```
#ifndef THREADPOOL_H
#define THREADPOOL_H

#include <pthread.h>
#include <semaphore.h>

#include <list>
#include <queue>

using namespace std;

typedef struct _routine {
    void (*fct)(void *);
    void *arg;
} routine;

typedef struct _executor_arg {
    queue<routine> waiting; // Routines en attente d'exécution

    // Verrouille et patiente sur la liste des routines en attente
    sem_t sem_waiting;
    pthread_mutex_t mutex_waiting;
} executor_args;

class ThreadPool
{
private:
    list<pthread_t> _threads;

    executor_args _queue;

    ThreadPool(const ThreadPool& other) {};

public:
    ThreadPool(const int n_threads);
    virtual ~ThreadPool();
    virtual ThreadPool& operator=(const ThreadPool& other);

    void inject(void (*fct)(void *), void *arg);
};

#endif // THREADPOOL_H
```