Семинар #2: Мьютекс. Домашнее задание.

Задача 1. Два потока

Напишите программу, в которой вам нужно будет создать два потока:

- Первый поток будет считывать текстовый файл построчно и записывать каждую строку из файла (за исключением пустых строк) поочерёдно в глобальную переменную типа `std::string`. После того как поток записал строку в глобальную переменную, он должен ждать когда второй поток её считает и только потом записывать следующую строку (ждать поток должен просто проверяя строку на пустоту в цикле).
- Второй поток должен ждать пока в глобальную строку что-то запишется (строка перестанет быть пустой). После этого он должен брать эту строку, превращать все строчные буквы в этой строке в заглавные и печатать строку на экран. После этого он должен делать строку пустой, чтобы первый поток мог в неё что-то записать.

Протестируйте вашу программу на файле `invisible_man.txt`.

Задача 2. Потокобезопасная очередь на основе `std::queue`

Реализуйте потокобезопасную очередь на основе стандартной очереди `std::queue`, используя мьютексы. Ваш класс должен содержать метод, который будет вставлять в конец очереди (`push`) и метод, который будет удалять элемент из начала очереди и возвращать его каким-либо образом `pop`. Метод `pop` каким либо образом сообщать получилось ли удалить элемент или очередь была пуста.
Очередь должна быть шаблонной, параметризуемой типом `T`. Протестируйте очередь.

Задача 3. Потокобезопасная очередь на основе односвязного списка

Реализуйте потокобезопасную очередь на основе односвязного списка, используя мьютексы. Ваш класс должен содержать метод, который будет вставлять в конец очереди (`push`) и метод, который будет удалять элемент из начала очереди и возвращать его каким-либо образом `pop`. Метод `pop` каким либо образом сообщать получилось ли удалить элемент или очередь была пуста. Очередь должна быть реализована таким образом, чтобы можно было одновременно добавлять в очередь и удалять из очереди в разных потоках. То есть нужно использовать два мьютекса: один будет защищать начало очереди, а другой – конец.
Очередь должна быть шаблонной, параметризуемой типом `T`. Протестируйте очередь.

Задача 4. Два потока, используя потокобезопасную очередь

Решите задачу 1, но вместо одной глобальной строки используйте глобальную потокобезопасную очередь строк.