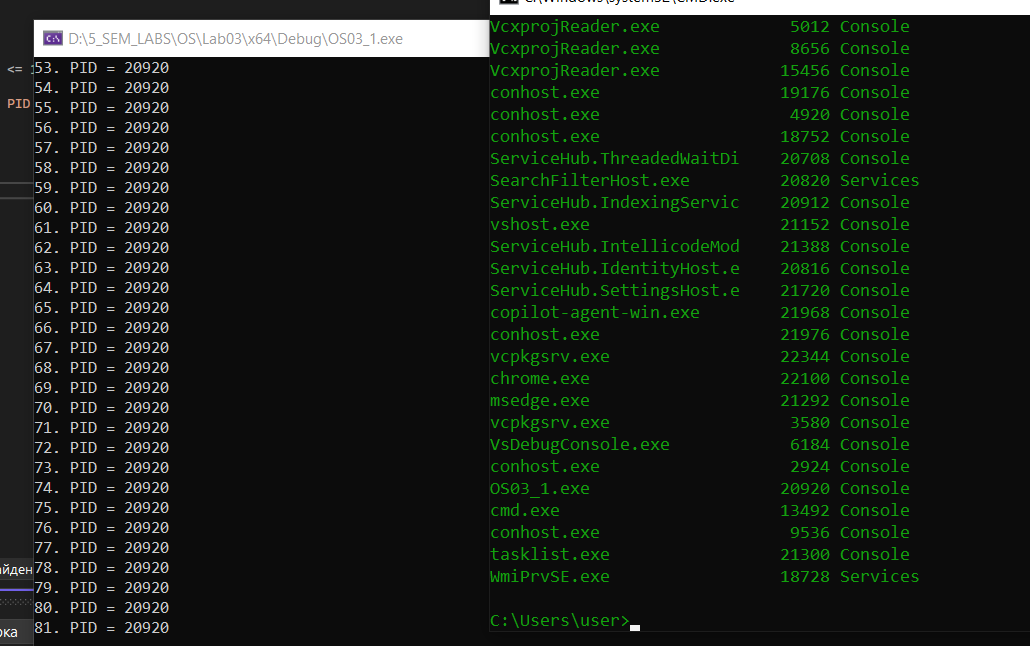
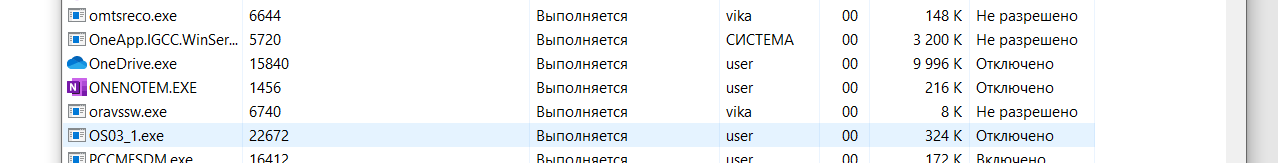
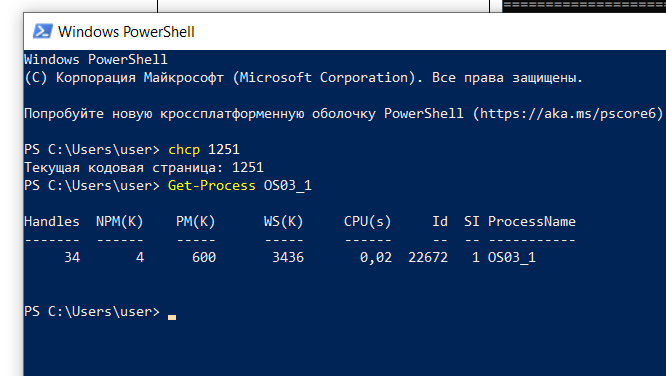
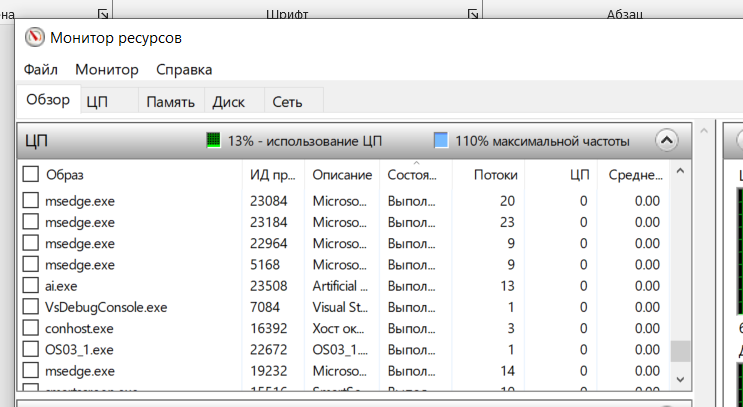
1. Разработайте консольное Windows-приложение OS03\_01 на языке С++, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификатора процесса.

