ОС, ПОИТ-3, Лекция 04

1. **OS:** предыдущие лекции, повторение

* процесс – единица работы OS;
* поток – единица работы процессора, ему выделяется квант процессорного времени;
* процессы и потоки создаются с помощью системных вызовов;
* процессы и потоки – объекты ядра операционной системы;
* в рамках любого процесса, как минимум 1 поток – main;
* процессы изолированы друг от друга;
* потоки в рамках одного процесса имеют доступ ко всем общим ресурсам процесса;
* процессы и потоки имеют контекст – память, в которой сохраняются данные процесса или потока необходимые для продолжения выполнения;
* процессу соответствует исполняемый файл;
* потоку соответствует потоковая функция (фрагмент кода процесса);
* системные процессы (windows-сервисы, Linux-демоны);
* системные процессы и потоки ядра OS;
* OS API позволяет управлять процессами и потоками (создавать, удалять, приостанавливать, …);
* процессу доступны три потока данных: ввода, вывода, ошибок;
* завершение родительского процесса или потока приводит к завершению всех дочерних процессов и потоков;
* многопоточность приоритетная и кооперативная;
* в OS есть процесс инициализации - прародитель всех процессов OS.

**OS Process and Threads dispatching**

1. **OS:** циклическоепланирование

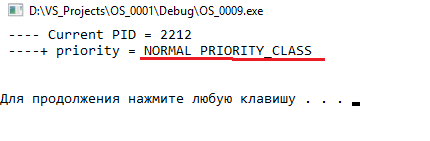


1. **OS:** приоритетное планирование



1. **OS:** кооперативное планирование
2. **OS:** гарантийное планирование, OS реального времени.
3. **OS:** автоматическое динамическое повышение приоритета.
4. **OS:** Windows,приоритет процесса





1. **OS:** Windows: значение приоритетов возрастающая (0-31)
2. **OS:** Windows**,** приоритеты процесса

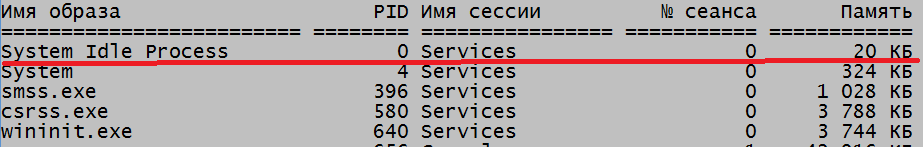
* фоновые процессы;
* процессы с нормальным приоритетом;
* процессы с высоким приоритетом;
* процессы реального времени.

1. **OS:** системы реального времени, управление, операционные системы реального времени, обратная связь, гарантированный интервал выполнения.



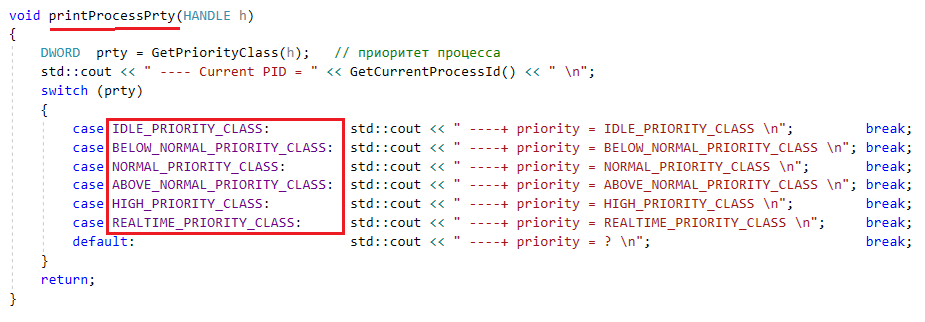


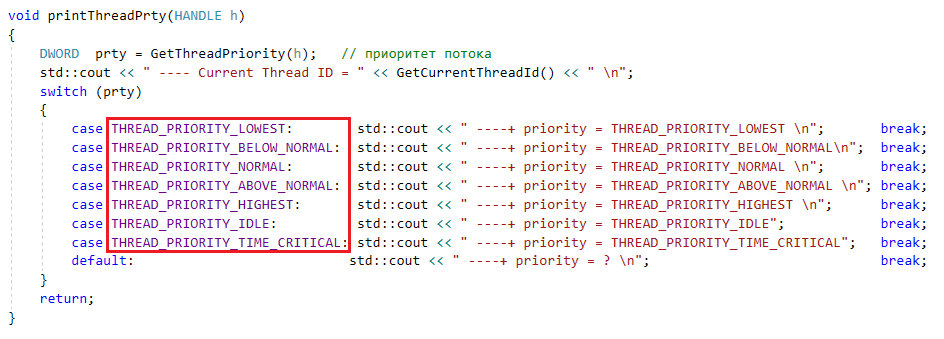
1. **OS:** Windows**,** System Idle Process

