```
// Name: Rodrigo Ignacio Rojas Garcia
// Course Number: ECE 2230
// Section: 001
// Semester: Spring 2017
// Assignment Number: 3
// Â© Rodrigo Rojas. All Rights Reserved.

#ifndef EVENT_QUEUE_H
#define EVENT_QUEUE_H

#include "structures.h"

// Typedef defines the structure of a event_queue_s which contains the list
typedef struct event_queue_s *event_queue_t;

// Function event_queue_t will create an event_queue which then will be allocated w
ith dynamic memory depending of the size that is passed on the
// function creating an array of event_queue_t pointers
event_queue_t event_queue_init(int size);

// Function event_queue_insert  will insert the new event at the end of the heap an
d then it will check if the event_time of the new event (child)
// entered is less than the event_time of the parent and if true it will switch the
ir positions. This will keep checking the parent and if true will switch
// if not it will stop and exit the while loop
int event_queue_insert(event_queue_t event_queue, event_t event);

// Function event_queue_remove will returned the first item on the event_queue and
will set the last event on the event_queue to the head
// of the event_queue and will determine if is bigger than the childs, if true, it
will switch posiitons until it finds something bigger than
// it.
event_t event_queue_remove(event_queue_t event_queue);

// Function event_Queue_empty will be passed an event_queue which will be checked i
f it has been allocated with memory and then will check if there has been any data
// stored in the first slot of the array, if true, it returns 0, if false it return
s -1
int event_queue_empty(event_queue_t event_queue);

// Function event_queue_full will be passed an event_queue and will first chec if t
he event_queue has been allocated with memory, if true, it wil check if the number
of event equals
// the size if the array, if true it means that the data is full and cannot store m
ore items, and returns a 0, if not there is space avaialble it will return a -1
int event_queue_full(event_queue_t eq);

// Function event_queue_finalize will enter a for loop which will free each event o
n the data and will set them to NULL until it reaches value of NULL in
// the array meaning the end. After this it will free the data of the even_queue an
d frees the event_queue
void event_queue_finalize(event_queue_t event_queue);

#endif
```