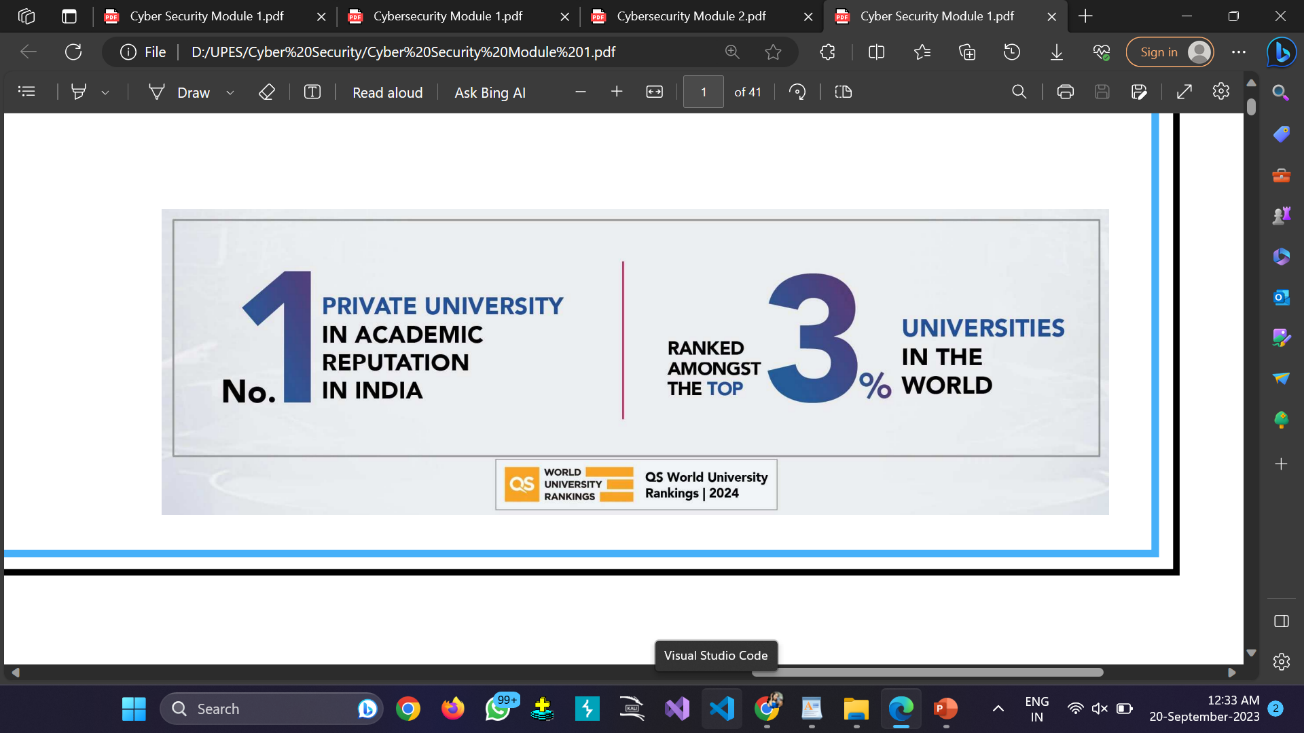
A picture containing text, clipart

Description automatically generated

1

**Lab Experiment: 05**

**Student Detail:**

**• Name:** Prashant Joshi

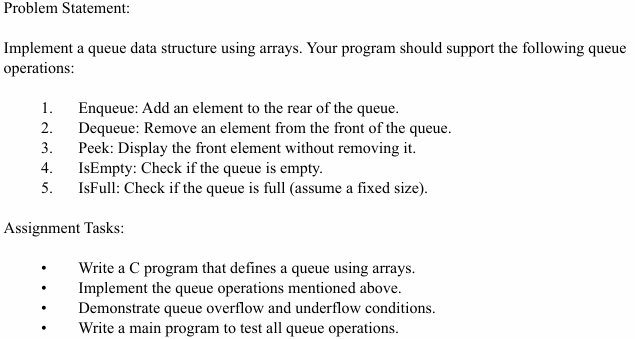
**• Student ID:** 590010879

**• Branch:** MCA

**• Batch:** B1

**• Instructor:** Dr. Sourbh Kumar

Lab Assignment 1: Queue Implementation Using Arrays



Solution:

#include <stdio.h>

#define MAX 5 *// Define the maximum size of the queue*

int queue[MAX];

int front = -1, rear = -1;

*// Function to check if the queue is empty*

int isEmpty() {

    return front == -1;

}

*// Function to check if the queue is full*

int isFull() {

    return rear == MAX - 1;

}

*// Function to add an element to the rear of the queue*

void enqueue(int value) {

    if (isFull()) {

        printf("Queue Overflow! Cannot enqueue %d\n", value);

    } else {

        if (front == -1) front = 0;

        queue[++rear] = value;

        printf("%d enqueued to the queue\n", value);

