

Практическая работа 6

Потоки и синхронизация

1 Дополнительные теоретические сведения

1.1 Демонстрация запуска потоков

Чтобы продемонстрировать тот факт, что процесс запустил нужное количество потоков, следует воспользоваться следующими средствами:

- в Windows нужно скачать средство Process Explorer от Microsoft (<https://docs.microsoft.com/en-us/sysinternals/downloads/process-explorer>), запустить, найти нужный процесс, зайти в Properties → Threads (рисунок 1.1). Также можно воспользоваться любым доступным аналогом на свой выбор;

- в Linux следует воспользоваться командой `top` с ключом `-H`, которую следует запустить из терминала и клавишами `↓ ↑` или `PageDown PageUp` прокрутить список процессов до нужного процесса (для выхода нажмите `q`). Ввиду особенностей организации потоков в Linux они будут отображены как отдельные процессы (рисунок 1.2). Также можно воспользоваться любым доступным аналогом на свой выбор.

Security	Environment	.NET Assemblies	.NET Performance	Strings
Image	Performance	Performance Graph	GPU Graph	Threads
Count: 19				
TID	CPU	Cycles Delta	Suspend Count	Start Address
15608				Adguard.exe+0...
3220				clr.dll!CoUniniti...
16120				clr.dll!_CorExe...
17284				clr.dll!CoUniniti...
16764				clr.dll!CoUniniti...
15660				clr.dll!CoUniniti...
16240				clr.dll!CoUniniti...
15776				clr.dll!CoUniniti...
17288				clr.dll!CoUniniti...
15636				clr.dll!CoUniniti...
3276				clr.dll!CoUniniti...
16236				clr.dll!CoUniniti...
17280				clr.dll!CoUniniti...
15656				clr.dll!SetRunti...
16124				clr.dll!CoUniniti...

Рисунок 1.1 – Демонстрация процессов в Windows

2126	vladimir	20	0	18844	548	468	S	0,0	0,0	0:00.00	main
2127	vladimir	20	0	18844	548	468	S	0,0	0,0	0:00.00	main
2128	vladimir	20	0	18844	548	468	S	0,0	0,0	0:00.00	main

Рисунок 1.2 – Демонстрация потоков в Linux

2 Задание

Для выполнения работы необходимо:

1) Повторно ознакомиться с теоретическим материалом из презентации «Потоки и синхронизация», а также рассмотреть теоретическую часть из данного файла (раздел 1).

Внимание! Все настройки, действия и манипуляции, проводимые для выполнения заданий из пункта 2, необходимо фиксировать в отчёте в виде снимков экрана или листингов вводимых команд.

2) С помощью языка программирования Си или C++ необходимо написать **две** программы с идентичным функционалом, решающих задачу из варианта, – одну для ОС Windows и одну для ОС Linux. Для выполнения задания может использоваться любая среда разработки. При выполнении необходимо учитывать следующие требования:

а) Задача считается решённой, если необходимое по варианту действие выполняется для всех допустимых входных данных.

б) Для работы с потоками (запуска, ожидания, отсоединения, завершения) следует использовать только предназначенные для этого функции из API соответствующих ОС. **Внимание! Не используйте `return` для завершения потоков в данной работе.** Однако для файлового и консольного ввода-вывода можно воспользоваться любыми доступными средствами.

в) Программа должна осуществлять параллельное выполнение расчётов, т.е. в определённый момент времени работы программы в ней должны **одновременно** работать несколько потоков, каждый из которых выполняет свою часть задачи.

г) Доступ ко **всем** (не только к тем, которые заданы искусственно для выполнения работы) разделяемым ресурсам должен быть синхронизирован с помощью соответствующих средств синхронизации, при этом для каждого ресурса должно использоваться

отдельный экземпляр (например, по одному мьютексу на каждый ресурс).

д) Программа должна корректно завершаться, не вызывая аварийный останов.

е) Программа должна брать входные данные из аргументов, переданных при запуске в консоли. В случае, если количество переданных аргументов не равно ожидаемому, программа должна вывести подсказку для пользователя, поясняющую правила её (программы) использования.

