

Student Information

- Name:- Aditya Kumar
- Sap Id :- 590015145
- Branch :- M.C.A
- Batch :- B1
- Instructor:- Dr. Sourbh Kumar

Lab Assignment 1: Queue Implementation Using Arrays

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 100
int queue[MAX_SIZE];
int front = -1, rear = -1;

int is_empty() {
    return front == -1;
}

int is_full() {
    return (rear + 1) % MAX_SIZE == front;
}

void enqueue(int item)
{
    if (is_full())
    {
        printf("Queue Overflow\n");
        return;
    }
    if (is_empty())
    {
        front = rear = 0;
    }
    else
    {
        rear = (rear + 1) % MAX_SIZE;
    }
    queue[rear] = item;
    printf("%d enqueued to queue\n", item);
}

int dequeue()
{
    if (is_empty())
    {
        printf("Queue Underflow\n");
        return -1;
    }
}
```

```

    int item = queue[front];
    if (front == rear)
    {
        front = rear = -1;
    }
    else
    {
        front = (front + 1) % MAX_SIZE;
    }
    return item;
}
int peek()
{
    if (is_empty())
    {
        printf("Queue is Empty\n");
        return -1;
    }
    return queue[front];
}

int main()
{
    enqueue(10);
    enqueue(20);
    enqueue(30);
    printf("Front element is %d\n", peek());

    printf("%d dequeued from queue\n", dequeue());
    printf("%d dequeued from queue\n", dequeue());

    enqueue(40);
    printf("Front element is %d\n", peek());

    // Trying to enqueue more elements than the queue capacity
    enqueue(50);
    enqueue(60);

    return 0;
}

```

```
array.c
array.exe
desktop.ini U
experiment1... U
experiment1... U
Experiment4.c U
Experiment4.... U
experiment5... U
experiment5... U
linked_list.c
linked_list.exe
linked_list.png
output.png

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SEARCH ERRORS

PS C:\Users\adi6r\Desktop\> cd "c:\Users\adi6r\Desktop\\"
5_lab1 }
10 enqueued to queue
20 enqueued to queue
30 enqueued to queue
Front element is 10
10 dequeued from queue
20 dequeued from queue
40 enqueued to queue
Front element is 30
50 enqueued to queue
60 enqueued to queue
PS C:\Users\adi6r\Desktop\>
```

Lab Assignment 2: Queue Implementation Using Linked Lists

```
#include <stdio.h>
#include <stdlib.h>
```

```
// Node structure
struct Node
{
    int data;
    struct Node* next;
};
```

```
// Queue structure
struct Queue
{
    struct Node* front;
    struct Node* rear;
};
```

```
// Function to create a new node
struct Node* newNode(int data)
{
    struct Node* node = (struct Node*)malloc(sizeof(struct Node));
    node->data = data;
    node->next = NULL;
    return node;
}
```

// Function to check if the queue is empty

int isEmpty(struct Queue* queue)

```
{
    return queue->front == NULL;
}
```

// Function to enqueue an item to the queue

void enqueue(struct Queue* queue, int data)

```
{
    struct Node* node = newNode(data);
    if (isEmpty(queue))
    {
        queue->front = queue->rear = node;
    }
    Else
    {
        queue->rear->next = node;
        queue->rear = node;
    }
}
```

// Function to dequeue an item from the queue

int dequeue(struct Queue* queue)

```
{
    if (isEmpty(queue))
    {
        printf("Queue is empty\n");
        return -1;
    }
}
```

```
int data = queue->front->data;
struct Node* temp = queue->front;
queue->front = queue->front->next;
```

```
if (queue->front == NULL)
{
    queue->rear = NULL;
}
```

```
free(temp);
return data;
```

```
}
```

// Function to peek the front element of the queue

int peek(struct Queue* queue)

```
{
    if (isEmpty(queue))
    {
        printf("Queue is empty\n");
        return -1;
    }
}
```

```

    }
    return queue->front->data;
}

int main()
{
    struct Queue* queue = (struct Queue*)malloc(sizeof(struct Queue));
    queue->front = queue->rear = NULL;

    enqueue(queue, 10);
    enqueue(queue, 20);
    enqueue(queue, 30);

    printf("Front element is %d\n", peek(queue));

    printf("%d dequeued from queue\n", dequeue(queue));
    printf("%d dequeued from queue\n", dequeue(queue));

    enqueue(queue, 40);
    printf("Front element is %d\n", peek(queue));

    return 0;
}

```

The screenshot shows a Windows File Explorer window on the left with a list of files and folders. On the right, a PowerShell terminal window displays the output of a C program execution.

File Explorer Files:

- array.exe
- desktop.ini
- experiment1...
- experiment1...
- Experiment4.c
- Experiment4....
- experiment5...
- experiment5...
- experiment5...
- linked_list.c
- linked_list.exe

PowerShell Terminal Output:

```

PS C:\Users\adi6r\Desktop\> cd "c:\Users\adi6r\Desktop\" ; if ($?) { gcc e
5_lab2 }
Front element is 10
10 dequeued from queue
20 dequeued from queue
Front element is 30
PS C:\Users\adi6r\Desktop\>

```