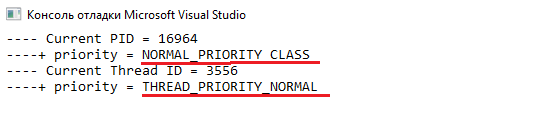


1. **OS:** Windows**,** уступить процессор Sleep(0)
2. **OS:** Windows**,** приоритет процесса и потока

****

****

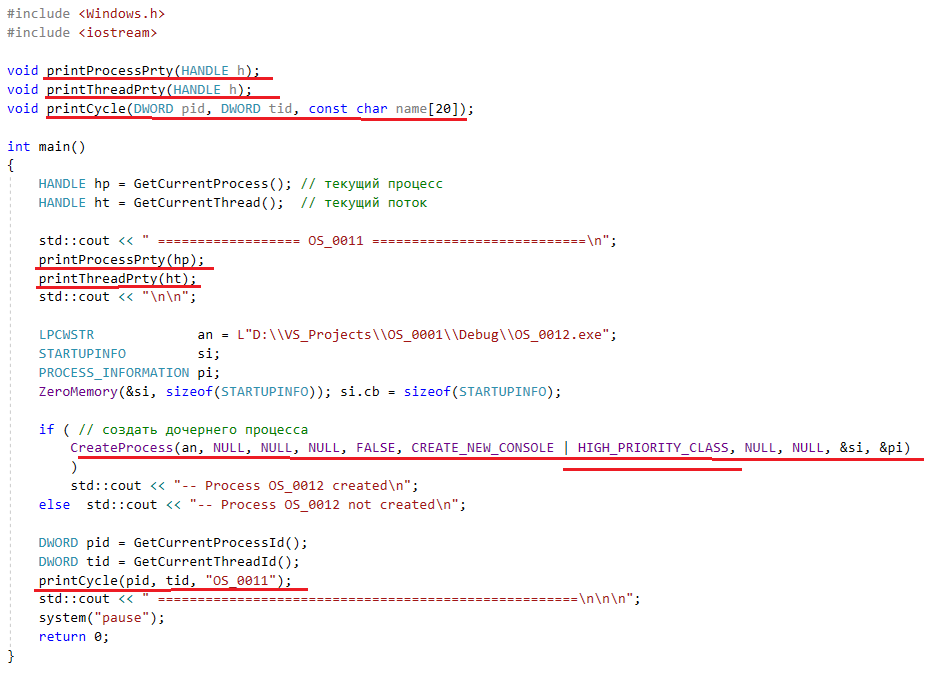
****

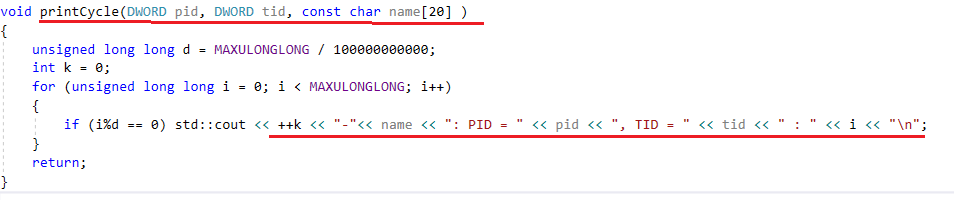
****

1. **OS:** Windows**,** базовые приоритеты потока

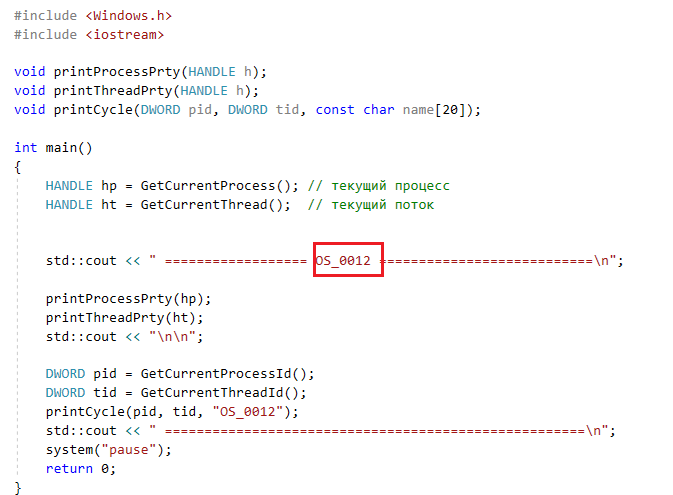
****

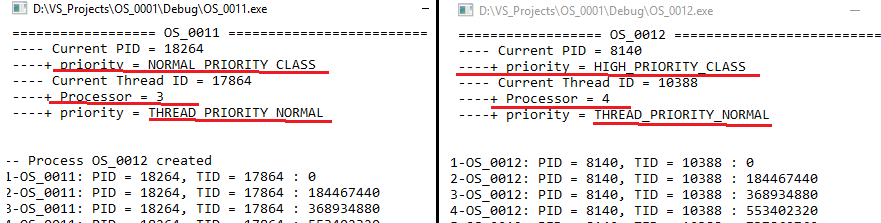
1. **OS:** Windows, CreateProcess(…HIGH\_PRIORITY\_CLASS…), GetPriorityClass(), GetThreadPriority(), SetThreadIdealProcessor(…MAXIMUM\_PROCESSORS)

