# Лабораторная работа №5

# Задание 1

### Необходимые знания

1. Компилирование программ с помощью gcc.
2. Состояние гонки.
3. Критическая секция.
4. POSIX threads: как создавать, как дожидаться завершения.
5. Как линковаться на бибилотеку `pthread`

Скомпилировать mutex.c без использования и с использованием мьютекса. Объяснить разницу в поведении программы.

### Ресурсы

1. Туториал по POSIX threads от университета Карнеги-Меллона
2. Статья о Race condition [wikipedia]
3. Статья о Critical Section [wikipedia]

## Без mutex:

```
@Egor228zorro →/workspaces/os_lab_2019 (master) $ cd ./lab5
@Egor228zorro →/workspaces/os_lab_2019/lab5 (master) $ cd ./src
@Egor228zorro →/workspaces/os_lab_2019/lab5/src (master) $ gcc mutex.c -o mutex_no_mutex -lpthread
@Egor228zorro →/workspaces/os_lab_2019/lab5/src (master) $ ./mutex_no_mutex
doing one thing
counter = 0
doing another thing
counter = 0
doing another thing
counter = 1
doing one thing
counter = 1
doing another thing
counter = 2
doing one thing
counter = 2
```

**doing one thing**

**counter = 0**

**doing another thing**

**counter = 0**

**doing another thing**

**counter = 1**

**doing one thing**

**counter = 1**

**doing another thing**

**counter = 2**

**doing one thing**

**counter = 2**

**doing another thing**

**counter = 3**

**doing one thing**

**counter = 3**

**doing another thing**

**counter = 4**

**doing one thing**

**counter = 4**

**doing another thing**

**counter = 5**

**doing one thing**

**counter = 5**

**doing another thing**

**counter = 6**

**doing one thing**

**counter = 6**

**doing another thing**

**counter = 7**

**doing one thing**

**counter = 7**

**doing another thing**

**counter = 8**

**doing one thing**

**counter = 8**

**doing another thing**

**counter = 9**

**doing one thing**

**counter = 9**

**doing another thing**

**counter = 10**

**doing one thing**

**counter = 10**

**doing another thing**

**counter = 11**

**doing another thing**

**counter = 12**

**doing one thing**

**counter = 11**

**doing another thing**

**counter = 13**

**doing one thing**

**counter = 12**

**doing another thing**

**counter = 14**

**doing one thing**

**counter = 13**

**doing another thing**

**counter = 15**

**doing one thing**

**counter = 14**

**doing another thing**

**counter = 16**

**doing one thing**

**counter = 15**

**doing another thing**

**counter = 17**

**doing one thing**

**counter = 16**

**doing another thing**

**counter = 18**

**doing one thing**

**counter = 17**

**doing another thing**

**counter = 19**

**doing one thing**

**counter = 18**

**doing another thing**

**counter = 20**

**doing one thing**

**counter = 19**

**doing another thing**

**counter = 21**

**doing one thing**

**counter = 22**

**doing another thing**

**counter = 22**

**doing one thing**

**counter = 23**

**doing another thing**

**counter = 23**

**doing one thing**

**counter = 24**

**doing another thing**

**counter = 24**

**doing one thing**

**counter = 25**

**doing another thing**

**counter = 25**

**doing another thing**

**counter = 26**

**doing one thing**

**counter = 26**

**doing another thing**

**counter = 27**

**doing one thing**

**counter = 27**

**doing another thing**

**counter = 28**

**doing one thing**

**counter = 28**

**doing another thing**

**counter = 29**

**doing one thing**

**counter = 29**

**doing another thing**

**counter = 30**

**doing one thing**

**counter = 30**

**doing another thing**

**counter = 31**

**doing one thing**

**counter = 31**

**doing another thing**

**counter = 32**

**doing one thing**

**counter = 32**

**doing another thing**

**counter = 33**

**doing one thing**

**counter = 33**

**doing another thing**

**counter = 34**

**doing one thing**

**counter = 34**

**doing another thing**

**counter = 35**

**doing one thing**

**counter = 35**

**doing another thing**

**counter = 36**

**doing one thing**

**counter = 36**

**doing another thing**

**counter = 37**

**doing one thing**

**counter = 37**

**doing another thing**

**counter = 38**

**doing one thing**

**counter = 