Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Немкович А.В.

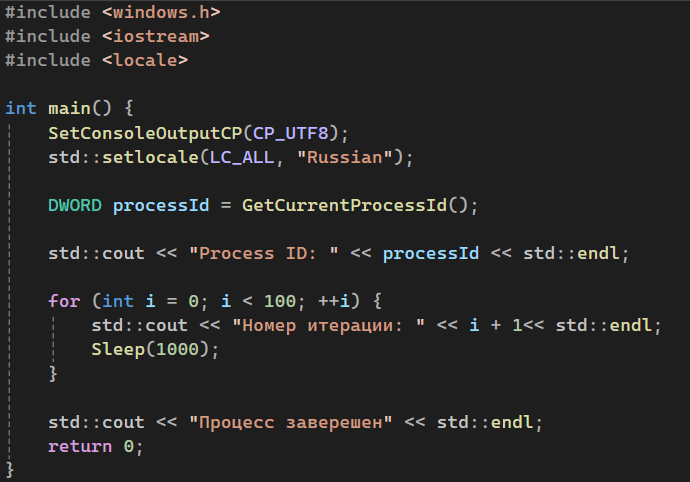
ФИТ 3 курс 1 группа

Преподаватель: Бернацкий П.В

Минск 2024

**Задание 01**

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

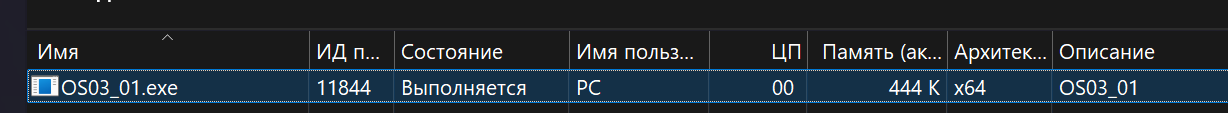


Update: число итераций было увеличено.

Продемонстрируйте информацию о процессе OS03\_01 с помощью утилит

Task Manager:

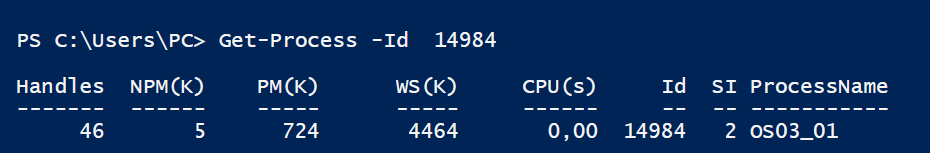




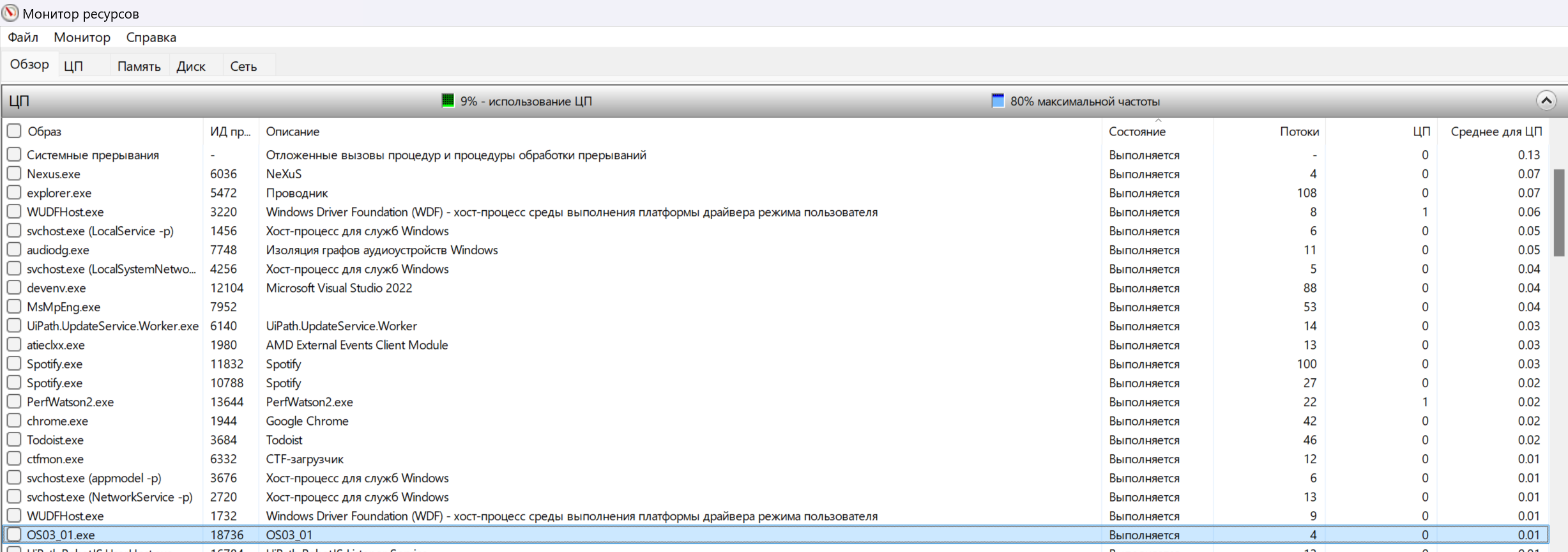
Tasklist:

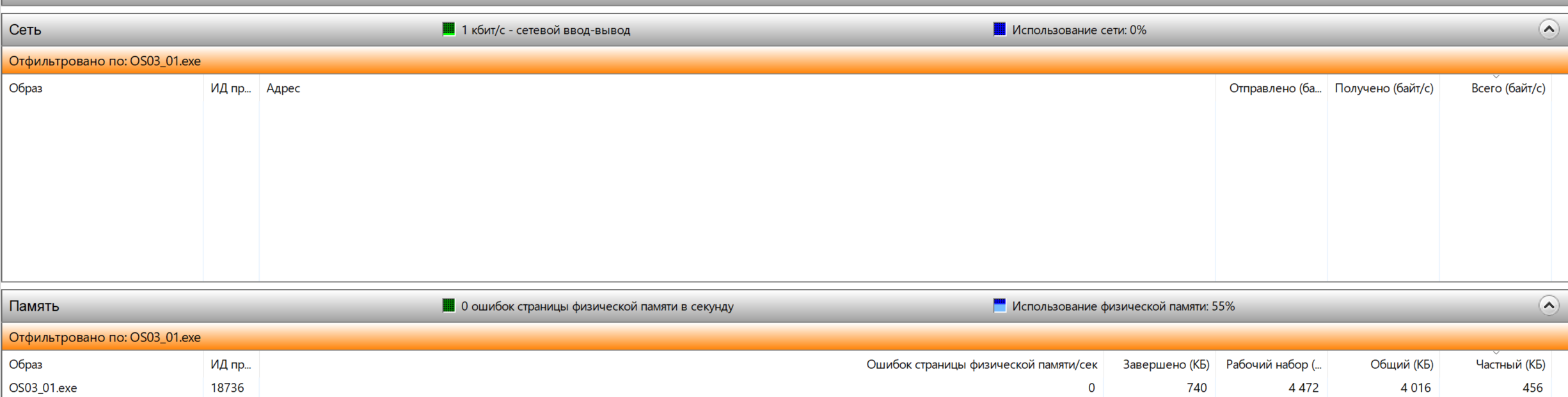


PowerShell ISE:



Performance Monitor:





**Задание 02**

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. Приложение OS03\_02 должно создавать два дочерних процесса OS03\_02\_1 и OS03\_02\_2.

lpApplicationName — имя исполняемого файла. Если NULL, используется командная строка.

lpCommandLine — командная строка, включая путь к приложению. Например, "C:\\Windows\\System32\\notepad.exe".

lpProcessAttributes — указатель на структуру безопасности для процесса. Можно оставить NULL.

lpThreadAttributes — указатель на структуру безопасности для потока. Можно оставить NULL.

bInheritHandles — флаг, указывающий, нужно ли наследовать дескрипторы из родительского процесса (TRUE/FALSE).

dwCreationFlags — флаги, определяющие поведение процесса (например, CREATE\_NEW\_CONSOLE).

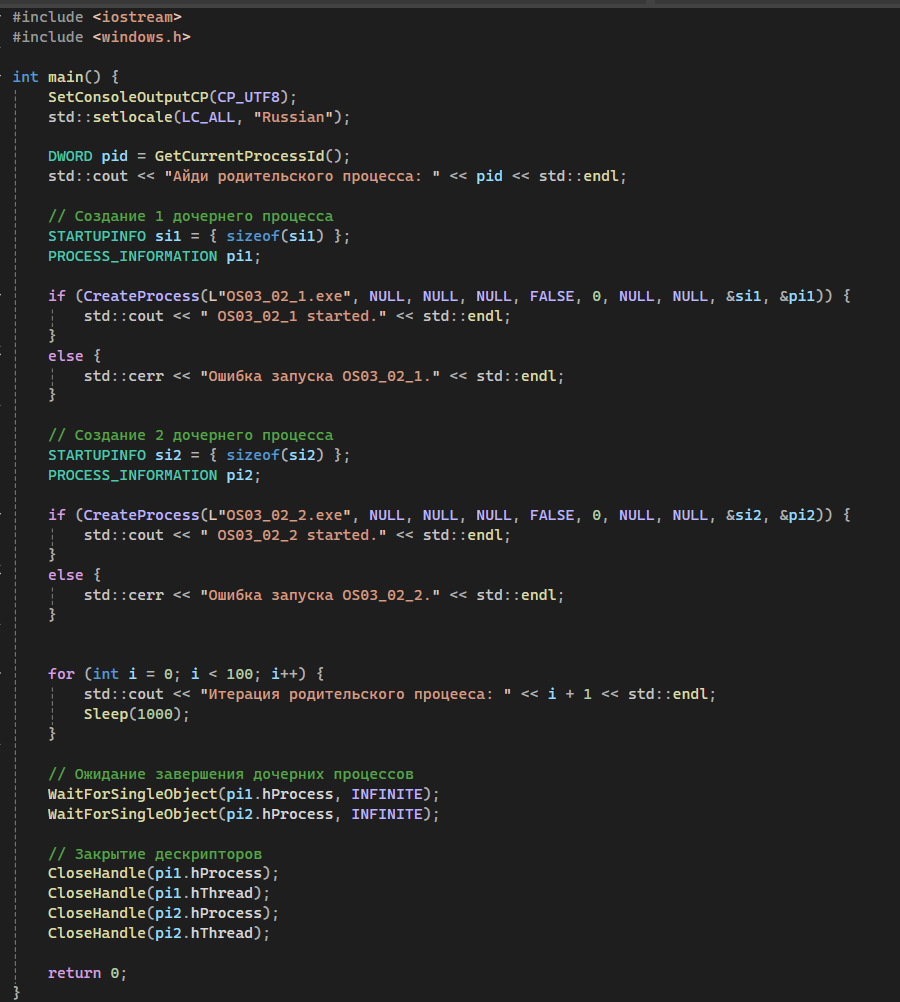
lpEnvironment — указатель на переменные окружения. Если NULL, используются переменные родителя.

lpCurrentDirectory — каталог, из которого будет запущен процесс. Если NULL, используется текущий каталог.