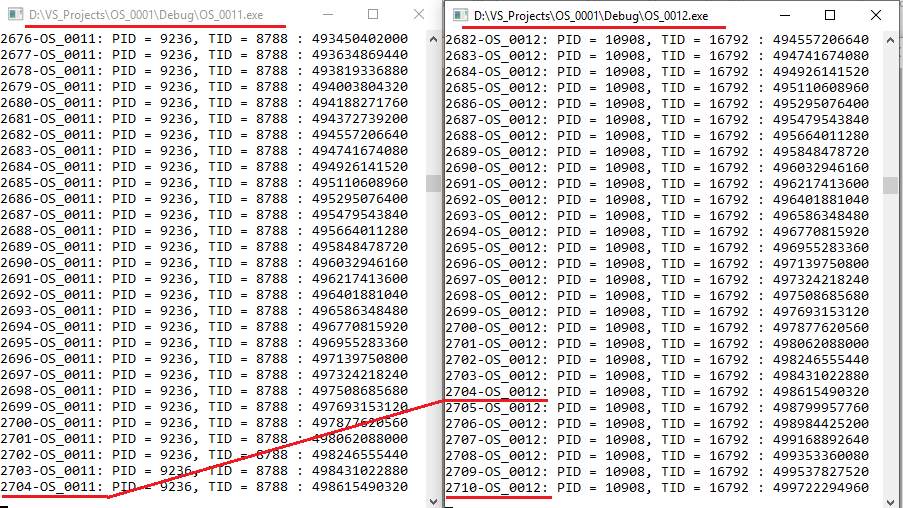


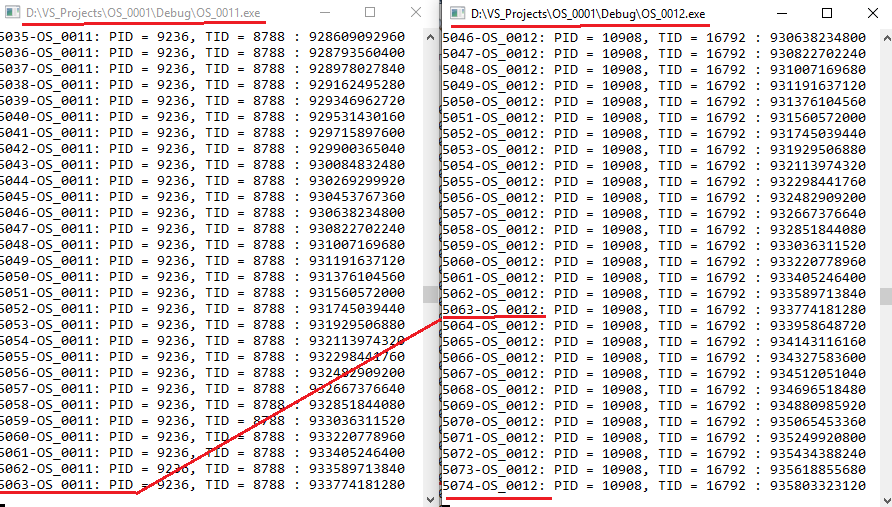






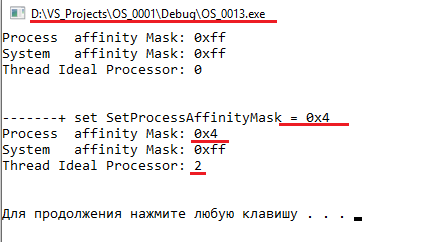


****

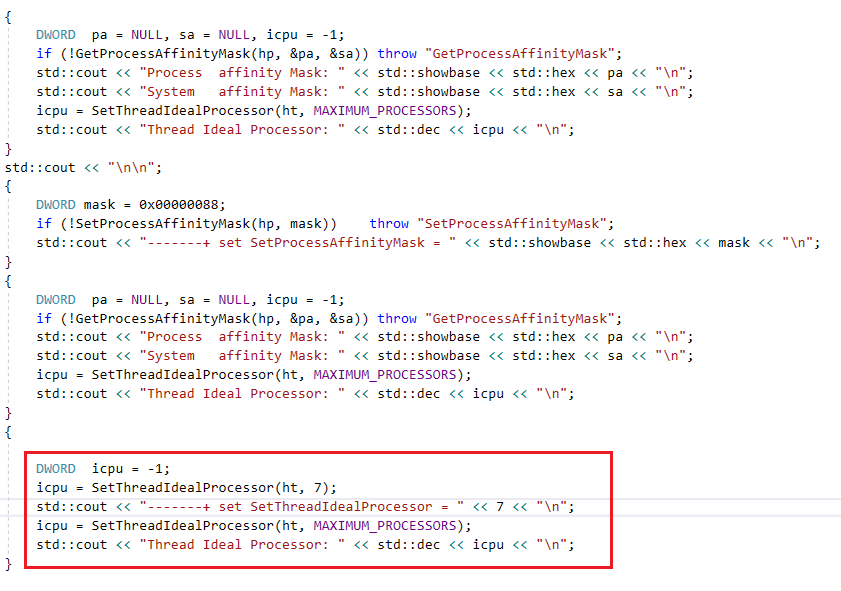


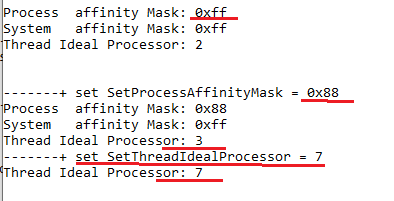