38**

**doing another thing**

**counter = 39**

**doing another thing**

**counter = 39**

**doing one thing**

**counter = 39**

**doing another thing**

**counter = 40**

**doing one thing**

**counter = 40**

**doing another thing**

**counter = 41**

**doing one thing**

**counter = 41**

**doing another thing**

**counter = 42**

**doing one thing**

**counter = 42**

**doing another thing**

**counter = 43**

**doing one thing**

**counter = 43**

**doing another thing**

**counter = 44**

**doing one thing**

**counter = 44**

**doing another thing**

**counter = 45**

**doing one thing**

**counter = 45**

**doing another thing**

**counter = 46**

**doing one thing**

**counter = 46**

**doing another thing**

**counter = 47**

**doing one thing**

**counter = 47**

**doing another thing**

**counter = 48**

**doing one thing**

**counter = 48**

**doing one thing**

**counter = 49**

**doing one thing**

**counter = 50**

**doing one thing**

**counter = 51**

**All done, counter = 52**

# C mutex:

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab5/src (master) $ gcc mutex.c -o mutex_yes_mutex -lpthread
@Egor228zorro ➜/workspaces/os_lab_2019/lab5/src (master) $ ./mutex_yes_mutex
doing one thing
counter = 0
doing one thing
counter = 1
doing one thing
counter = 2
doing one thing
counter = 3
doing one thing
counter = 4
doing one thing
```

**doing another thing**

**counter = 0**

**doing another thing**

**counter = 1**

**doing another thing**

**counter = 2**

**doing another thing**

**counter = 3**

**doing another thing**

**counter = 4**

**doing another thing**

**counter = 5**

**doing another thing**

**counter = 6**

**doing another thing**

**counter = 7**

**doing another thing**

**counter = 8**

**doing another thing**

**counter = 9**

**doing another thing**

**counter = 10**

**doing another thing**

**counter = 11**

**doing another thing**

**counter = 12**

**doing another thing**

**counter = 13**

**doing another thing**

**counter = 14**

**doing another thing**

**counter = 15**

**doing another thing**

**counter = 16**

**doing another thing**

**counter = 17**

**doing another thing**

**counter = 18**

**doing another thing**

**counter = 19**

**doing another thing**

**counter = 20**

**doing another thing**

**counter = 21**

**doing another thing**

**counter = 22**

**doing another thing**

**counter = 23**

**doing another thing**

**counter = 24**

**doing another thing**

**counter = 25**

**doing another thing**

**counter = 26**

**doing another thing**

**counter = 27**

**doing another thing**

**counter = 28**

**doing another thing**

**counter = 29**

**doing another thing**

**counter = 30**

**doing another thing**

**counter = 31**

**doing another thing**

**counter = 32**

**doing another thing**

**counter = 33**

**doing another thing**

**counter = 34**

**doing another thing**

**counter = 35**

**doing another thing**

**counter = 36**

**doing another thing**

**counter = 37**

**doing another thing**

**counter = 38**

**doing another thing**

**counter = 39**

**doing another thing**

**counter = 40**

**doing another thing**

**counter = 41**

**doing another thing**

**counter = 42**

**doing another thing**

**counter = 43**

**doing another thing**

**counter = 44**

**doing another thing**

**counter = 45**

**doing another thing**

**counter = 46**

**doing another thing**

**counter = 47**

**doing another thing**

**counter = 48**

**doing another thing**

**counter = 49**

**doing one thing**

**counter = 50**

**doing one thing**

**counter = 51**

**doing one thing**

**counter = 52**

**doing one thing**

**counter = 53**

**doing one thing**

**counter = 54**

**doing one thing**

**counter = 55**

**doing one thing**

**counter = 56**

**doing one thing**

**counter = 57**

**doing one thing**

**counter = 58**

**doing one thing**

**counter = 59**

**doing one thing**

**counter = 60**

**doing one thing**

**counter = 61**

**doing one thing**

**counter = 62**

**doing one thing**

**counter = 63**

**doing one thing**

**counter = 64**

**doing one thing**

