Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп’ютерних наук

# ЛАБОРАТОРНА РОБОТА № 7

з дисципліни «Операційні системи»

Тема: «Потоки. Синхронізація I»

Виконав:

студент 3 курсу

групи КС-32

Безрук Юрій Русланович

Перевірив: Споров О.Є.

Харків – 2020

Целью данной работы является изучение проблем, которые могут возникнуть в следствии совместной работы нескольких потоков с одним «объектом» и рассмотреть некоторые пути их решения: мьютексы и блокировки чтения-записи.

# ХОД РАБОТЫ

Во всех заданиях по умолчанию отключена буферизация стандартного потока вывода в консоль.

**Задание 1**

Задание 1 наглядно демонстрирует последствия условий гонки потоков. Требуется написать программу, которая создаёт заданное количество пар потоков-потомков, один из которых заданное количество раз увеличивает значение переменной на 1, а другой – уменьшает.

Был выбран такой способ реализации: каждая пара запускается отдельно, после чего главный поток дожидается ее завершения и выводит значение переменной, после чего запускает следующую пару.

Количество потоков и операций можно передавать при помощи опций, благодаря чему можно проследить, при каких значениях условия гонки сильнее оказывают влияние на взаимодействие процессов.

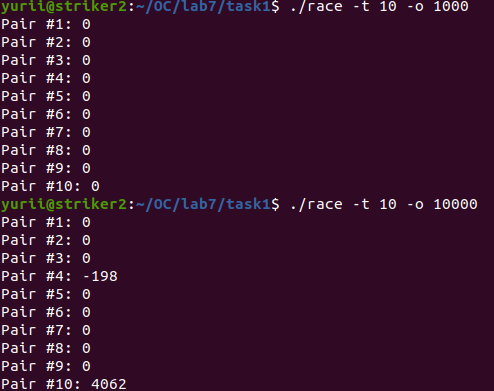


Рисунок 1 - Последствия условий гонок

**Задание 2**

В задании 2 необходимо реализовать синхронизацию потоков при помощи мьютексов.

Требуется посчитать численное значение определённого интеграла на указанном промежутке с указанной точностью, разбив интегральную область между потоками.

При запуске программа из командной строки получает количество потоков, диапазон интегрирования и точность расчётов. Далее инициализируется мьютекс и создается массив структур-аргументов потока. После этого диапазон интергирования разбивается между потоками, каждый из которых высчитывает свою часть, блокирует мьютекс и прибавляет ее к общему результату, после чего освобождает мьютекс.

Для вычисления интеграла каждый поток вызывает функцию integrate(), которая считает его по составному методу средних прямоугольников, разбивая данную область на еще меньшие интервалы. Количество интервалов удваивается, пока не будет достигнута указанная точность.

Вычисление интеграла не зависит от функции, т.к. она передается потоку в качестве аргумента наряду с диапазоном, точностью, адресом мьютекса и общей переменной. Для примера были реализованы три простейшие математические функции. В комментариях рядом с ними написано, какие значения у интеграла должны быть на самом деле, их можно сравнить с теми, которые выводит программа. При установке достаточной точности вычислений они совпадают.

Пример выполнения программы для функции, заданной в условиях задания:

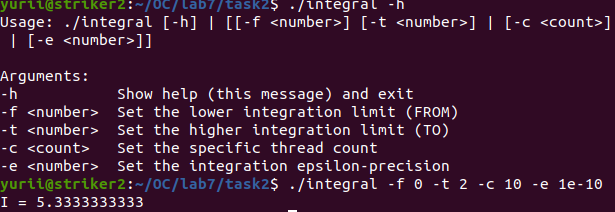


Рисунок 2 - вычисление определённого интеграла на промежутке

**Задание 3**

В задании 3 необходимо рассмотреть другой механизм синхронизации потоков – блокировки чтения/записи.

Блокировки чтения/записи аналогичны мьютексам, за исключением того, что у них есть три режима работы – разблокированный, заблокированный на чтение и заблокированный на запись, что позволяет более практично распределять ресурсы, т.к. несколько потоков могут одновременно читать информацию, но не могут ее записывать в крит. секции.

Итак, в программе существует массив заданного размера, и несколько потоков, которые его обслуживают. Это потоки-писатели, которые генерируют псевдослучайное число в указанном диапазоне и записывают его в случайную ячейку массива (установив блокировку на запись), потоки-читатели, которые считывают информацию из массива, устанавливая блокировку на чтение, что позволяет нескольким читателям считывать одновременно, и один отсоединенный поток-информатор, который выводит состояние массива (для этого он так же устанавливает блокировку на чтение). Основной поток создаёт одного за другим все эти потоки, после чего ожидает из завершения (кроме отсоединённого, который завершается при помощи отмены потока тогда, когда основной поток завершит все остальные).

Для наглядности читателям и писателям была дана случайная задержка перед выполнением их работы, а информатору – постоянная задержка в 1 секунду после каждого информационного сообщения.

Все значения, влияющие на работу программы можно задать из консоли при помощи коротких опций (как и в предыдущих программах, можно вывести справку опцией -h).

Пример работы программы:

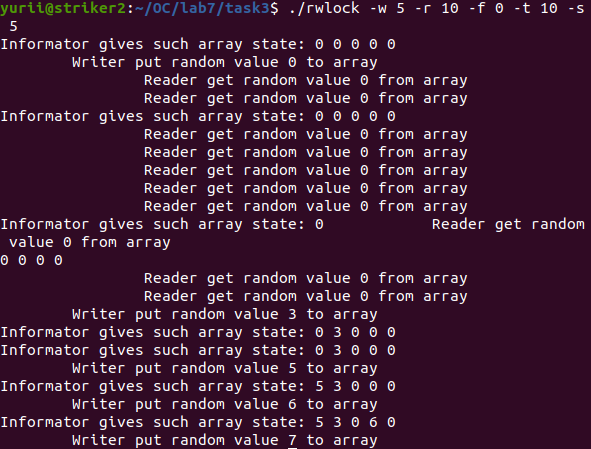


Рисунок 3 - пример работы с блокировками чтения/записи

Как можно видеть, блокировки записи не блокируют друг друга (информатор прерывается читателем), но писателей прервать нельзя.

# ВЫВОДЫ

Таким образом, в ходе выполнения данной работы были рассмотрены возможности синхронизации потоков при помощи мьютексов и блокировок чтений/записи. Соответствующие исходные файлы, исполняемые файлы и скрипты сборки помещены в архив.