1. **OS:** GetProcessAffinityMask(), SetProcessAffinityMask(),  SetThreadIdealProcessor(…MAXIMUM\_PROCESSORS),



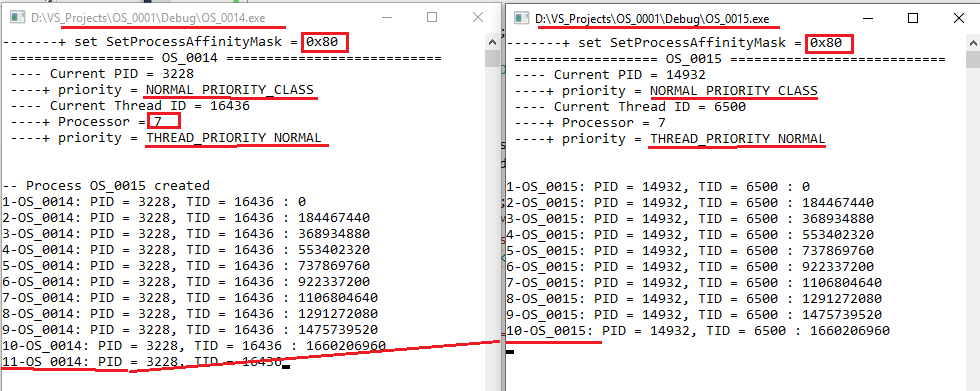


1. **OS:** Windows, SetProcessAffinityMask(), SetThreadIdealProcessor()

