Лабораторная работа №7  
Соколов Арсений

**Задание**

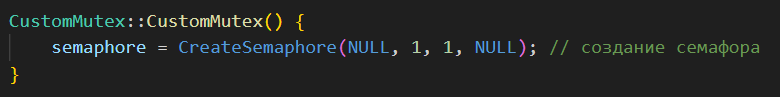
Реализуйте собственный мьютекс либо на основе атомарных переменных и методе compare\_exchange, либо на основе семафора. Он должен представлять собой класс, у которого методы ***lock*** и ***unlock*** **обязательны**.

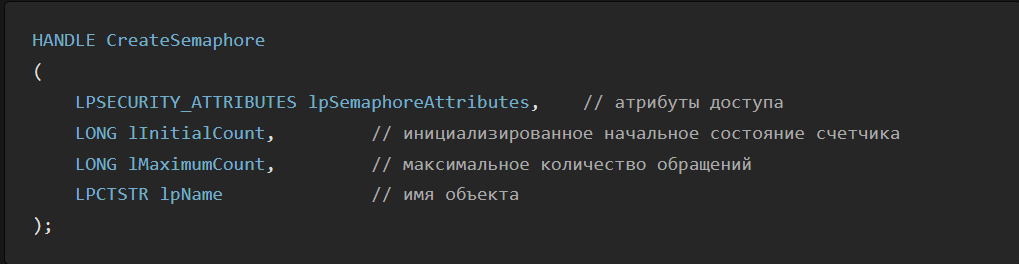
Возьмите своё решение задачи из лабораторной №5 про делёж монет. Подставьте вместо мьютекса из ***stl*** собственную реализацию. Продемонстрируйте работу своего решения. Правильность работы программы должна сохраниться соответственно условию 5й лабораторной.

**Решение**

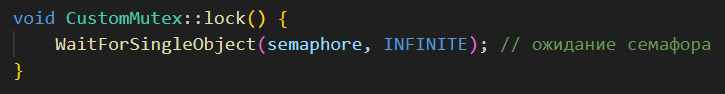
В windows.h реализовано достаточно много штатных инструментов синхронизации (так называемые «объекты синхронизации»). Среди основных: критическая область, событие, мьютекс, семафор.

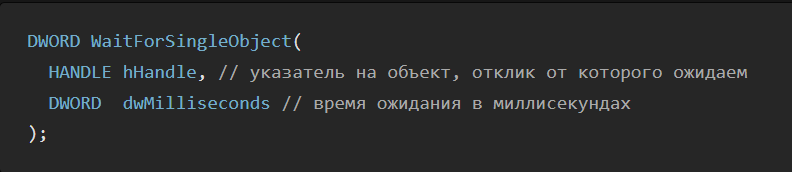
Семафор призван регулировать количество одновременно запущенных потоков. Допустим, у нас 1000 потоков, но одновременно могут работать только 2 (в нашем случае 1).



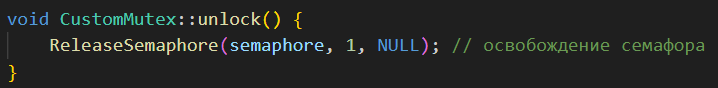


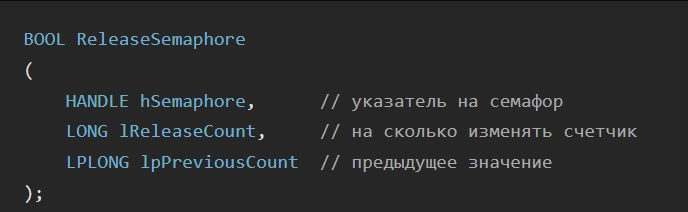
Блокируем текущий поток, он будет ждать неограниченное время (INFINITE), пока семафор не станет доступным.



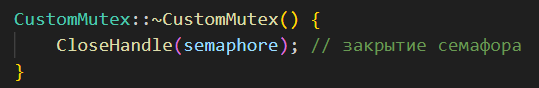


Счетчик семафора у нас меняется (поток выполняется и появляется вакантное место), поэтому состояние семафора нужно периодически менять.