**counter = 65**

**doing one thing**

**counter = 66**

**doing one thing**

**counter = 67**

**doing one thing**

**counter = 68**

**doing one thing**

**counter = 69**

**doing one thing**

**counter = 70**

**doing one thing**

**counter = 71**

**doing one thing**

**counter = 72**

**doing one thing**

**counter = 73**

**doing one thing**

**counter = 74**

**doing one thing**

**counter = 75**

**doing one thing**

**counter = 76**

**doing one thing**

**counter = 77**

**doing one thing**

**counter = 78**

**doing one thing**

**counter = 79**

**doing one thing**

**counter = 80**

**doing one thing**

**counter = 81**

**doing one thing**

**counter = 82**

**doing one thing**

**counter = 83**

**doing one thing**

**counter = 84**

**doing one thing**

**counter = 85**

**doing one thing**

**counter = 86**

**doing one thing**

**counter = 87**

**doing one thing**

**counter = 88**

**doing one thing**

**counter = 89**

**doing one thing**

**counter = 90**

**doing one thing**

**counter = 91**

**doing one thing**

**counter = 92**

**doing one thing**

**counter = 93**

**doing one thing**

**counter = 94**

**doing one thing**

**counter = 95**

**doing one thing**

**counter = 96**

**doing one thing**

**counter = 97**

**doing one thing**

**counter = 98**

**doing one thing**

**counter = 99**

**All done, counter = 100**

## Программа без мьютекса:

**Вывод показывает состояние гонки(гонка потоков): два потока одновременно обращаются к переменной common, которая используется как общий счётчик.**

**Потоки читают старое значение переменной, выполняют изменения независимо друг от друга и записывают обратно некорректное значение. Оба потока пытаются одновременно считывать и изменять значение счетчика, что приводит к потере данных и в конечном итоге мы получаем число меньше.**

**doing one thing**

**counter = 11**

**doing another thing**

**counter = 12**

**doing one thing**

**counter = 11**

**Поток do_one_thing прочитал значение common = 11, но до того, как он записал новое значение, поток do_another_thing также прочитал старое значение (common = 11) и увеличил его.**

**В итоге оба потока одновременно работали с одним и тем же значением common, что привело к неправильным обновлениям счётчика.**

## Программа с мьютексом:

**Мьютекс предотвращает состояние гонки(гонка потоков): потоки синхронизированы при доступе к переменной common.**

**Операции внутри критической секции (pthread_mutex_lock и pthread_mutex_unlock) выполняются атомарно. Пока один поток изменяет переменную, другой поток ожидает освобождения мьютекса. как итог:**

**Чтение, модификация и запись переменной выполняются корректно.**

**Итоговое значение переменной common точно равно количеству итераций, выполненных обоими потоками (в данном случае 50 + 50 = 100)**

**doing one thing**

**counter = 11**

**doing another thing**

**counter = 12**

**doing one thing**

**counter = 13**

**Поток do_one_thing завершает обновление переменной common до того, как поток do_another_thing начнёт с ней работать.**

**Значения counter увеличиваются строго последовательно.**

## 1. Компилирование программ с помощью GCC:

**Основная команда:**

**gcc -o output_file source_file.c**

**Для многопоточных программ, использующих POSIX threads, добавляется флаг -pthread:**

**gcc -o program program.c -pthread**

## 2. Состояние гонки :

**Происходит, когда несколько потоков одновременно читают и записывают данные в общий ресурс(и порядок выполнения влияет на результат, что приводит к некорректным результатам.**

**Пример: два потока одновременно увеличивают одно и то же значение переменной.**

## 3. Критическая секция:

**Часть кода, где происходит доступ потоков к общим данным.**

**Чтобы избежать состояния гонки, такой код защищают: например, с помощью мьютексов, чтобы только один поток мог выполнять его в данный момент. В mutex.c критическая секция — это часть, где потоки читают, изменяют и записывают значение common.**

**Без защиты:**

```
work = *pnum_times;  // Чтение
work++;              // Изменение
*pnum_times = work;  // Запись
```

**С защитой мьютексом:**