2. Продемонстрируйте информацию о процессе OS03\_01 с помощью утилит Task Manager, tasklist, PowerShell ISE и Performance Monitor.



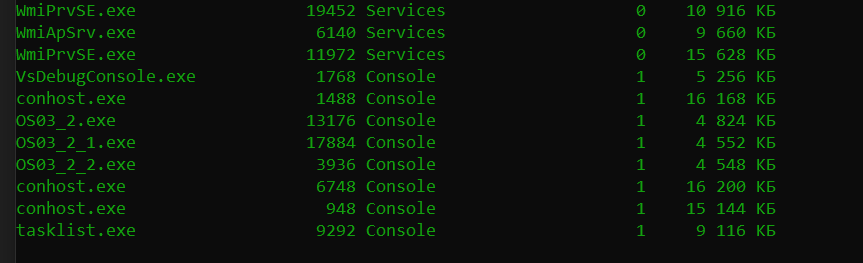


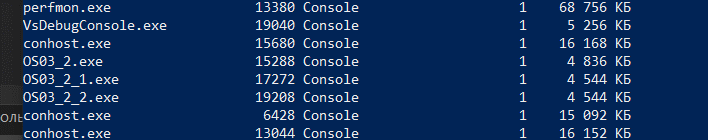


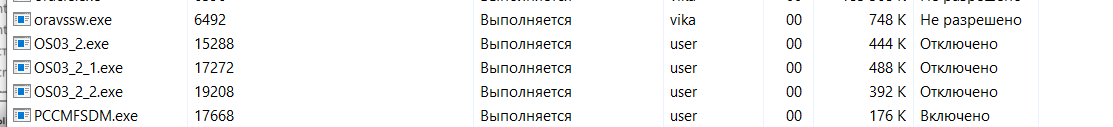


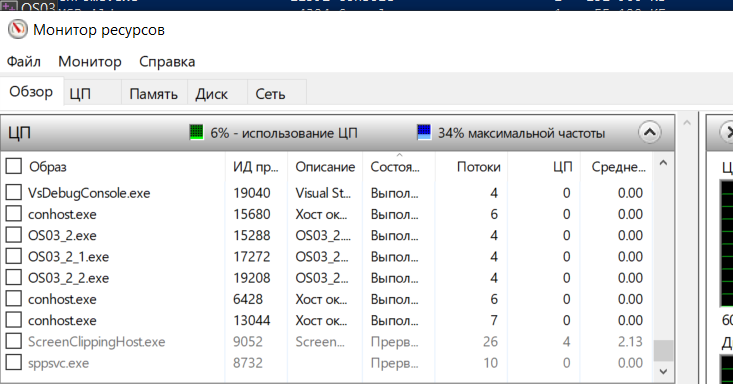
Задание 02

3. Разработайте консольное Windows-приложение OS03\_02 на языке С++, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.









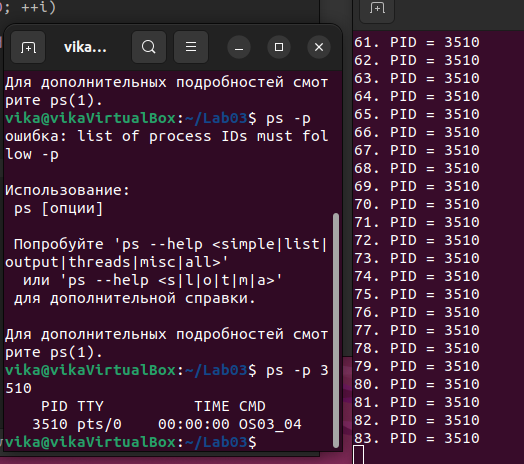
Задание 03.

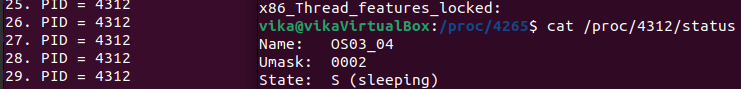
Разработайте консольное Windows-приложение OS03\_03 на языке С++, выводящее на консоль перечень выполняющихся процессов в данный момент в OS.



