#include <stdio.h>

#include <stdlib.h>

#define MAX 5

struct Queue {

int items[MAX];

int front;

int rear;

};

void initQueue(struct Queue\* q) {

q->front = -1;

q->rear = -1;

}

int isEmpty(struct Queue\* q) {

return (q->front == -1);

}

int isFull(struct Queue\* q) {

return (q->rear == MAX - 1);

}

void enqueue(struct Queue\* q, int patient\_id) {

if (isFull(q)) {

printf("Queue is full! Cannot add patient.\n");

return;

}

if (q->front == -1) {

q->front = 0; // The first patient is added

}

q->rear++;

q->items[q->rear] = patient\_id;

printf("Patient with ID %d added to the queue.\n", patient\_id);

}

int dequeue(struct Queue\* q) {

if (isEmpty(q)) {

printf("Queue is empty! No patients to process.\n");

return -1;

}

int patient\_id = q->items[q->front];

if (q->front == q->rear) {

q->front = q->rear = -1;

} else {

q->front++;

}

return patient\_id;

}

int front(struct Queue\* q) {

if (isEmpty(q)) {

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

return -1;

}

return q->items[q->front];

}

void printQueue(struct Queue\* q) {

if (isEmpty(q)) {

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

return;

}

printf("Patients in the queue: ");

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

printf("%d ", q->items[i]);

}

printf("\n");

}

int main() {

struct Queue queue;

initQueue(&queue);

enqueue(&queue, 101);

enqueue(&queue, 102);

enqueue(&queue, 103);

printQueue(&queue);

int patient\_id = dequeue(&queue);

printf("Processing patient with ID %d.\n", patient\_id);

printQueue(&queue);

enqueue(&queue, 104);

enqueue(&queue, 105);

printQueue(&queue);

while (!isEmpty(&queue)) {

patient\_id = dequeue(&queue);

printf("Processing patient with ID %d.\n", patient\_id);

}

return 0;

}