**EXPT 4:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_QUEUE\_SIZE 100

// Structure to represent a message

struct Message {

char data[100];

};

// Structure to represent a queue

struct Queue {

struct Message items[MAX\_QUEUE\_SIZE];

int front;

int rear;

};

// Function to initialize the queue

void initializeQueue(struct Queue \*queue) {

queue->front = -1;

queue->rear = -1;

}

// Function to check if the queue is empty

int isEmpty(struct Queue \*queue) {

return (queue->rear == -1);

}

// Function to check if the queue is full

int isFull(struct Queue \*queue) {

return (queue->rear == MAX\_QUEUE\_SIZE - 1);

}

// Function to enqueue a message

void enqueue(struct Queue \*queue, struct Message message) {

if (isFull(queue)) {

printf("Queue Overflow\n");

return;

} else if (isEmpty(queue)) {

queue->front = 0;

queue->rear = 0;

} else {

queue->rear++;

}

strcpy(queue->items[queue->rear].data, message.data);

}

// Function to dequeue a message

struct Message dequeue(struct Queue \*queue) {

struct Message message;

if (isEmpty(queue)) {

printf("Queue Underflow\n");

strcpy(message.data, "");

return message;

} else if (queue->front == queue->rear) {

queue->front = -1;

queue->rear = -1;

} else {

queue->front++;

}

strcpy(message.data, queue->items[queue->front - 1].data);

return message;

}

// Function to display the contents of the queue

void displayQueue(struct Queue \*queue) {

if (isEmpty(queue)) {

printf("Queue is empty\n");

return;

}

printf("Queue contents:\n");

for (int i = queue->front; i <= queue->rear; i++) {

printf("%s\n", queue->items[i].data);

}

}

int main() {

struct Queue queue;

initializeQueue(&queue);

// Message sender

while (1) {

char message[100];

printf("Message Sender: Enter message (or 'exit' to quit): ");

scanf("%s", message);

if (strcmp(message, "exit") == 0) {

break;

}

struct Message msg;

strcpy(msg.data, message);

enqueue(&queue, msg);

printf("Message sent to queue.\n");

}

// Message receiver

while (1) {

char choice[10];

printf("Message Receiver: (d)equeue or (q)uit: ");

scanf("%s", choice);

if (strcmp(choice, "q") == 0) {

break;

}

if (strcmp(choice, "d") == 0) {

struct Message receivedMessage = dequeue(&queue);

if (strcmp(receivedMessage.data, "") != 0) {

printf("Message received from queue: %s\n", receivedMessage.data);

}

}

}

// Display queue contents

displayQueue(&queue);

return 0;

}