```
pthread_mutex_lock(&mut);    // Захват мьютекса
work = *pnum_times;          // Чтение
work++;                      // Изменение
*pnum_times = work;          // Запись
pthread_mutex_unlock(&mut);  // Освобождение мьютекса
```

**Использование мьютекса гарантирует, что только один поток выполняет эту секцию кода одновременно.**

## 4.POSIX Threads: как создавать, как дожидаться завершения?

**Создание потока: Поток запускается с помощью системной функции, которая указывает, что он должен делать.**

**Ожидание завершения: Чтобы основной поток дождался завершения других, используют специальную функцию ожидания. Это важно, чтобы программа не завершилась раньше времени.**

## 5.Как линковаться на библиотеку pthread?

**Библиотека pthread реализует POSIX потоки. Чтобы её использовать, при компиляции нужно добавить флаг: -pthread**

**Он сообщает компилятору и компоновщику, что нужно подключить поддержку многопоточности.**

# Задание 2

### Необходимые знания

1. POSIX threads: как создавать, как дожидаться завершения.
2. Как линковаться на бибилотеку `pthread`
3. Как использовать мьютексы.

Написать программу для паралелльного вычисления факториала по модулю `mod` ( `k!` ), которая будет принимать на вход следующие параметры (пример: `-k 10 --pnum=4 --mod=10` ):

1. `k` - число, факториал которого необходимо вычислить.
2. `pnum` - количество потоков.
3. `mod` - модуль факториала

Для синхронизации результатов необходимо использовать мьютексы.

### Ресурсы

1. Туториал по POSIX threads от университета Карнеги-Меллона

```
@Egor228zorro →/workspaces/os_lab_2019/lab5/src (master) $ gcc -pthread -o factorial factorial.c
@Egor228zorro →/workspaces/os_lab_2019/lab5/src (master) $ ./factorial -k 10 --pnum=4 --mod=10
Result: 0
```

# 1.Мьютексы в POSIX Threads

**Мьютекс (взаимное исключение) — это механизм, который позволяет только одному потоку войти в критическую секцию в каждый момент времени. Он используется для предотвращения состояния гонки при доступе к общим данным.**

# 2.Как использовать мьютекс без кода

1. **Перед использованием мьютекс нужно инициализировать.**
2. **Перед входом в критическую секцию поток захватывает мьютекс. Если он уже занят другим потоком, текущий будет ждать.**
3. **После выхода из критической секции поток освобождает мьютекс, чтобы другие потоки могли продолжить.**
4. **После завершения работы с мьютексом его нужно уничтожить, чтобы освободить ресурсы.**

# Задание 3

### Необходимые знания

1. Состояние deadlock

Напишите программу для демонстрации состояния deadlock.

### Ресурсы

1. Статья о deadlock [wikipedia]

**Состояние deadlock (взаимная блокировка) возникает, когда два или более потока блокируют друг друга, ожидая освобождения ресурсов, которые они никогда не смогут получить. Это часто происходит**

**при неправильной последовательности захвата мьютексов или других синхронизирующих объектов.( ситуация, когда два процесса ожидают завершения друг друга)**

```
@Egor228zorro ➜/workspaces/os_lab_2019/lab5/src (master) $ gcc -pthread -o deadlock deadlock.c
@Egor228zorro ➜/workspaces/os_lab_2019/lab5/src (master) $ ./deadlock
Поток 1: Пытаюсь заблокировать Мьютекс 1...
Поток 1: Заблокировал Мьютекс 1.
Поток 2: Пытаюсь заблокировать Мьютекс 2...
Поток 2: Заблокировал Мьютекс 2.
Поток 1: Пытаюсь заблокировать Мьютекс 2...
Поток 2: Пытаюсь заблокировать Мьютекс 1...
```

}

## Как избежать deadlock?

**Единый порядок захвата ресурсов:**

Всегда захватывать мьютексы в одинаковом порядке (например, сначала mutex1, затем mutex2).

Это гарантирует, что deadlock не произойдёт, так как потоки не будут ждать освобождения мьютексов в противоположном порядке.