    }

}

*// Function to remove an element from the front of the queue*

void dequeue() {

    if (isEmpty()) {

        printf("Queue Underflow! Cannot dequeue\n");

    } else {

        printf("%d dequeued from the queue\n", queue[front]);

        if (front == rear) {

            front = rear = -1; *// Reset the queue if it's empty*

        } else {

            front++;

        }

    }

}

*// Function to display the front element of the queue*

void peek() {

    if (isEmpty()) {

        printf("Queue is empty\n");

    } else {

        printf("Front element is %d\n", queue[front]);

    }

}

*// Main function to test the queue operations*

int main() {

    int choice, value;

    do {

        printf("\nQueue Operations:\n");

        printf("1. Enqueue\n");

        printf("2. Dequeue\n");

        printf("3. Peek\n");

        printf("4. Check if Empty\n");

        printf("5. Check if Full\n");

        printf("6. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                printf("Enter value to enqueue: ");

                scanf("%d", &value);

                enqueue(value);

                break;

            case 2:

                dequeue();

                break;

            case 3:

                peek();

                break;

            case 4:

                if (isEmpty()) {

                    printf("Queue is empty\n");

                } else {

                    printf("Queue is not empty\n");

                }

                break;

            case 5:

                if (isFull()) {

                    printf("Queue is full\n");

                } else {

                    printf("Queue is not full\n");

                }

                break;

            case 6:

                printf("Exiting program\n");

                break;

            default:

                printf("Invalid choice! Please try again.\n");

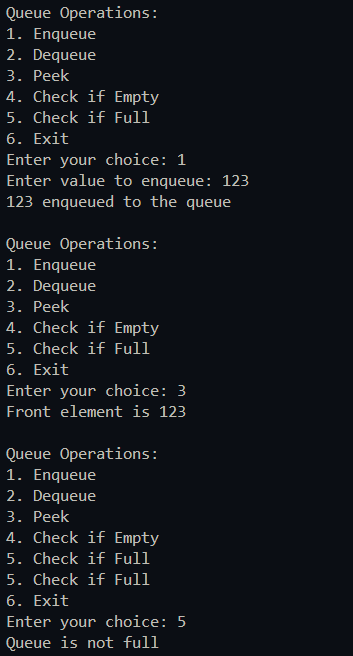
        }

    } while (choice != 6);

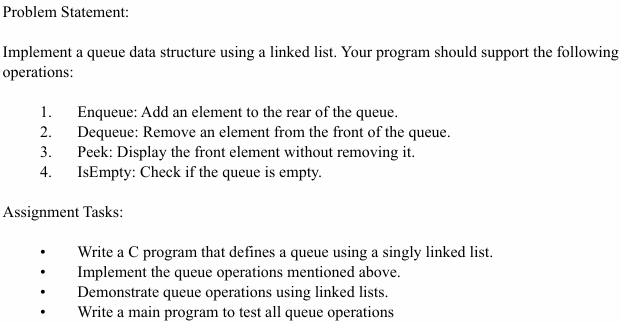
    return 0;

}

Output:



Lab Assignment 2: Queue Implementation Using Linked Lists



#include <stdio.h>

#include <stdlib.h>

// Define a node structure for the queue

struct Node {

int data;

struct Node\* next;

};

// Front and rear of the queue

struct Node\* front = NULL;

struct Node\* rear = NULL;

// Function to check if the queue is empty

int isEmpty() {

return front == NULL;

}

// Function to add an element to the rear of the queue

void enqueue(int value) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

if (!newNode) {

printf("Heap Overflow! Cannot enqueue %d\n", value);

return;

}

newNode->data = value;

newNode->next = NULL;

if (rear == NULL) {

front = rear = newNode;

} else {

rear->next = newNode;

rear = newNode;

}

printf("%d enqueued to the queue\n", value);

}

// Function to remove an element from the front of the queue

void dequeue() {

if (isEmpty()) {

printf("Queue Underflow! Cannot dequeue\n");

return;

}

struct Node\* temp = front;

printf("%d dequeued from the queue\n", front->data);

front = front->next;

if (front == NULL) {

rear = NULL;

}

free(temp);

}

// Function to display the front element of the queue

void peek() {

if (isEmpty()) {

printf("Queue is empty\n");

} else {

printf("Front element is %d\n", front->data);

}

}

// Main function to test the queue operations

int main() {

int choice, value;

do {

printf("\nQueue Operations:\n");

printf("1. Enqueue\n");

printf("2. Dequeue\n");

printf("3. Peek\n");

printf("4. Check if Empty\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("Enter value to enqueue: ");

scanf("%d", &value);

enqueue(value);

break;

case 2:

dequeue();

break;

case 3:

peek();

break;

case 4:

if (isEmpty()) {

printf("Queue is empty\n");

} else {

printf("Queue is not empty\n");

}

break;

case 5:

printf("Exiting program\n");

break;

default:

printf("Invalid choice! Please try again.\n");

}

} while (choice != 5);

return 0;

}

Output:

