void LockSemaph() {

//Префикс lock, помещенный перед командой, устанавливает сигнал на линии LOCK системной шины и запрещает доступ к шине другим процессорам на время выполнения данной команды.

//Команда BTS сохраняет значение бита, из первого операнда со смещением, указанным вторым операндом, во флаге CF, а затем устанавливает этот бит в 1.

\_asm {

CriticalSection:

lock bts check, 0;

jc CriticalSection

}

}

void UnlockSemaph() {

//Команда BTR сохраняет значение бита, из первого операнда со смещением, указанным вторым операндом, во флаге CF, а затем обнуляет этот бит.

\_asm lock btr check, 0

}

**1. Дайте определение понятию «синхронизация потоков».**

Синхронизация - механизм упорядочивания выполнения программных блоков двух или более потоков. (Лекция)

Синхронизация потоков (Thread Synchronization) — это обобщенный термин, относящийся к процессу взаимодействия и взаимосвязи потоков. (<https://naparah.com/evm/09131501.html>)

Синхронизация потоков – механизм, позволяющий потокам согласовывать свою работу с общими ресурсами. Этот механизм представляет собой набор объектов операционной системы, которые создаются и управляются программно, являются общими для всех нитей в системе (некоторые - для нитей, принадлежащих одному процессу) и используются для координирования доступа к ресурсам. В качестве ресурсов может выступать все, что может быть общим для двух и более нитей - файл на диске, порт, запись в базе данных, объект GDI, и даже глобальная переменная программы (которая может быть доступна из нитей, принадлежащих одному процессу). (http://www.codenet.ru/progr/cpp/process-threads-sync.php)

**2. Объясните понятие «взаимная блокировка».**

**Взаимная блокировка** – это ситуация в которой, два или более процесса(потока) занимая некоторые ресурсы, пытаются заполучить некоторые другие ресурсы, занятые другими процессами/потоками и ни один из процессов/потоков не может занять необходимый им ресурс, и соответственно освободить занимаемый. (<https://javarush.ru/groups/posts/1422-vzaimnaja-blokirovkadeadlock-v-java-i-metodih-borjhbih-s-ney>)

**Взаимная блокировка** (deadlock) - явление при котором все потоки находятся в режиме ожидания. Происходит, когда достигаются состояния:

· взаимного исключения: по крайней мере один ресурс занят в режиме неделимости и, следовательно, только один поток может использовать ресурс в любой данный момент времени.

· удержания и ожидания: поток удерживает как минимум один ресурс и запрашивает дополнительные ресурсов, которые удерживаются другими потоками.

· отсутствия предочистки: операционная система не переназначивает ресурсы: если они уже заняты, они должны отдаваться удерживающим потокам сразу же.

· цикличного ожидания: поток ждёт освобождения ресурса другим потоком, который в свою очередь ждёт освобождения ресурса заблокированного первым потоком.

3. Перечислите механизмы авторизации(синхранизации) OS.

- Critical section ;

- Mutex;

- Semaphore;

- Atomic operation (interlocking function)

- Event;

- Writable timer.

(лекция)

В книге

Критические области

Взаимное исключение с активным ожиданием

Приостановка и активизация .

Семафоры

Мьютексы

Мониторы

Передача сообщений

Барьеры

**4. Поясните в чем разница между механизмом mutex и semaphore.**

Семафоры — это объекты режима ядра, поэтому они имеют дескрипторы безопасности и описатели. Мьютексы — это тоже объекты режима ядра, используемые для синхронизации, но они проще семафоров, поскольку не имеют счетчиков. (Книга)

В основе семафора лежит счётчик, над которым можно производить две [атомарные операции](https://ru.wikipedia.org/wiki/%D0%90%D1%82%D0%BE%D0%BC%D0%B0%D1%80%D0%BD%D0%B0%D1%8F_%D0%BE%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D1%8F): увеличение и уменьшение значения на единицу, мьютекс можно представить в виде переменной, которая может находиться в двух состояниях: в заблокированном и в незаблокированном. При входе в свою критическую секцию поток вызывает функцию перевода мьютекса в заблокированное состояние. При выходе из критической секции поток вызывает функцию перевода мьютекса в незаблокированное состояние.

5. Почему **mutex,** **semaphore, event** создают объект ядра OS, а **critical section** нет.

Critical section – механизм синхронизации нескольких потоков одного процесса, НЕ ЯВЛЯЕТСЯ объектом ядра OS. (Лекция)