ж) Возвращаемые значения **всех** вызываемых функций должны проверяться на предмет возникновения ошибок. В случае возникновения ошибки необходимо вывести сообщение, оповещающее пользователя о произошедшем, содержащее в обязательном порядке **код ошибки и её текстовое описание** (предоставленное ОС). В случае, если в результате возникшей ошибки программа должна быть завершена, перед завершением необходимо освободить все занятые ресурсы (очистить выделенную память, закрыть открытые файлы).

з) Программа должна работать с любым количеством данных без необходимости внесения изменений в код и перекомпиляции. Если количество потоков, которые необходимо породить, больше количества входных данных, необходимо запустить кол-во потоков, вдвое меньшее кол-ва входных данных и вывести соответствующее предупреждение.

3) Оформить отчёт в соответствии с шаблоном из файла «Шаблон и требования к оформлению отчёта по практической работе». Не забыть прикрепить к отчёту **исходные коды** обеих программ **в виде приложения (не приложить отдельными файлами, а именно вставить в сам файл с отчётом под заголовками «Приложение А», «Приложение Б» и т.д.)**. В отчёте необходимо отразить, как минимум:

а) Результаты работы программы в виде снимков экрана, демонстрирующих работу программы для 2-3 вариантов верных и для каждого типа неверных входных данных (чтобы продемонстрировать все предусмотренные сообщения об ошибках).

б) Процесс сборки и запуска программы.

в) Изменения ключевых значений (в переменных) в ходе работы программы.

г) Описание реализованного алгоритма, содержащее листинги соответствующих отрезков кода программы. В описание следует включать только ключевые моменты работы программы (т.е. описывать обработку ошибок и освобождение памяти не требуется).

д) Для обеих программ необходимо продемонстрировать тот факт, что в процессе работы программы было порождено необходимое количество потоков.

4) Защитить работу на занятии. Во время защиты необходимо:

а) Продемонстрировать работу программы.

б) При необходимости изменить входные данные и продемонстрировать, что программа отрабатывает верно на новых данных.

в) При необходимости внести изменения в программный код для демонстрации порождённых потоков.

г) Ответить на вопросы по логике работы программы (почему и зачем была написана та или иная строка, каково её назначение и вклад в решение данной задачи).

д) Ответить на вопросы по теоретической части (**в том числе по лекциям**).

5) Дополнительное задание: дополнительно реализовать третью кроссплатформенную программу, решающую задачу по варианту на языке программирования, отличном от C/C++ и Python, **или** с помощью высокоуровневых кроссплатформенных средств C++ для работы с потоками

(**std::thread**, **std::mutex** и т.д.). Даёт два дополнительных балла. **Не обязательно к выполнению!**

3 Варианты заданий

Общее требование для всех заданий: если количество входных данных (чисел или символов) не делится нацело на количество потоков, числа должны распределяться между потоками следующим образом:

Пусть кол-во входных данных равно M , кол-во потоков равно N , тогда первым $N - 1$ потокам должны быть назначены фрагменты входных данных длины:

$$n = \left\lfloor \frac{M}{N} \right\rfloor,$$

а оставшемуся N -ному потоку:

$$n_N = M - (N - 1) \left\lfloor \frac{M}{N} \right\rfloor,$$

где $\lfloor x \rfloor$ – округление вещественного числа x в меньшую сторону. Например, если дано 10 чисел и их необходимо распределить между 4 потоками, получим:

$$n = \left\lfloor \frac{10}{4} \right\rfloor = \lfloor 2,5 \rfloor = 2,$$

$$n_N = 10 - (4 - 1) \left\lfloor \frac{10}{4} \right\rfloor = 10 - 3 \cdot 2 = 10 - 6 = 4.$$

Таким образом, первым трём потокам достанется по 2 числа, а последнему – 4 числа.

Однако, если $N > \frac{M}{2}$, следует уменьшить количество потоков до $\left\lfloor \frac{M}{2} \right\rfloor$ и вывести соответствующее предупреждение для пользователя.

Ещё одно общее требование для всех заданий: для решения задачи по варианту необходимо искусственно создать разделяемый ресурс (например, глобальную переменную), даже если задачу можно решить и без его использования. Доступ к этому ресурсу из различных потоков необходимо синхронизировать:

- для Linux – с помощью мьютексов;
- для Windows – с помощью объектов **CRITICAL_SECTION**;
- для выполнения доп. задания – любыми доступными средствами синхронизации.