Задание 04

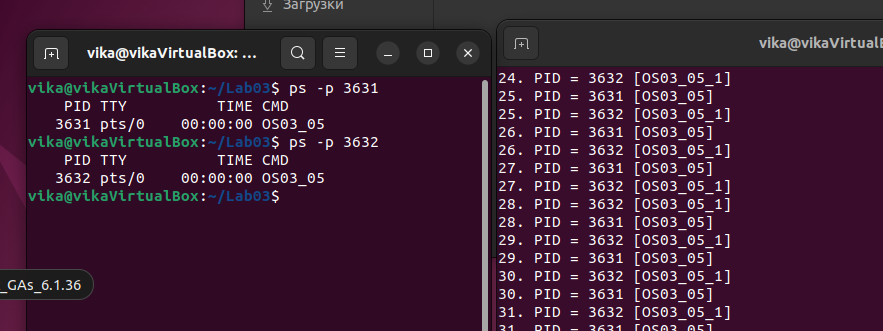
10. Разработайте консольное Linux-приложение OS03\_04 на языке С, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификатора процесса. 11. Продемонстрируйте информацию о процессе OS03\_04 с помощью файловой системы /proc. 12. Продемонстрируйте информацию о процессе OS03\_04 с помощью утилиты ps.

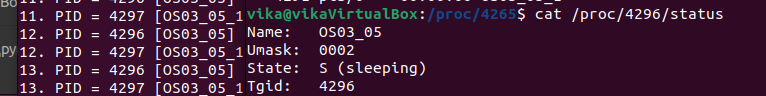


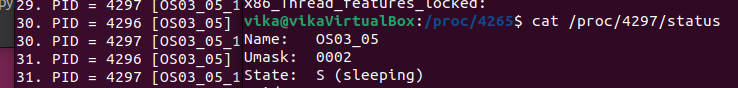


Задание 05

13. Разработайте консольное Linux-приложение OS03\_05 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса. 14. Приложение OS03\_05 должно создавать один дочерний процесс OS03\_05\_1 с помощью системного вызова fork. Процесс OS03\_05\_1 в этом случае не является отдельным модулем, а встроен (fork) в программный модуль OS03\_05. 15. Процесс OS03\_05\_1 - консольное Linux-приложение, выполняющее цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса. 16. Продемонстрируйте информацию о процессах OS03\_05 и OS03\_05\_1 с помощью файловой системы /proc. 17. Продемонстрируйте информацию о процессах OS03\_05 и OS03\_05\_1 с помощью утилиты ps.

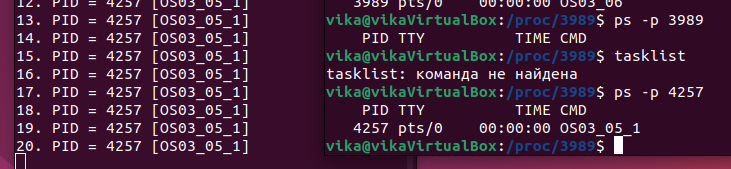


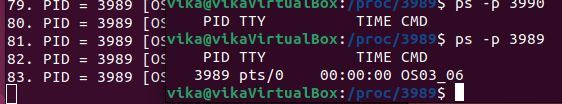


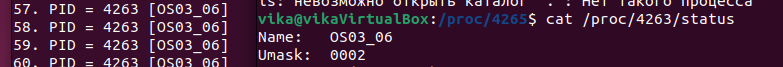


Задание 06

18 Разработайте консольное Linux-приложение OS03\_06 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса. 19. Приложение OS03\_06 должно создавать один дочерний процесс OS03\_05\_1 (отдельный модуль) с помощью системного вызова system. 20. Продемонстрируйте информацию о процессах OS03\_06 и OS03\_05\_1 с помощью файловой системы /proc. 21. Продемонстрируйте информацию о процессах OS03\_06 и OS03\_05-1 с помощью утилиты ps.

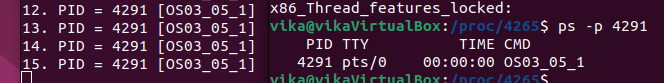




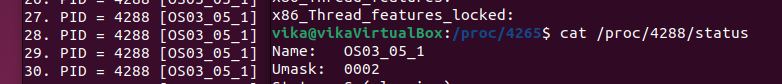


Задание 07

22. Разработайте консольное Linux-приложение OS03\_07 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса 23. Приложение OS03\_07 должно создавать один дочерний процесс OS03\_05\_1 (отдельный модуль) с помощью системного вызова exec. 24. Продемонстрируйте информацию о процессах OS03\_07 и OS03\_05\_1 с помощью файловой системы /proc. 25. Продемонстрируйте информацию о процессах OS03\_07 и OS03\_05-1 с помощью утилиты ps. 26. Продемонстрируйте разницу системных вызовов system и exec.







**27. Что такое процесс?**

Процесс — это экземпляр выполняемой программы. Он включает в себя код, данные и системные ресурсы, необходимые для выполнения программы. Каждый процесс имеет свое собственное адресное пространство и идентификатор (PID), что позволяет операционной системе управлять им независимо от других процессов.

