perations

#include <stdio.h>

#include <stdlib.h>

#define MAX\_QUEUE\_SIZE 5

// Function declarations

void addq(char item);

char deleteq();

void queueFull();

char queueEmpty();

void displayStatus();

char queue[MAX\_QUEUE\_SIZE];

int front = 0, rear = 0;

// Menu-driven program

int main() {

char choice;

char item;

while (1) {

printf("\nMenu:\n");

printf("a. Insert an Element\n");

printf("b. Delete an Element\n");

printf("c. Demonstrate Overflow\n");

printf("d. Demonstrate Underflow\n");

printf("e. Display the status\n");

29

printf("f. Exit\n");

printf("Enter your choice: ");

scanf(" %c", &choice);

switch (choice) {

case 'a':

printf("Enter the element to insert: ");

scanf(" %c", &item);

addq(item);

break;

case 'b':

item = deleteq();

if (item != queueEmpty()) {

printf("Deleted element: %c\n", item);

} else {

printf("Queue is empty.\n");

}

break;

case 'c':

printf("Demonstrating Overflow...\n");

for (char ch = 'A'; ch <= 'F'; ch++) {

addq(ch);

}

displayStatus();

break;

case 'd':

printf("Demonstrating Underflow...\n");

for (int i = 0; i < MAX\_QUEUE\_SIZE; i++) {

deleteq();

}

displayStatus();

break;

case 'e':

printf("Displaying the status of Circular QUEUE:\n");

displayStatus();

break;

case 'f':

printf("Exiting the program...\n");

exit(0);

default:

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

}

}

30

return 0;

}

// Function to add an item to the queue

void addq(char item) {

rear = (rear + 1) % MAX\_QUEUE\_SIZE;

if (front == rear) {

queueFull();

//exit(1);

}

queue[rear] = item;

}

// Function to remove front element from the queue

char deleteq() {

if (front == rear) {

return queueEmpty();

}

front = (front + 1) % MAX\_QUEUE\_SIZE;

return queue[front];

}

// Function to handle queue-full situation

void queueFull() {

printf("Queue is full. Overflow occurred.\n");

}

// Function to handle queue-empty situation

char queueEmpty() {

printf("Queue is empty. Underflow occurred.\n");

return '\0'; // Returning null character as the error key

}

// Function to display the status of the queue

void displayStatus() {

printf("Current Queue status: ");

if (front == rear) {

printf("Empty\n");

} else {

int i;

for (i = (front + 1) % MAX\_QUEUE\_SIZE; i != rear; i = (i + 1) %

31

MAX\_QUEUE\_SIZE) {

printf("%c ", queue[i]);

}

printf("%c\n", queue[i]);

}

}