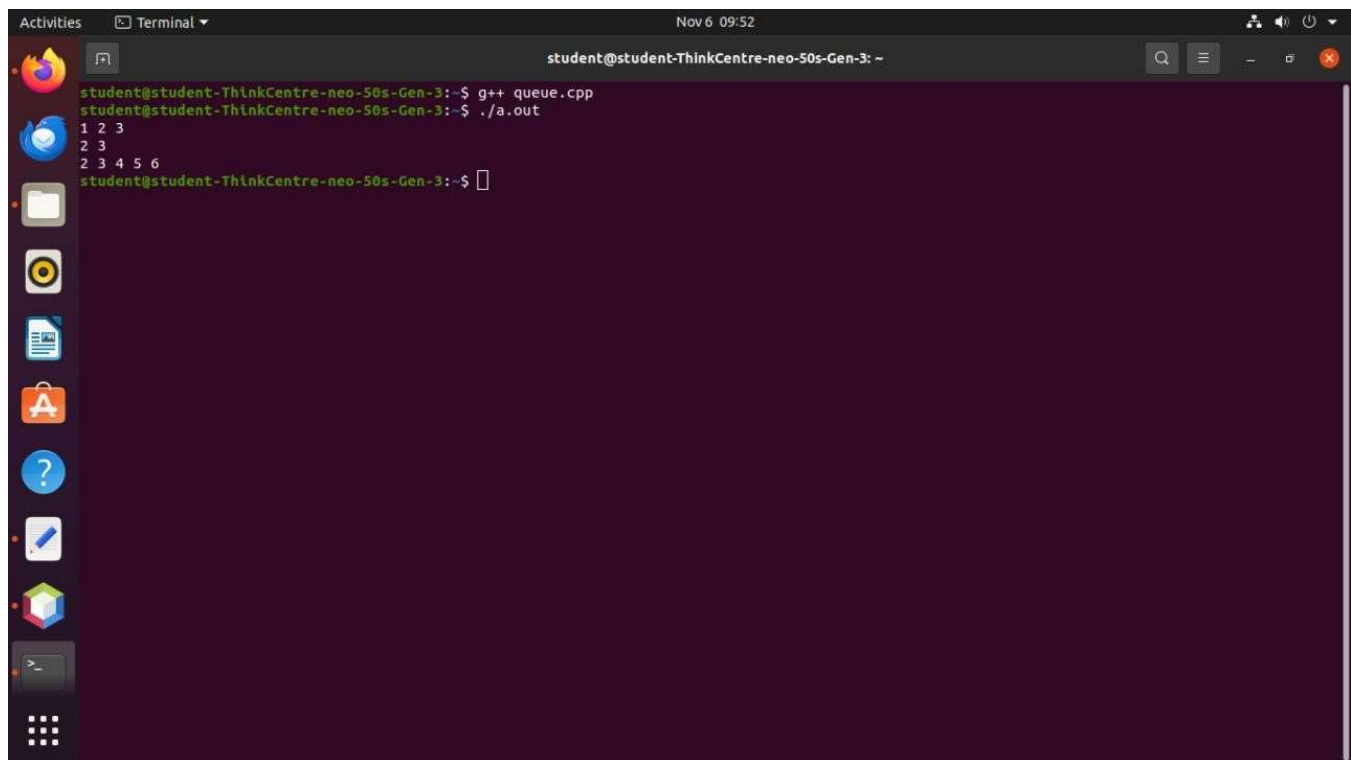
    queue.enqueue(4);    queue.enqueue(5);

    queue.enqueue(6); // Queue is full at this point
queue.display();

    return 0;
}

```

## Output:



A terminal window titled "student@student-ThinkCentre-neo-50s-Gen-3: ~" with a dark purple background. The window shows the execution of a C++ program. The prompt is "student@student-ThinkCentre-neo-50s-Gen-3:~\$". The first command is "g++ queue.cpp", which is followed by a new prompt. The second command is "./a.out", which produces the output "1 2 3" on the next line, "2 3" on the line after, and "2 3 4 5 6" on the line after. The prompt then returns to "student@student-ThinkCentre-neo-50s-Gen-3:~\$". The terminal window has a standard Ubuntu-style top bar with "Activities", "Terminal", and the date/time "Nov 6 09:52". A sidebar on the left contains icons for various applications like Firefox, LibreOffice, and the Dash icon at the bottom.

```
student@student-ThinkCentre-neo-50s-Gen-3:~$ g++ queue.cpp
student@student-ThinkCentre-neo-50s-Gen-3:~$ ./a.out
1 2 3
2 3
2 3 4 5 6
student@student-ThinkCentre-neo-50s-Gen-3:~$
```

