Implement simple thread pool. An example of a required (but not sufficient) interface:

class SimpleThreadPool {

public:

explicit SimpleThreadPool(std::size\_t threadCount);

~SimpleThreadPool();

SimpleThreadPool(const SimpleThreadPool & ) = delete;

SimpleThreadPool & operator = (const SimpleThreadPool & ) = delete;

template < typename Fnc\_T >

auto Post(Fnc\_T task) -> std::future < decltype(task()) > ;

void WorkOn();

void Destroy();

private:

size\_t m\_threadCount;

std::vector < std::thread > threads;

std::queue < std:: function < void() >> tasks;

std::condition\_variable condition;

std::mutex mut;

bool stop;

};

Program should have main() function with the example of usage SimpleThreadPool.