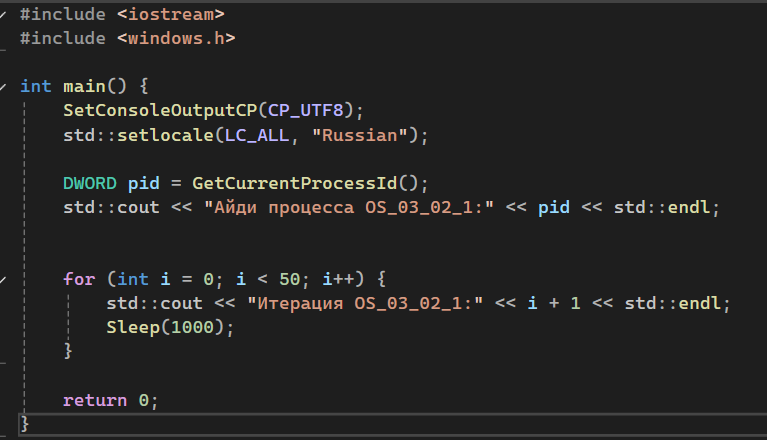
lpStartupInfo — структура STARTUPINFO, содержащая информацию о создании окна процесса.

lpProcessInformation — структура PROCESS\_INFORMATION, в которую записываются идентификаторы процесса и потока.

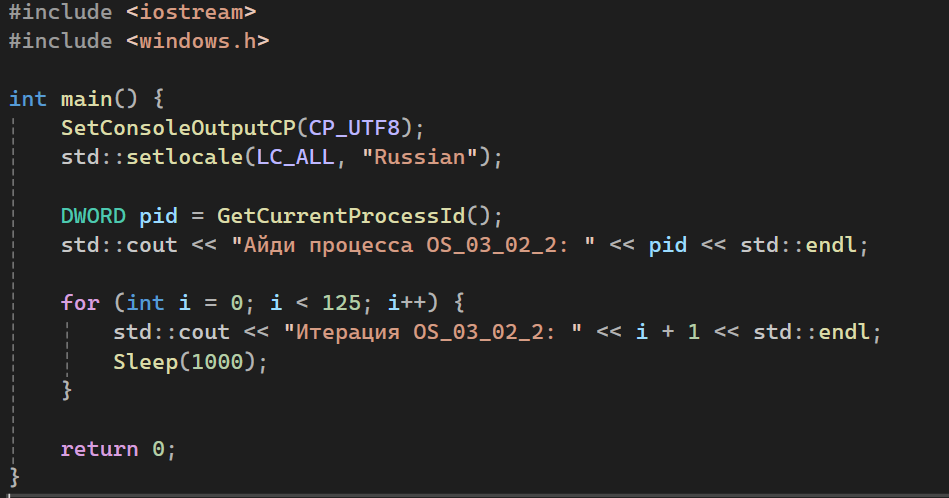
Этот код создаст проц



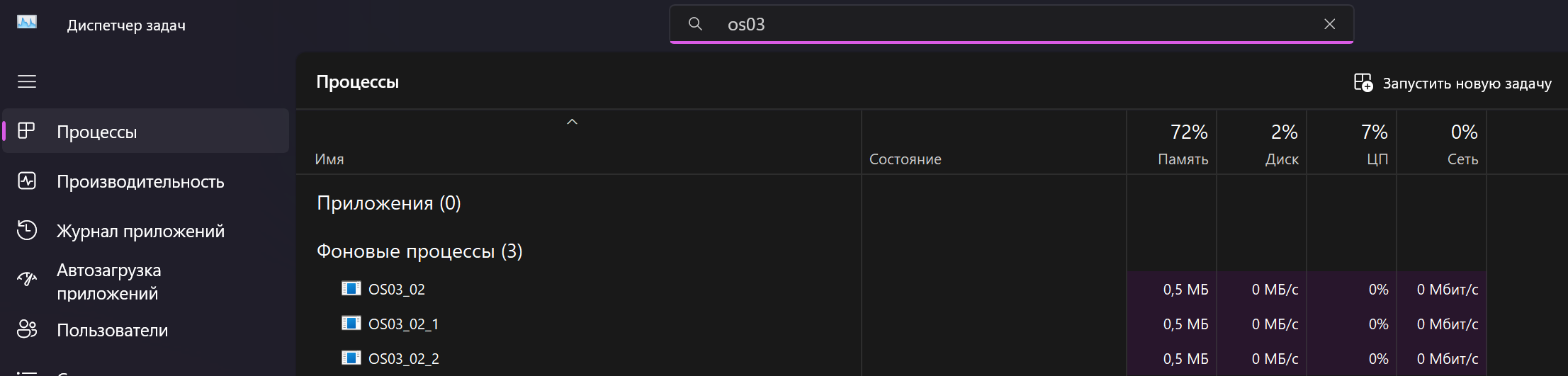
Процесс OS03\_02\_1 – консольное Windows-приложение, выполняющее цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.

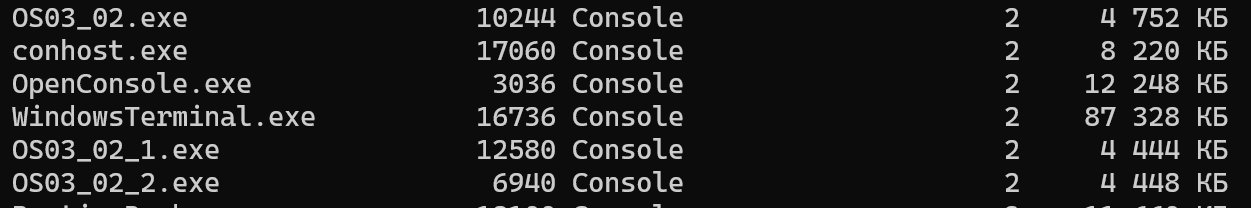


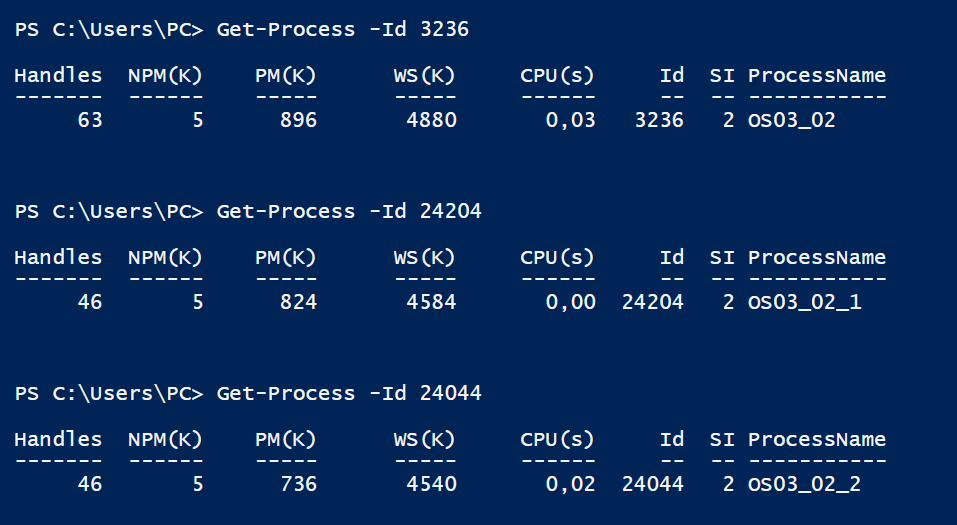
Процесс OS03\_02\_2 - консольное Windows-приложение выполняющее цикл 125 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.



Продемонстрируйте информацию о процессах OS03\_02, OS03\_02\_1 и OS03\_02\_2 с помощью утилит Task Manager, tasklist, PowerShell ISE и Performance Monitor.







