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

#define MAX_SIZE 100

struct Queue {
		int front, rear, size;
		unsigned capacity;
		int* array;
};

struct Queue* createQueue(unsigned capacity) {
		struct Queue* queue = (struct Queue*)malloc(sizeof(struct Queue));
		queue->capacity = capacity;
		queue->front = queue->size = 0;
		queue->rear = capacity - 1;
		queue->array = (int*)malloc(queue->capacity * sizeof(int));
		return queue;
}

int isFull(struct Queue* queue) {
		return (queue->size == queue->capacity);
}

int isEmpty(struct Queue* queue) {
		return (queue->size == 0);
}

void enqueue(struct Queue* queue, int item) {
		if (isFull(queue)) {
				printf("Queue is full. Cannot enqueue %d\n", item);
				return;
		}
		queue->rear = (queue->rear + 1) % queue->capacity;
		queue->array[queue->rear] = item;
		queue->size++;
		printf("%d enqueued to the queue\n", item);
}

int dequeue(struct Queue* queue) {
		if (isEmpty(queue)) {
				printf("Queue is empty. Cannot dequeue\n");
				return -1;
		}
```

```
        int item = queue->array[queue->front];
        queue->front = (queue->front + 1) % queue->capacity;
        queue->size--;
        return item;
}

int front(struct Queue* queue) {
        if (isEmpty(queue)) {
                printf("Queue is empty\n");
                return -1;
        }
        return queue->array[queue->front];
}

int rear(struct Queue* queue) {
        if (isEmpty(queue)) {
                printf("Queue is empty\n");
                return -1;
        }
        return queue->array[queue->rear];
}

int main() {
        struct Queue* queue = createQueue(MAX_SIZE);

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

        printf("Front element is %d\n", front(queue));
        printf("Rear element is %d\n", rear(queue));

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

        printf("Front element is %d\n", front(queue));
        printf("Rear element is %d\n", rear(queue));

        return 0;
}
```