**28. Что такое контекст процесса?**

Контекст процесса — это набор информации, необходимой для управления процессом. Он включает в себя сведения о состоянии процессора, регистры, указатели на стек, адреса памяти, открытые файловые дескрипторы и другую информацию, которая позволяет операционной системе возобновить выполнение процесса после его приостановки. Контекст процесса сохраняется в момент переключения контекста между процессами.

**29. Что такое родительский и дочерний процесс?**

* **Родительский процесс** — это процесс, который создает другой процесс с помощью системного вызова.
* **Дочерний процесс** — это процесс, созданный родительским процессом. Дочерний процесс наследует некоторые ресурсы и характеристики родительского процесса, такие как открытые файловые дескрипторы и контекст.

**30. Что такое процесс инициализации OS?**

Процесс инициализации операционной системы (обычно называется init в Unix/Linux или System в Windows) — это первый процесс, который запускается при загрузке операционной системы. Он отвечает за инициализацию системы, запуск других процессов и управление жизненным циклом системы. В Linux init имеет PID 1, и от него зависят все остальные процессы.

**31. Перечислите области памяти процесса и поясните их назначение.**

* **Код (текст)**: Содержит исполняемый код программы.
* **Данные**: Хранит глобальные и статические переменные.
* **Стек**: Используется для хранения локальных переменных, адресов возврата и управления вызовами функций.
* **Куча**: Область памяти для динамического выделения памяти во время выполнения (например, с помощью malloc в C).

**32. Чем отличаются системные процессы от пользовательских?**

* **Системные процессы**: Запускаются операционной системой для выполнения системных задач, таких как управление ресурсами, планирование процессов, обработка ввода-вывода и другие функции. Они обычно работают с высокими привилегиями.
* **Пользовательские процессы**: Запускаются пользователями для выполнения приложений и программ. Они работают с ограниченными привилегиями и могут взаимодействовать с системными процессами через определенные интерфейсы.

**33. Что такое Windows-сервисы, Linux-демоны?**

* **Windows-сервисы**: Фоновая программа, которая запускается при старте операционной системы и может работать независимо от пользовательских сессий. Сервисы часто используются для выполнения длительных задач или предоставления сетевых функций.
* **Linux-демоны**: Фоновые процессы, которые запускаются при загрузке системы и работают постоянно, ожидая событий или запросов. Демоны часто обрабатывают сетевые соединения, управление заданиями и другие фоновые задачи.

**34. С помощью каких системных вызовов можно создать дочерний процесс в Windows? Поясните разницу.**

В Windows для создания дочернего процесса используются функции:

* **CreateProcess**: Создает новый процесс и его основной поток.
* **ForkProcess**: Не существует в Windows, так как Windows не имеет прямого аналога Linux fork.

CreateProcess предоставляет больше контроля над созданием процесса, так как позволяет задать параметры, такие как безопасность, приоритет и рабочий каталог.

**35. С помощью каких системных вызовов можно создать дочерний процесс в Linux? Поясните разницу.**

В Linux для создания дочернего процесса используются:

* **fork**: Создает копию текущего процесса (дочерний процесс) и возвращает PID дочернего процесса в родительском процессе и 0 в дочернем процессе.
* **exec**: После fork может использоваться для замены дочернего процесса новым исполняемым файлом.

Разница в том, что fork создает новый процесс, а exec заменяет текущий процесс другим программным кодом.

**36. Какие потоки данных доступны любому процессу автоматически?**

Каждому процессу автоматически доступны три стандартных потока данных:

* **Стандартный ввод (stdin)**: Используется для ввода данных.
* **Стандартный вывод (stdout)**: Используется для вывода данных.
* **Стандартный вывод ошибок (stderr)**: Используется для вывода сообщений об ошибках.

**37. Поясните назначение системного вызова WaitForSingleObject в Windows-приложении.**

WaitForSingleObject — это системный вызов, который приостанавливает выполнение текущего потока до тех пор, пока указанный объект (например, поток или процесс) не станет доступным (завершится) или не истечет заданное время ожидания. Это полезно для синхронизации потоков и управления состоянием процессов.

**38. Поясните назначение системного вызова wait в Linux-приложении.**

wait — это системный вызов, который заставляет родительский процесс ждать завершения дочернего процесса. Он возвращает PID завершившегося дочернего процесса и может также предоставлять информацию о статусе завершения. Это важно для управления ресурсами и предотвращения создания "зависших" процессов (зомби).

**39. Дайте развернутое определение процесса OS.**

Процесс в операционной системе — это программа, находящаяся в состоянии выполнения, с выделенными ресурсами и собственным адресным пространством. Он включает в себя код программы, данные, стек и кучу. Процессы управляются операционной системой, которая распределяет ресурсы, контролирует выполнение и обеспечивает взаимодействие между процессами. Каждый процесс имеет уникальный идентификатор (PID) и может создавать дочерние процессы, взаимодействовать с другими процессами и управлять ресурсами системы.