Айди родительского процесса: 3236

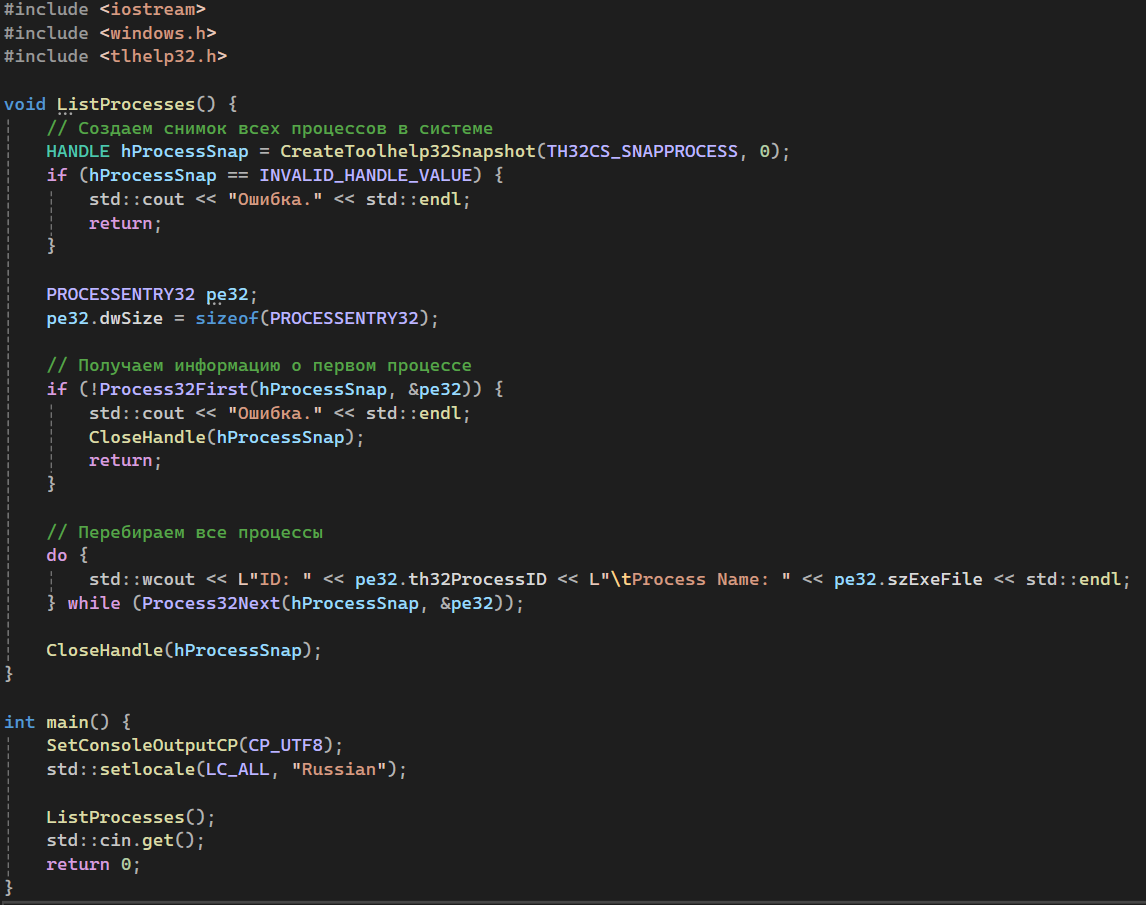
Айди процесса OS\_03\_02\_1:24204

Айди процесса OS\_03\_02\_2: 24044

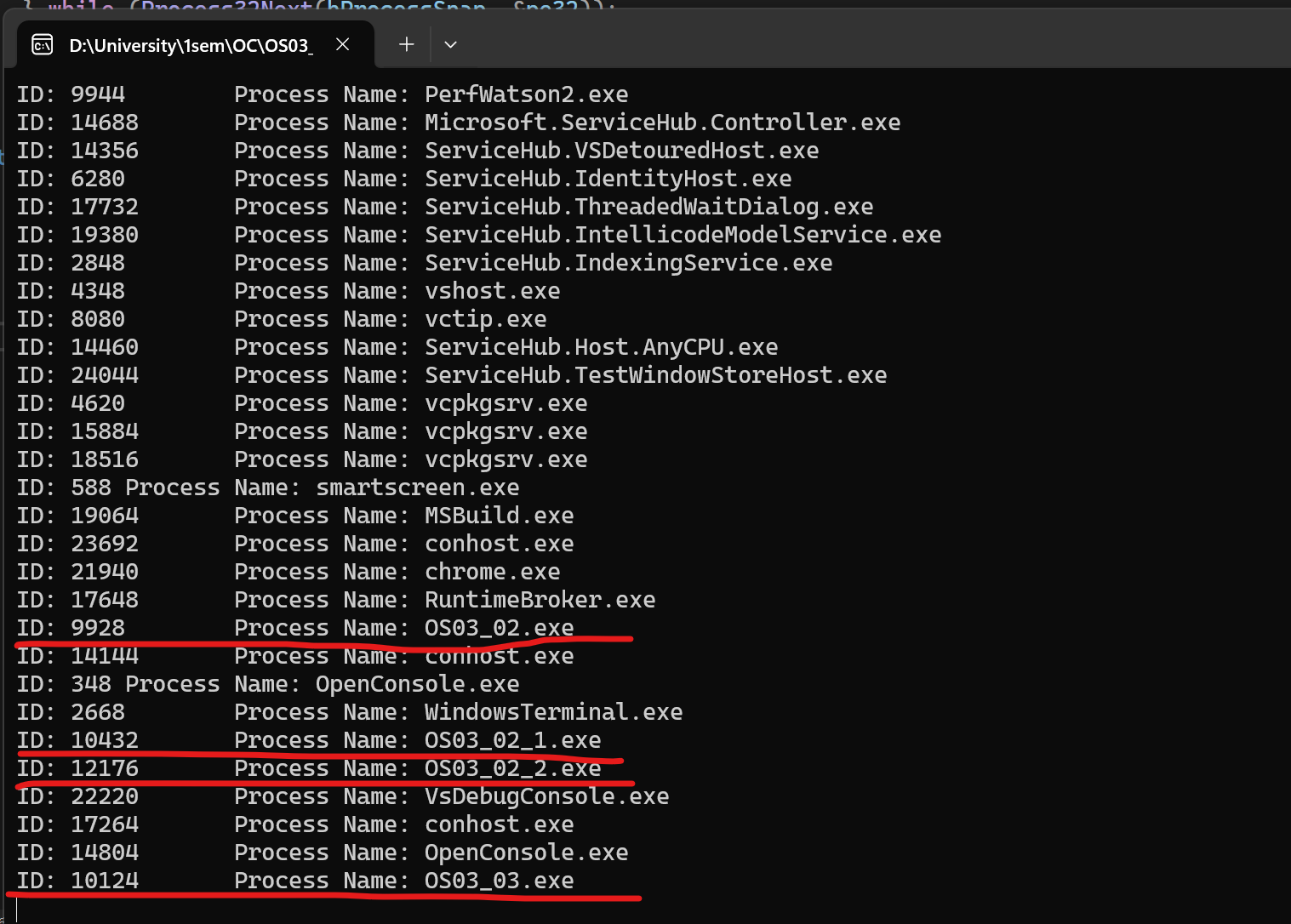


**Задание 03**

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

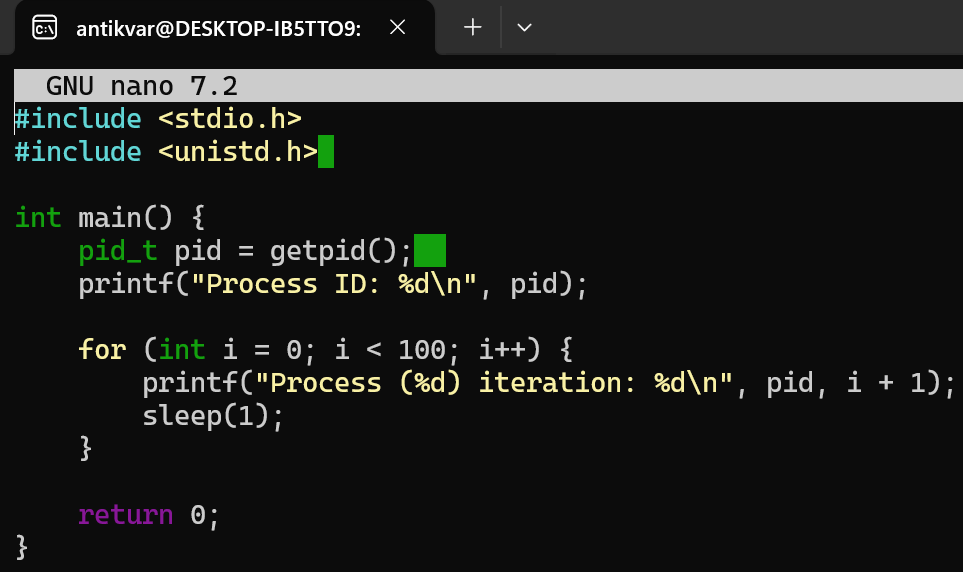


Запустите приложение OS03\_02 и продемонстрируйте с помощью приложения OS03\_03 в перечне процессов OS03\_02, OS03\_02\_1, OS03\_02\_2 и OS03\_03.



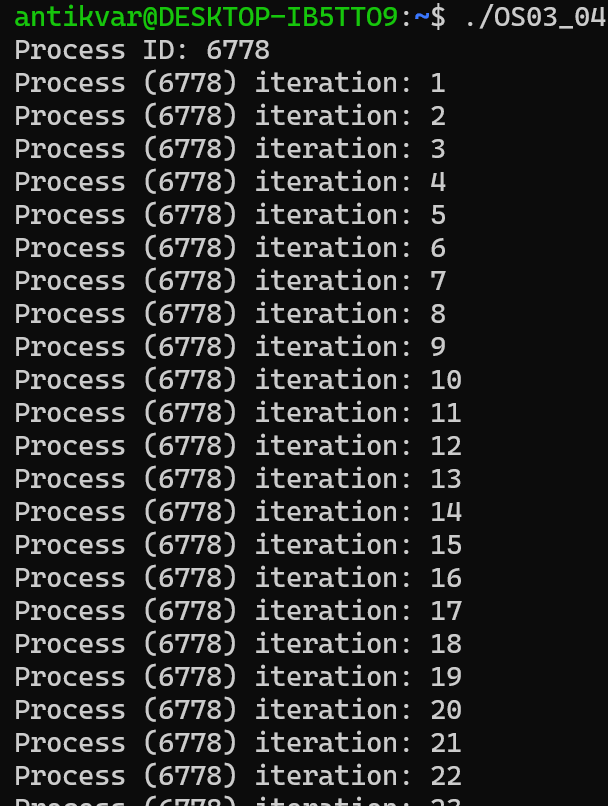
**Задание 04**

Разработайте консольное Linux-приложение OS03\_04 на языке С, выполняющее длинный цикл с временной задержкой и с выводом на консоль идентификатора процесса.

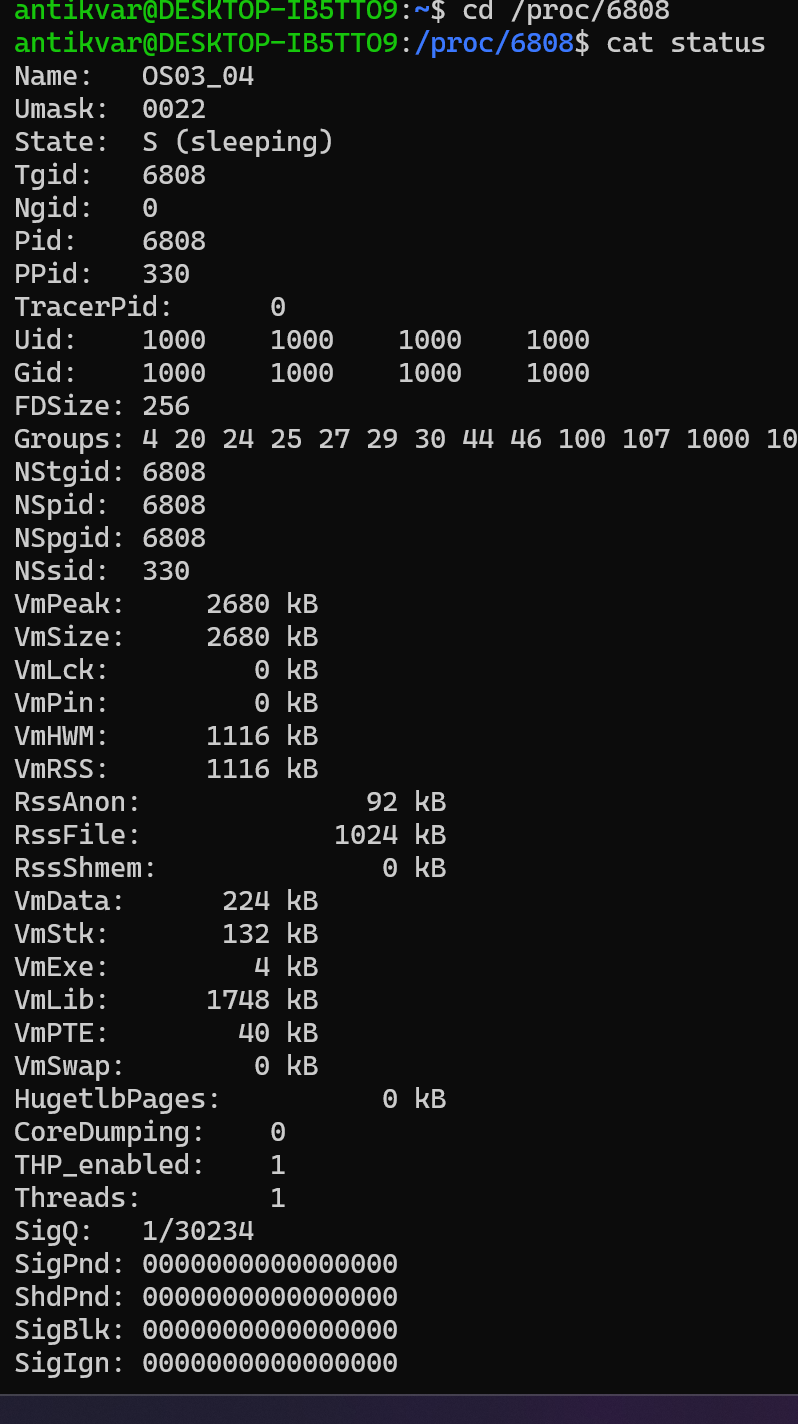


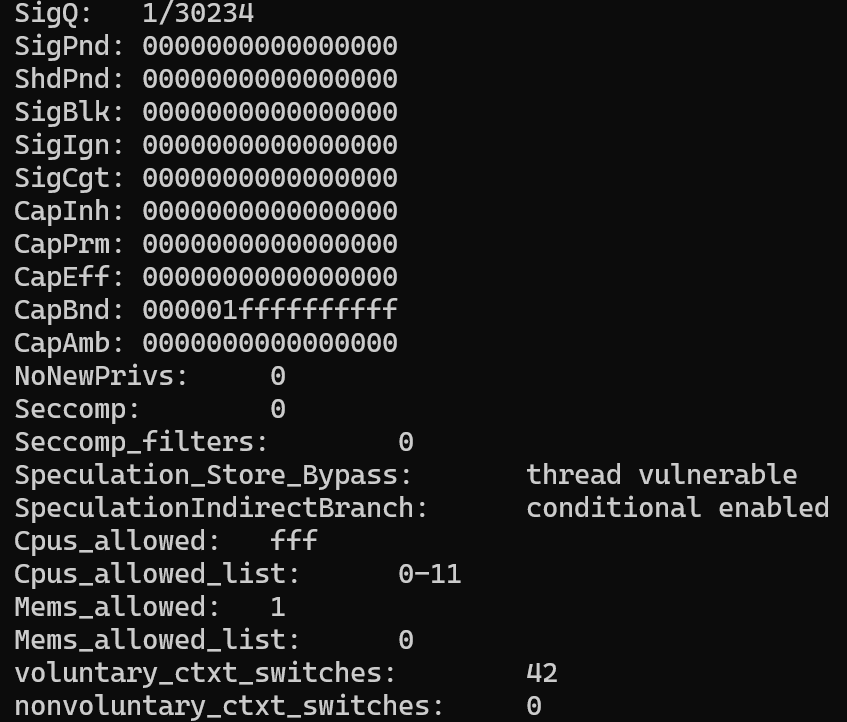
nano OS03\_04.c

gcc OS03\_04.c -o OS03\_04

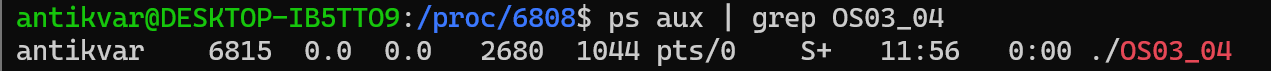


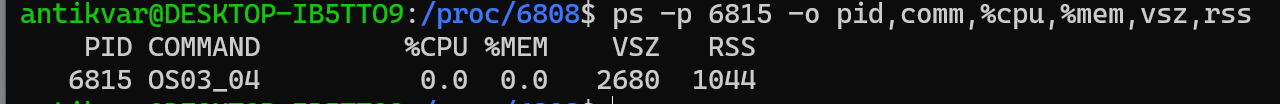
Продемонстрируйте информацию о процессе OS03\_04 с помощью файловой системы /proc.





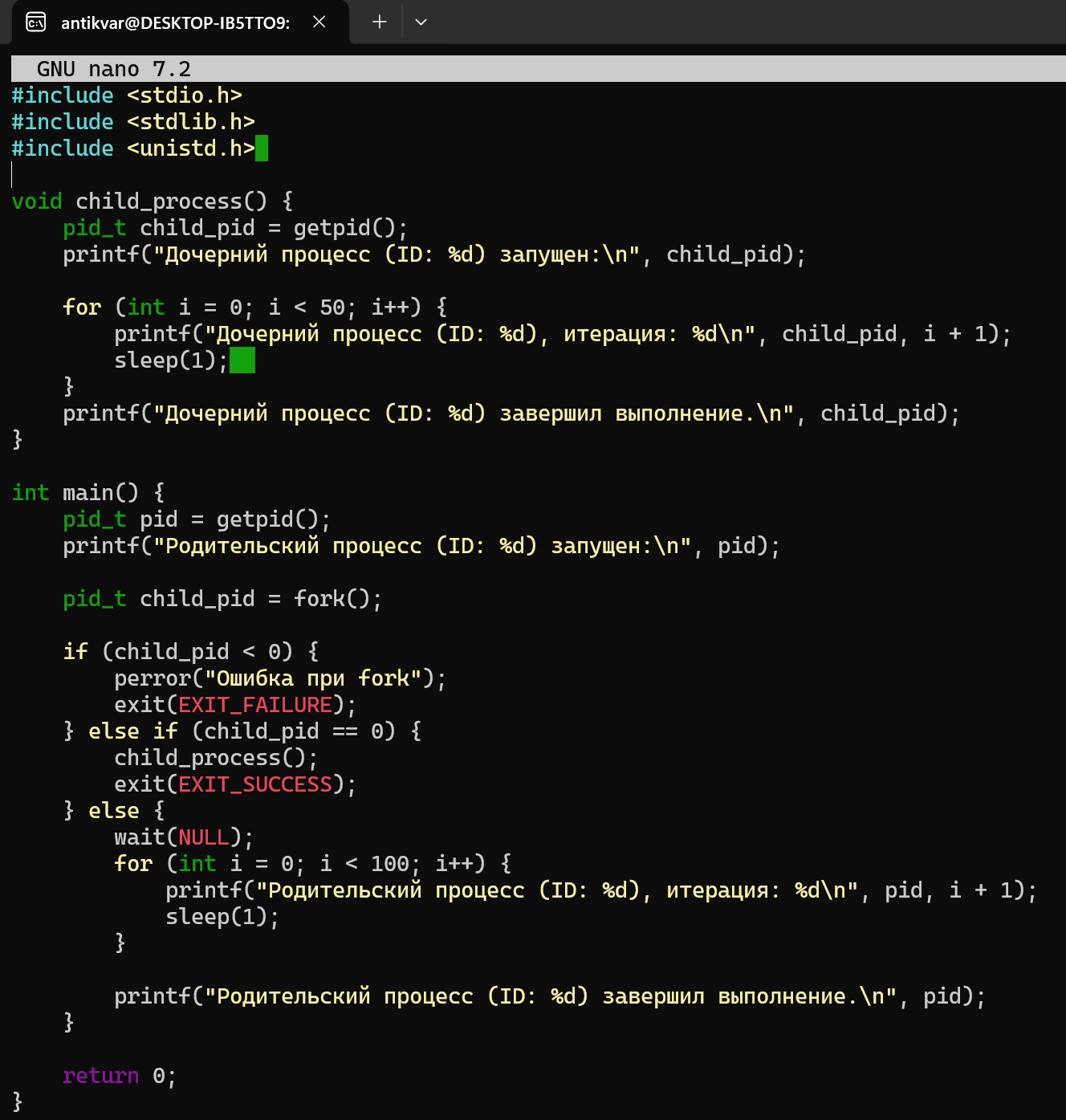
Продемонстрируйте информацию о процессе OS03\_04 с помощью утилиты ps.