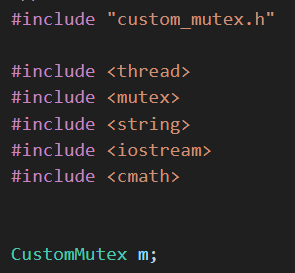


Деструктор

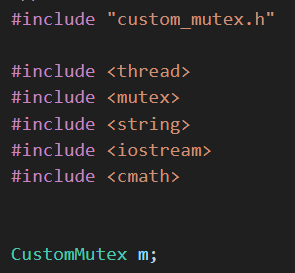


(полный код кастомного мьютекса см в Приложении 1-2)

Добавляем наш модуль в код 5 лабы

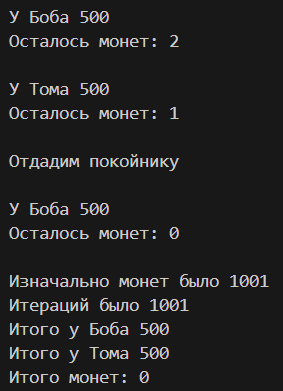


Заменяем системный мьютекс на кастомный



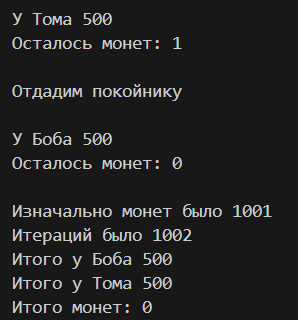
**Результат работы** лабы 5 с кастомным мьютексом на основе семафора:

*Запуск 1* (1001 монета)



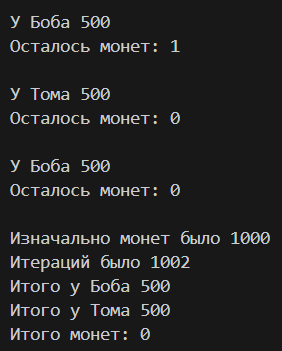
Насколько я понимаю, такое количество операций обусловлено тем, что семафор дает доступ первому потоку, второй «встает в очередь». Когда первый освобождается, начинается второй и первый «встает» за ним. Из-за того, что потоков всего два, а место в семафоре мы сделали всего одно, получается, что они выполняются по очереди, которая еще и совпадает с очередностью вора на взятие монеты.

*Запуск 5*(1001 монета)

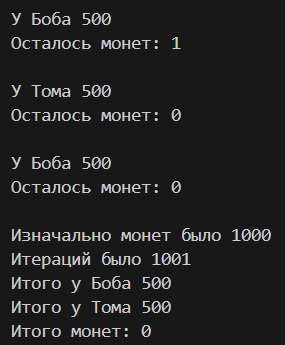


Тут получили на 1 итерацию больше, видимо, из-за того, что Том попытался взять первую монету первым (особенность кода гарантирует, что первую монету всегда берет Боб).

*Запуск 1* (1000 монет)



*Запуск 2* (1000 монет)



**ПРИЛОЖЕНИЕ 1**

***Код из файла custom\_mutex.cpp***

#include "custom\_mutex.h"

CustomMutex::CustomMutex() {

    semaphore = CreateSemaphore(NULL, 1, 1, NULL); // создание семафора

}

void CustomMutex::lock() {

    WaitForSingleObject(semaphore, INFINITE); // ожидание семафора

}

void CustomMutex::unlock() {

    ReleaseSemaphore(semaphore, 1, NULL); // освобождение семафора

}

CustomMutex::~CustomMutex() {

    CloseHandle(semaphore); // закрытие семафора

}

**ПРИЛОЖЕНИЕ 2**

***Код из файла custom\_mutex.h***

#ifndef CUSTOM\_MUTEX\_H

#define CUSTOM\_MUTEX\_H

#include <Windows.h>

class CustomMutex {

public:

    CustomMutex();

    void lock();

    void unlock();

    ~CustomMutex();

private:

    HANDLE semaphore;

};

#endif // CUSTOM\_MUTEX\_H