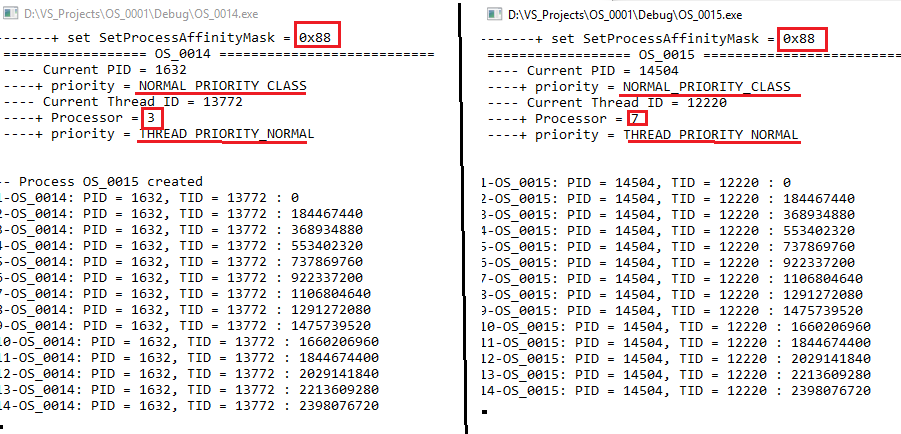
****

****

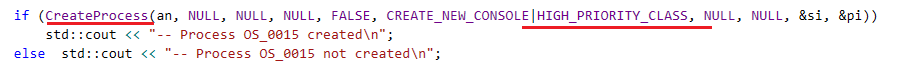
1. **OS:** Windows, один процессор, два процесса с одним приоритетом.

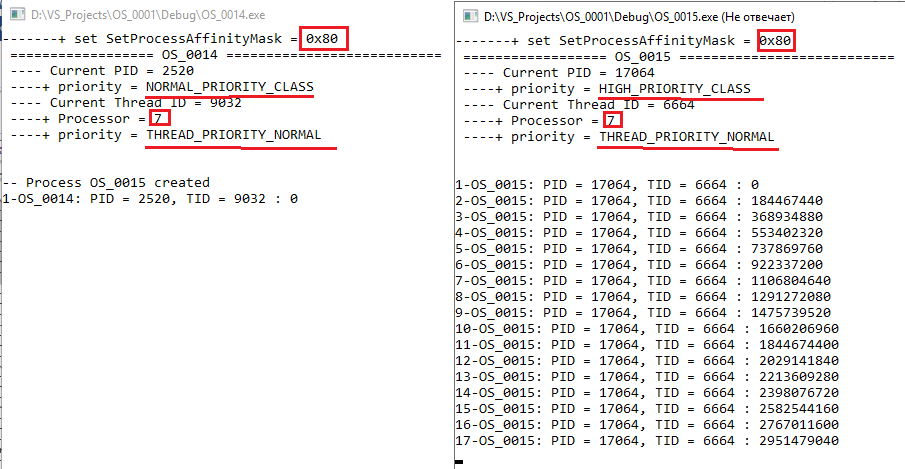


1. **OS:** Windows, два процессора, два процесса с одним приоритетом.