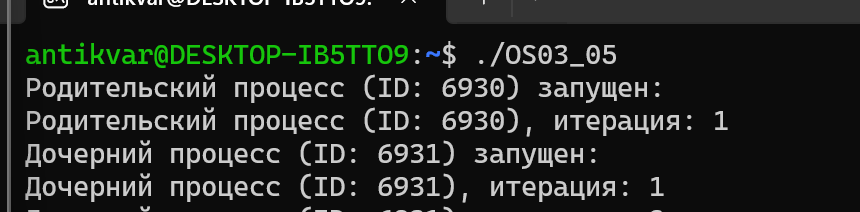




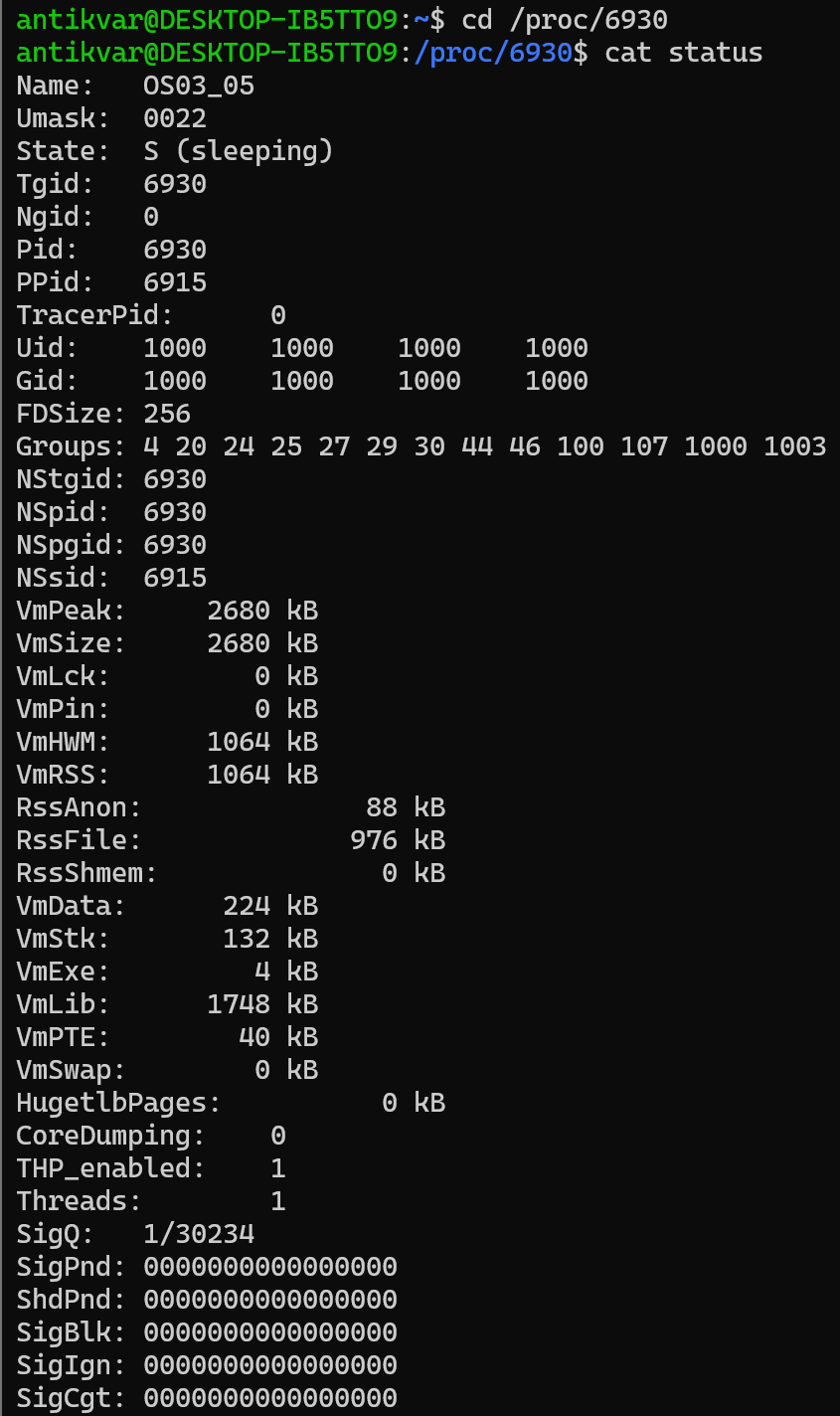
**Задание 05**

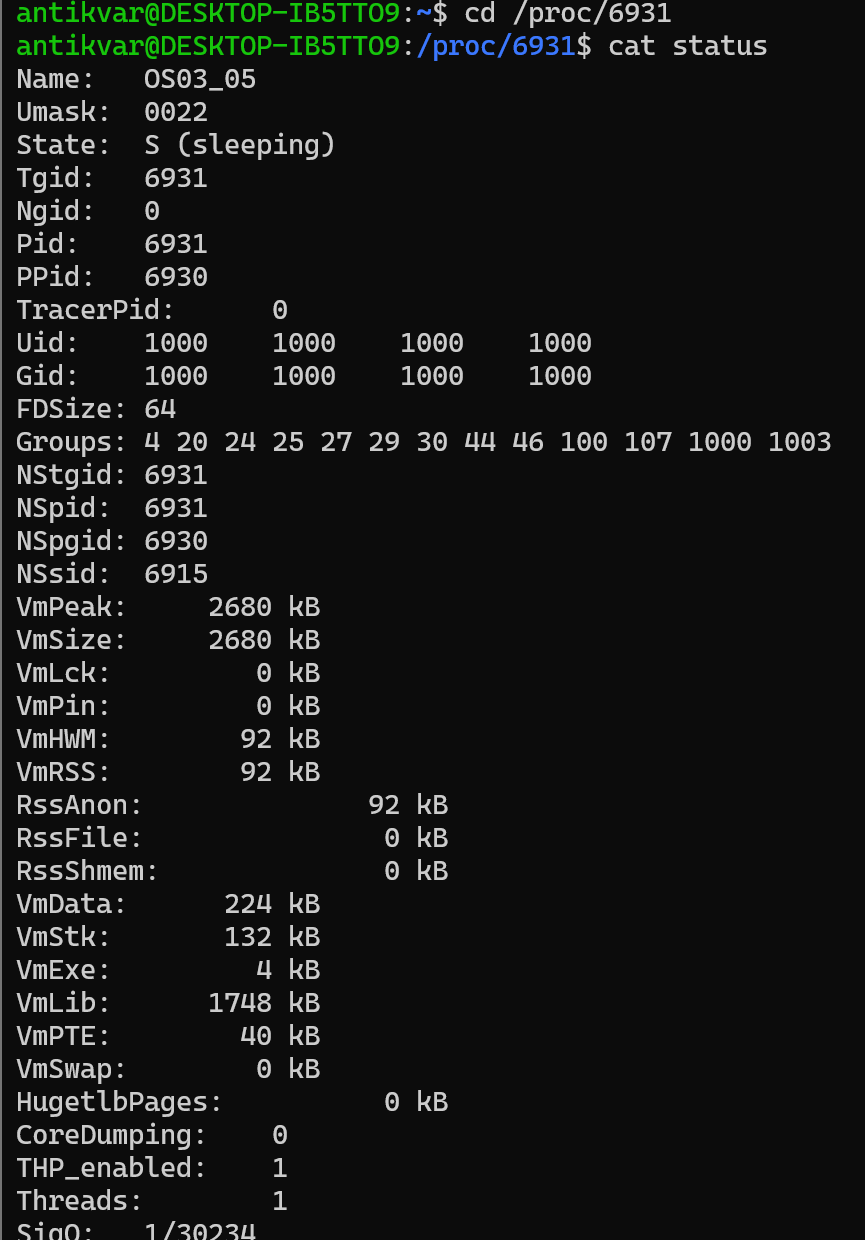
Разработайте консольное Linux-приложение OS03\_05 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса. Приложение OS03\_05 должно создавать один дочерний процесс OS03\_05\_1 с помощью системного вызова fork Процесс OS03\_05\_1 в этом случае не является отдельным модулем, а встроен (fork) в программный модуль OS03\_05. Процесс OS03\_05\_1 - консольное Linux-приложение, выполняющее цикл 50 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.



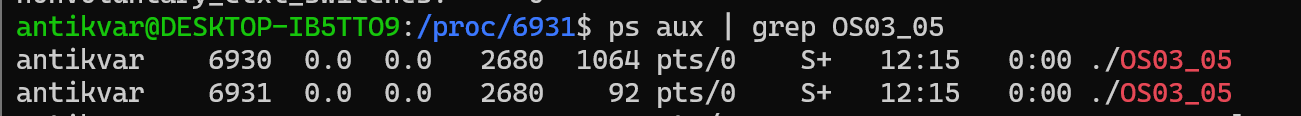


Продемонстрируйте информацию о процессах OS03\_05 и OS03\_05\_1 с помощью файловой системы /proc.