К примеру, в задаче с суммированием чисел можно объявить глобальную переменную, изначально инициализированную значением 0, к которой каждый из потоков будет прибавлять свой результат, а основной поток потом будет просто считывать значение этой переменной в качестве итоговой суммы.

№	Задача
1	В файле записан ряд целых чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму записанных в файл чисел. Для расчёта суммы программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и

	<p>вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
2	<p>В файле записан ряд вещественных чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму записанных в файл чисел. Для расчёта суммы программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
3	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и посчитать, сколько заглавных букв содержится в строке. Для расчёта программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть строки. Каждый из дочерних потоков должен рассчитать количество заглавных букв в своей части строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
4	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и посчитать, сколько строчных (не заглавных) букв содержится в строке. Для расчёта программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть строки.</p>

	<p>Каждый из дочерних потоков должен рассчитать количество строчных (не заглавных) букв в своей части строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
5	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и посчитать, сколько цифр (не чисел, а именно цифр) содержится в строке. Для расчёта программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть строки. Каждый из дочерних потоков должен рассчитать количество цифр в своей части строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
6	<p>В файле записан текст. Программа должна считать имя файла из первого аргумента командной строки и посчитать, сколько управляющих символов (с номерами 0x00-0x1F и 0x7F в таблице ASCII) содержится в тексте. Для расчёта программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть текста. Каждый из дочерних потоков должен рассчитать количество управляющих символов в своей части текста и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>

7	В файле записан ряд целых чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму чётных чисел, записанных в файл. Для расчёта суммы программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму чётных из переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.
8	В файле записан ряд целых чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму нечётных чисел, записанных в файл. Для расчёта суммы программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму нечётных из переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.
9	В файле записан ряд целых чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму квадратов записанных в файл чисел. Для расчёта суммы квадратов программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму квадратов переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить

	значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.
10	В файле записан ряд вещественных чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму квадратов записанных в файл чисел. Для расчёта суммы квадратов программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму квадратов переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.
11	В файле записан ряд положительных целых чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму квадратные корни записанных в файл чисел. Для расчёта суммы квадратные корни программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму квадратные корни переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.
12	В файле записан ряд положительных вещественных чисел, разделённых пробелом. Программа должна считать имя файла из первого аргумента командной строки и рассчитать сумму квадратные корни записанных в файл чисел.

	<p>Для расчёта суммы квадратных корней программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученных чисел. Каждый из дочерних потоков должен рассчитать сумму квадратных корней переданных ему чисел и прибавить её к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 чисел, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
13	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и рассчитать количество вхождений в эту строку одного символа, переданного в качестве третьего аргумента командной строки. Для расчёта количества вхождений программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученной строки. Каждый из дочерних потоков должен рассчитать количество вхождений символа в переданную ему часть строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы.</p>
14	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и рассчитать количество таких символов в этой строке, которые стоят в алфавите раньше символа, переданного в качестве третьего аргумента командной строки. Для расчёта количества символов программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученной строки. Каждый из дочерних потоков должен рассчитать количество соответствующих условию</p>

	<p>символов в переданной ему части строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы. Под позицией в алфавите в данной задаче понимается позиция в таблице ASCII.</p>
15	<p>В файле записана строка. Программа должна считать имя файла из первого аргумента командной строки и рассчитать количество таких символов в этой строке, которые стоят в алфавите после символа, переданного в качестве третьего аргумента командной строки. Для расчёта количества символов программа должна создать N дочерних потоков (N передаётся вторым аргументом командной строки) и передать каждому из них часть полученной строки. Каждый из дочерних потоков должен рассчитать количество соответствующих условию символов в переданной ему части строки и прибавить его к значению разделяемой переменной. Родительский поток должен получить значение разделяемой переменной и вывести его в консоль. Если исходный файл не существует, или в нём записано менее 2 символов, следует вывести соответствующее сообщение для пользователя и завершить работу программы. Под позицией в алфавите в данной задаче понимается позиция в таблице ASCII.</p>