Что необходимо для взаимного исключения

- **Только один процесс может находиться в критической секции для ресурса.** Тут важно понимать, что есть критические секции для разных ресурсов, а нас интересует именно нахождения в критической секции для одного и того же ресурса. Если программа А пишет что-то файл – это не означает, что программа Б не имеет доступа к другому файлу в этот же момент времени, она может писать что-то в другой файл, главное, чтобы это был не один и тот же файл. Это опять же зависит от устройства. Возможно само устройство не позволяет двум процессом одновременно им пользоваться даже если на этом устройстве есть несколько файлов.
- **Процесс, который завершается в некритической секции не должен мешать другим процессам.** Это довольно очевидно. Если процесс завершился, то его завершение не должно блокировать другие процессы, оно не должно влиять на друге процессы.
- **Таких проблем, как взаимная блокировка – дедлок – и ресурсное голодание не должно быть.** Система должна быть спроектирована так, чтобы эти ситуации были невозможными. Или хотя бы из этих ситуаций был нормальный выход, чтобы эти ситуации можно было, во-первых, обнаружить, а во-вторых, разрешить.
- **Процесс не должен ждать доступа к критической секции ресурса если он свободен** (этот ресурс). То есть там не нужно ждать больше, чем нужно.
- **Не должно быть никаких допущений о последовательности и количестве процессов**. Это то, что связано с состоянием гонки, с race condition. Операционной системе должно быть наплевать, в каком порядке выполняются процессы, сколько процессов. Естественно, ей не наплевать на это, это её задача, но результат какой-то операции не должен зависеть от последовательности, то есть операционная система не должна никогда делать допущения. *Если мы выполним этот процесс вот в такой последовательности, то всё будет нормально* – таких допущений быть не должно!
- **Процесс может находиться в критической секции только ограниченное количество времени.** Это очевидно, но не всегда.