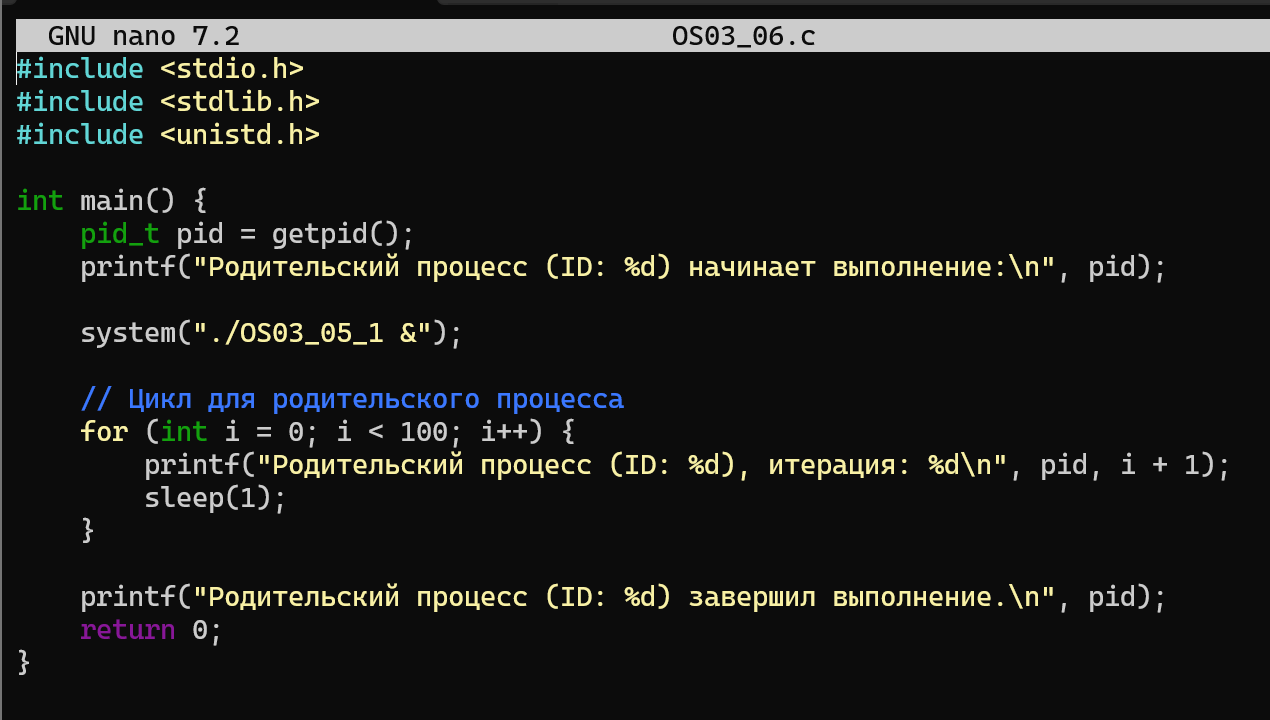


Продемонстрируйте информацию о процессах OS03\_05 и OS03\_05\_1 с помощью утилиты ps.

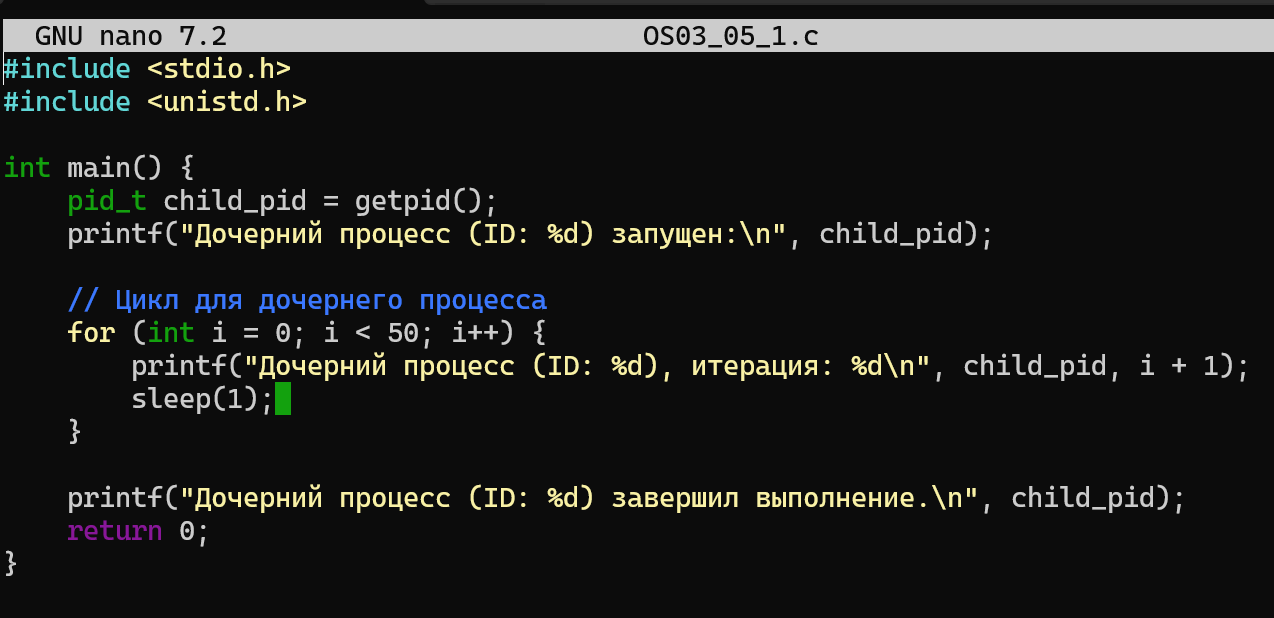


**Задание 06**

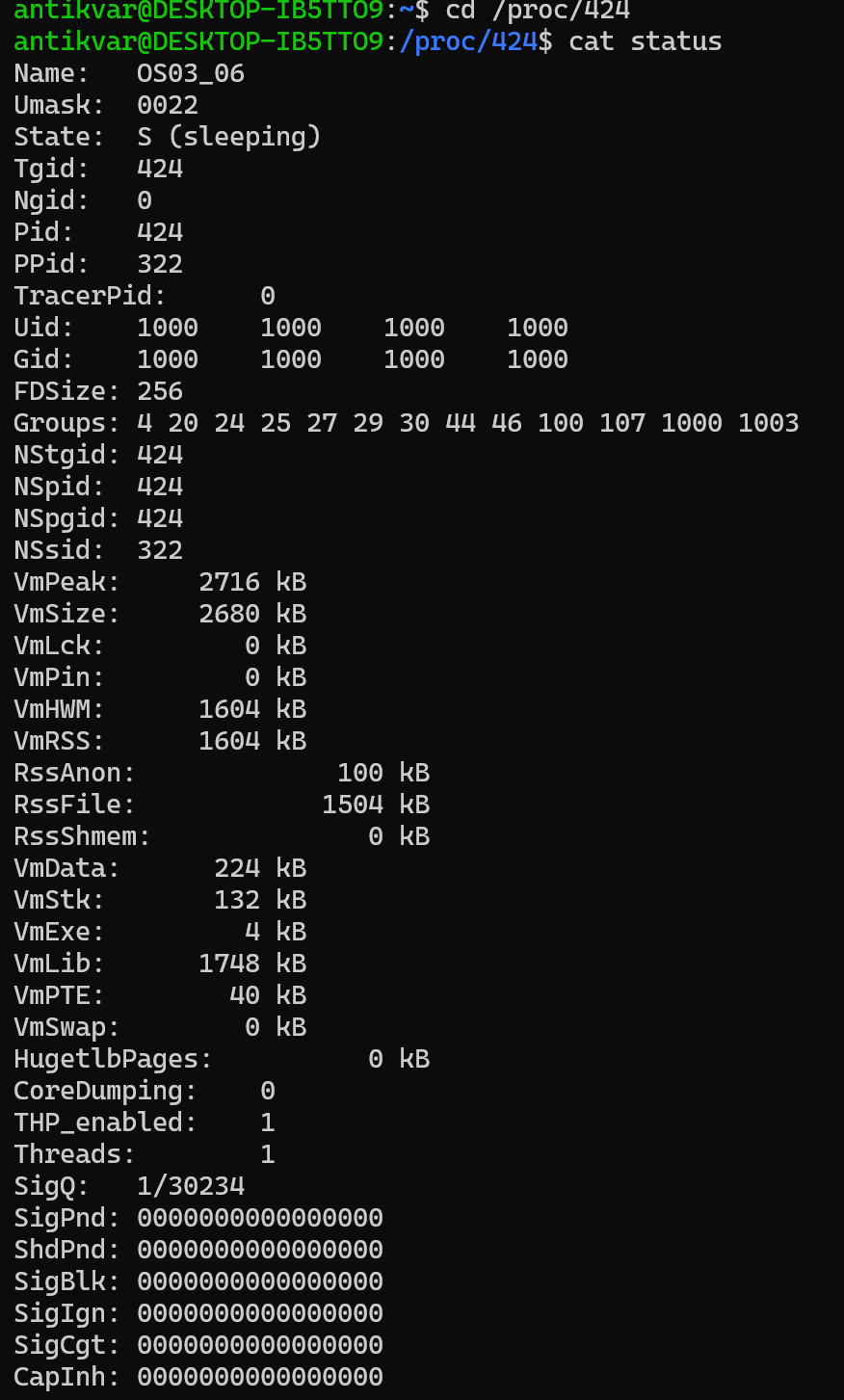
Разработайте консольное Linux-приложение OS03\_06 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса.

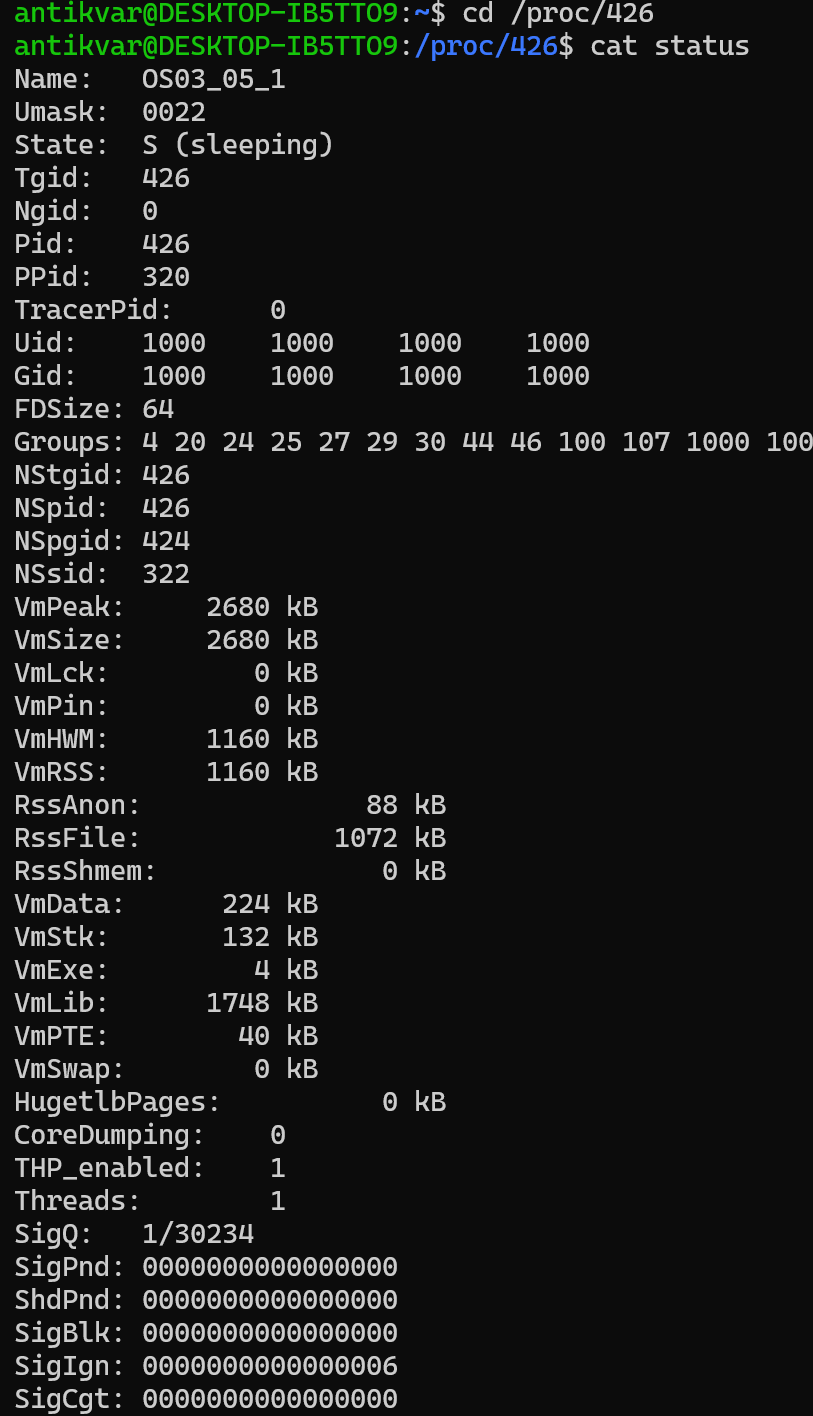


Приложение OS03\_06 должно создавать один дочерний процесс OS03\_05\_1 (отдельный модуль) с помощью системного вызова system.

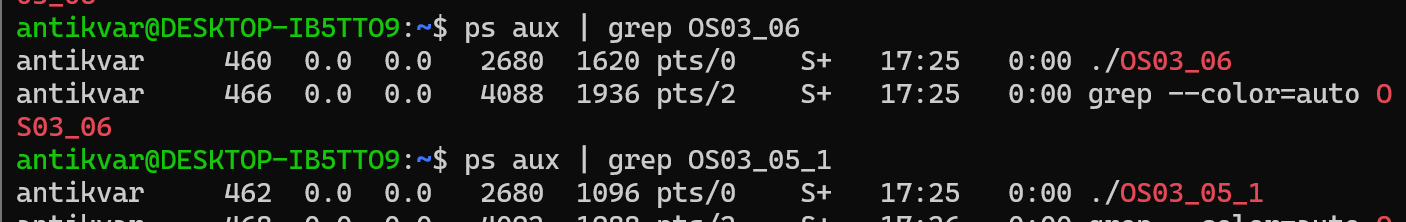


Продемонстрируйте информацию о процессах OS03\_06 и OS03\_05\_1 с помощью файловой системы /proc.





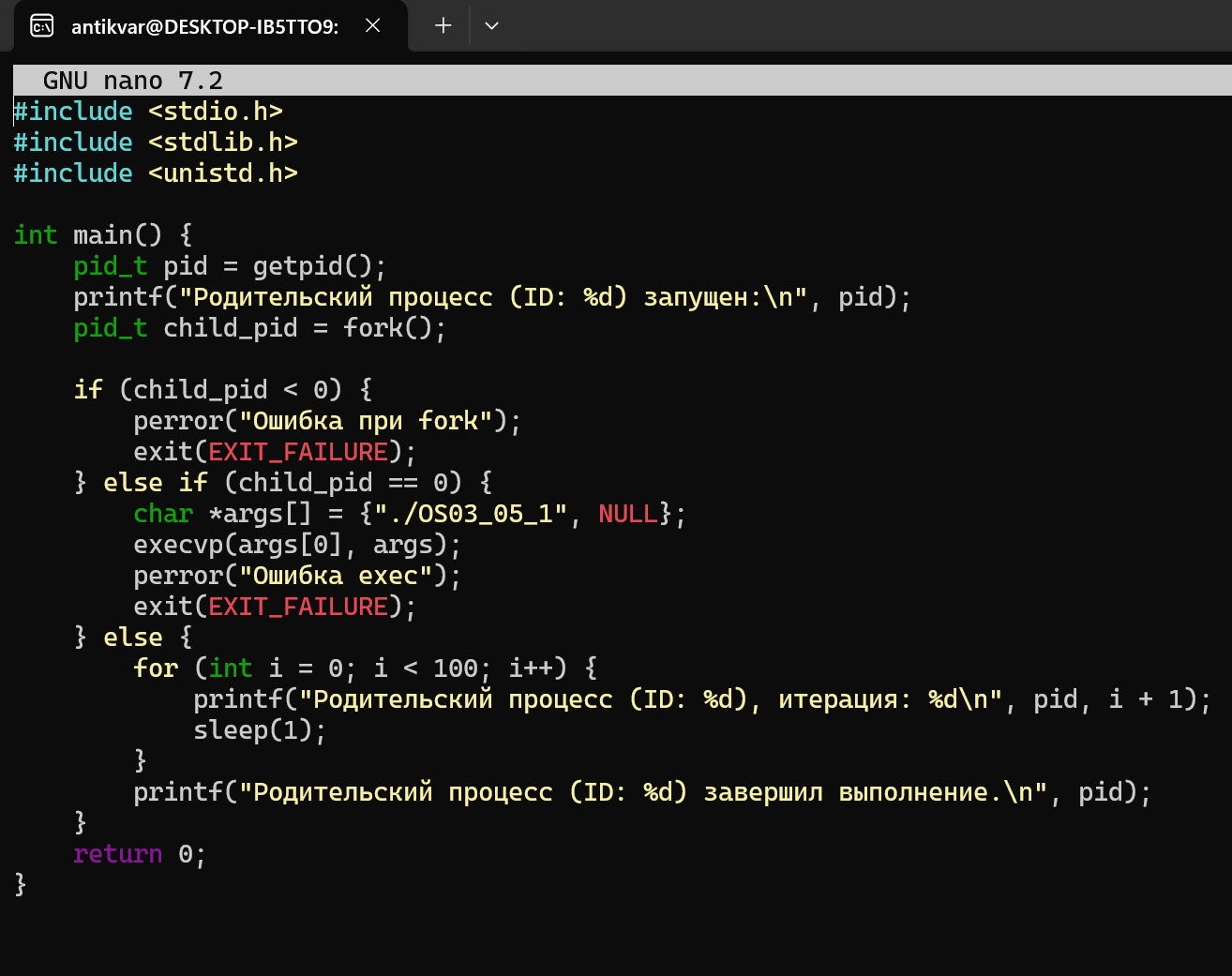
Продемонстрируйте информацию о процессах OS03\_06 и OS03\_05-1 с помощью утилиты ps.



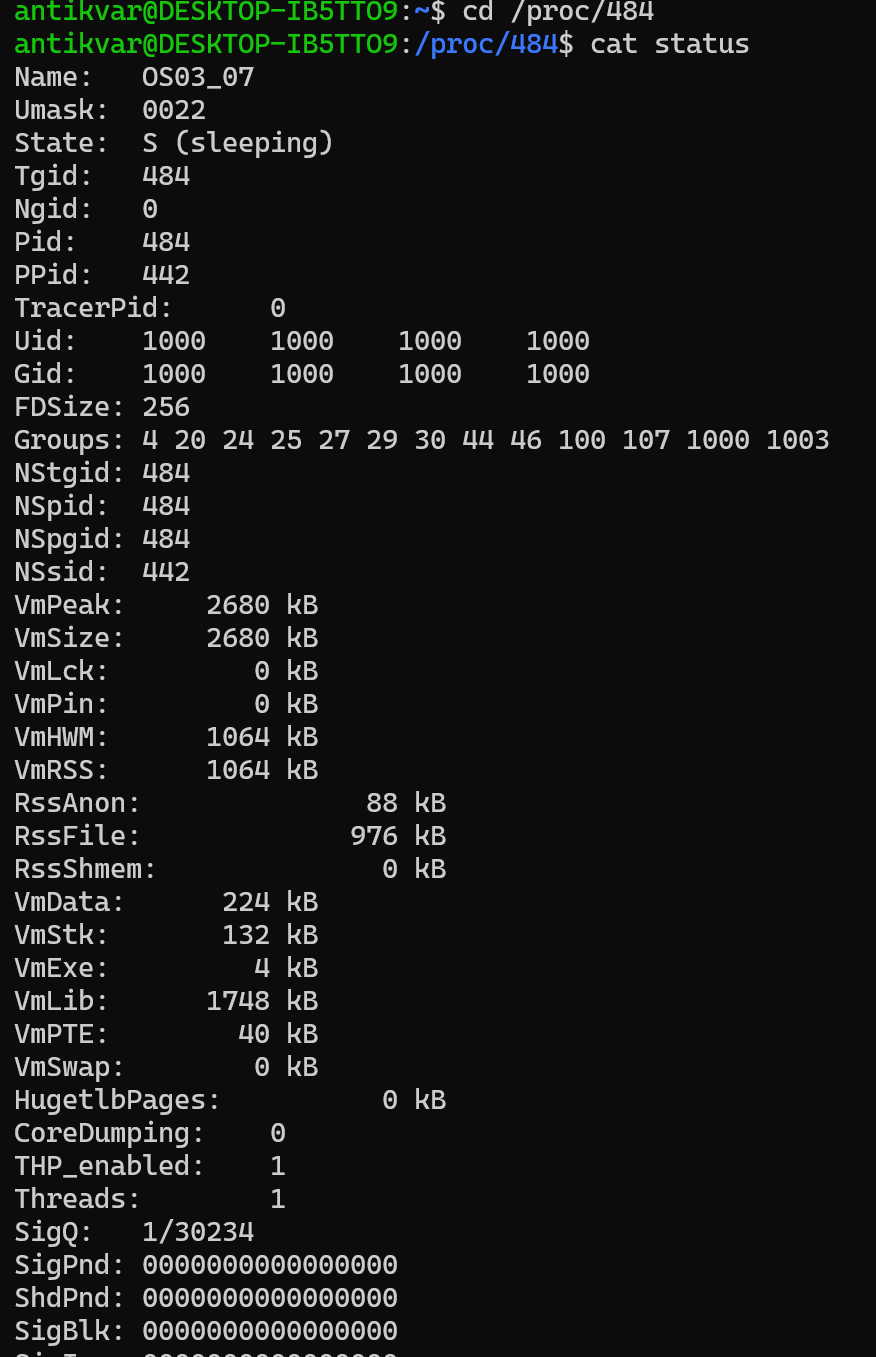
**Задание 07**

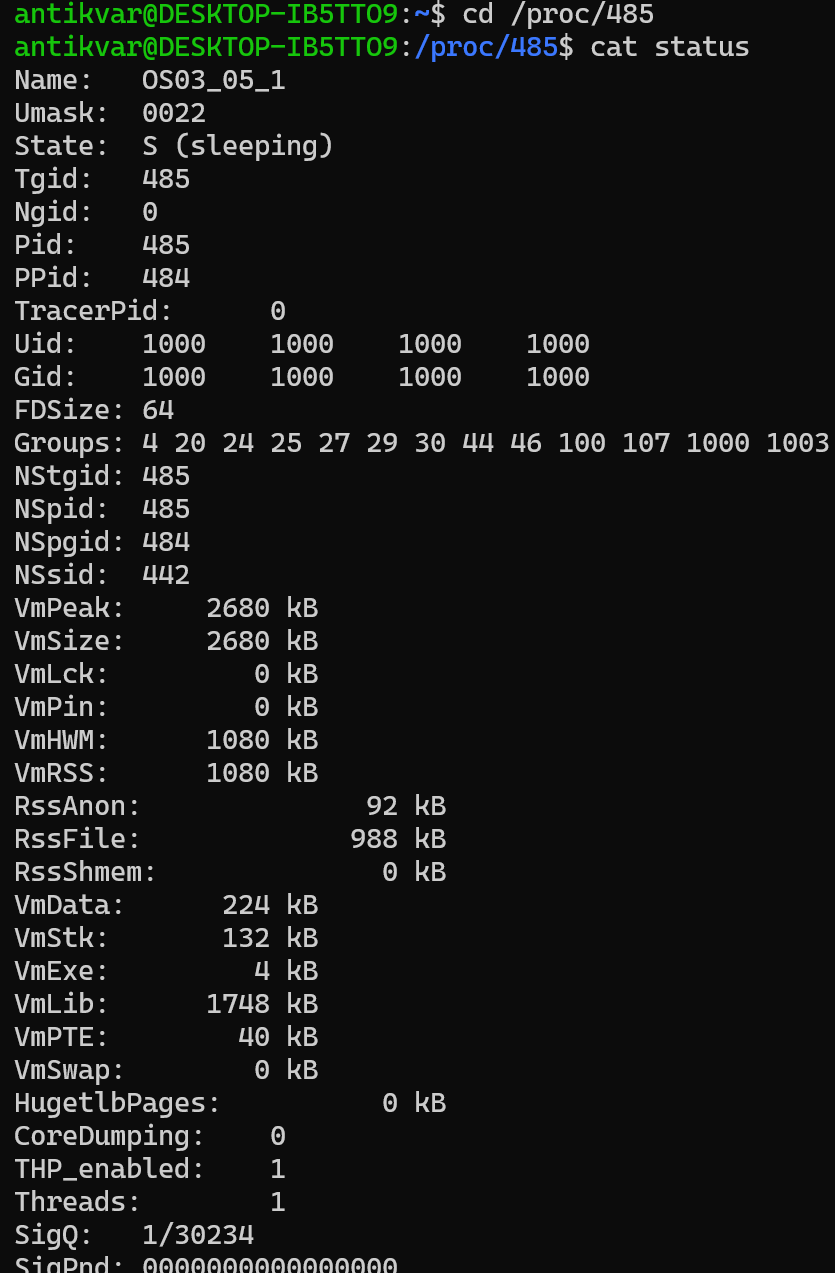
Разработайте консольное Linux-приложение OS03\_07 на языке С, выполняющее цикл 100 итераций с временной задержкой в 1 сек. с выводом на консоль идентификатора процесса

Приложение OS03\_07 должно создавать один дочерний процесс OS03\_05\_1 (отдельный модуль) с помощью системного вызова exec.

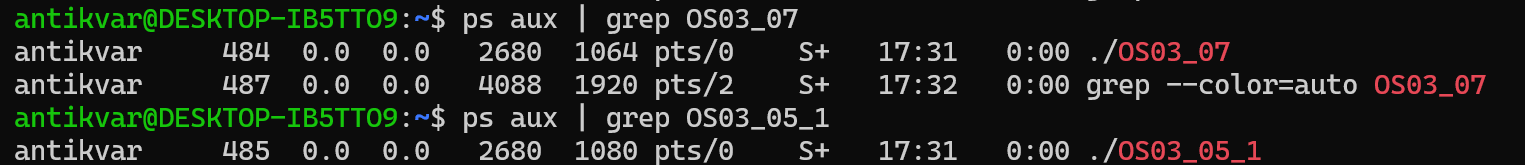


Продемонстрируйте информацию о процессах OS03\_07 и OS03\_05\_1 с помощью файловой системы /proc.

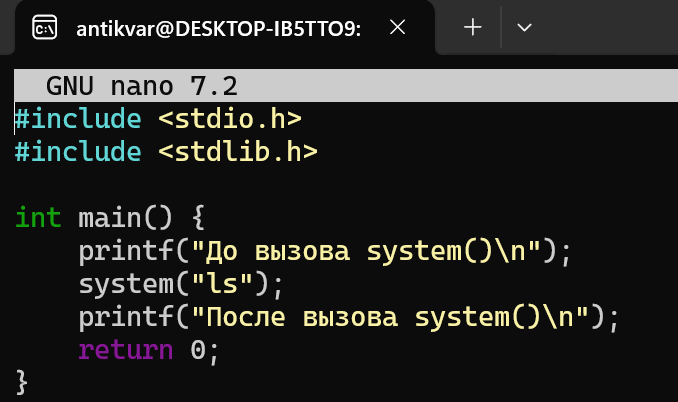


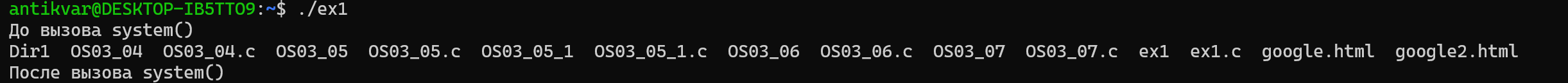


Продемонстрируйте информацию о процессах OS03\_07 и OS03\_05-1 с помощью утилиты ps.

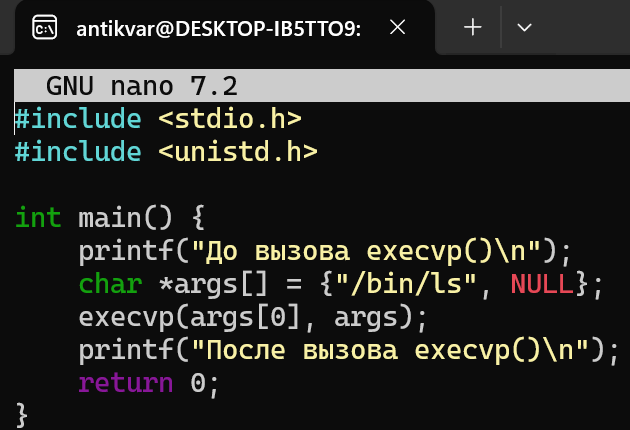


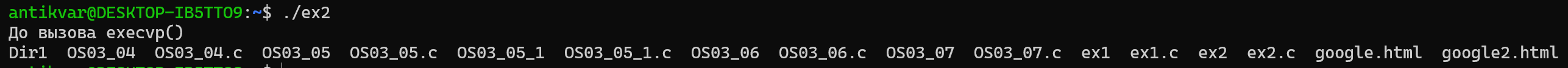
Продемонстрируйте разницу системных вызовов system и exec.





system("ls") выполняет команду ls в командной оболочке (как если бы вы её ввели в терминале). После завершения команды программа продолжает своё выполнение. В этом примере сначала выводится "До вызова system()", затем выполняется команда ls, которая покажет список файлов в текущей папке. После этого программа выводит "После вызова system()".





Программа вызывает exec(), текущий процесс заменяется другой программой, и после вызова exec() программа больше не продолжает свою работу. Используется, когда нужно заменить процесс без созия нового.

Exec()