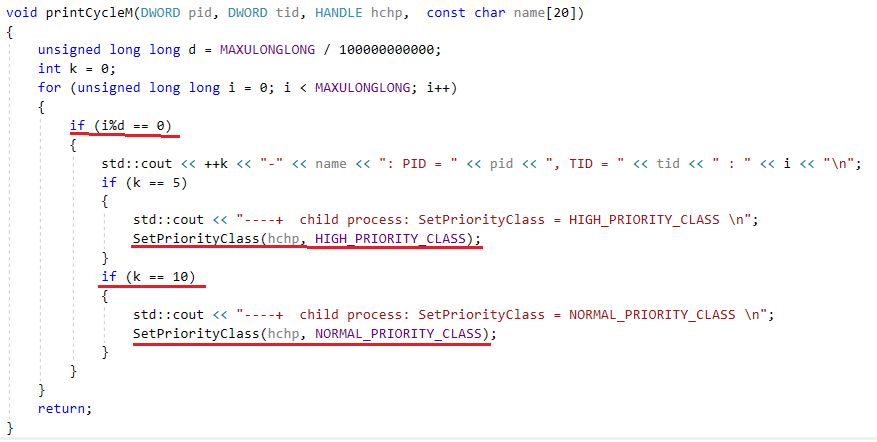


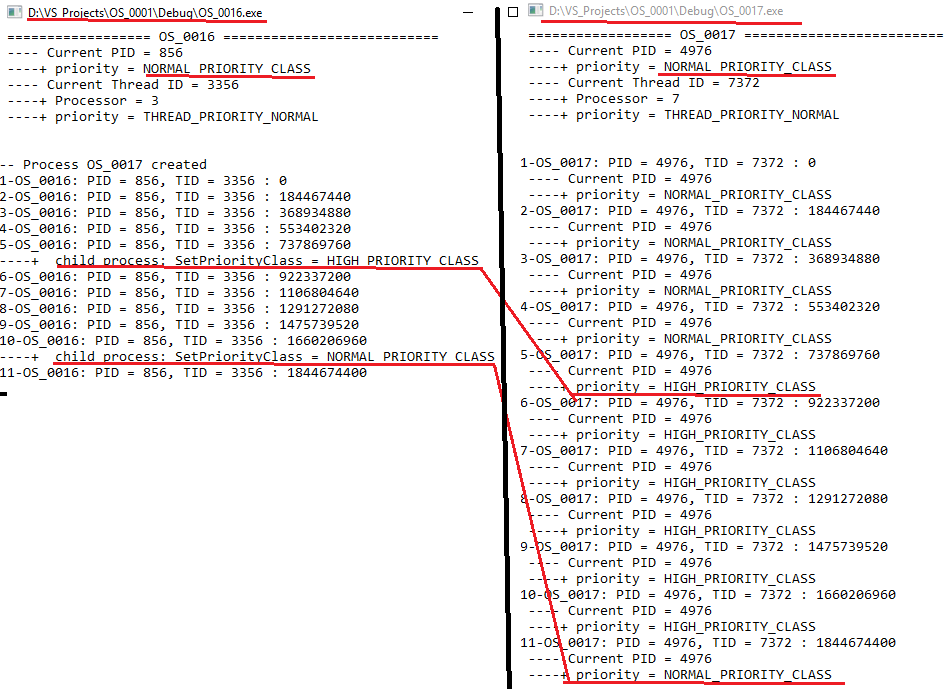
1. **OS:** Windows, один процессор, два процесса с разными приоритетами.



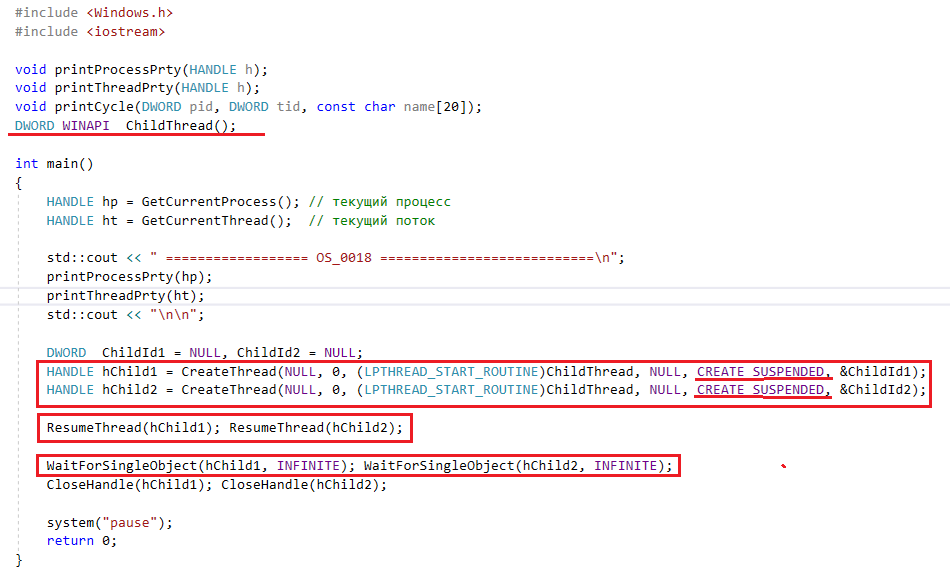


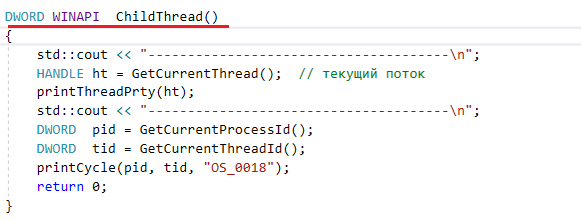
1. **OS:** Windows, SetPriorityClass()

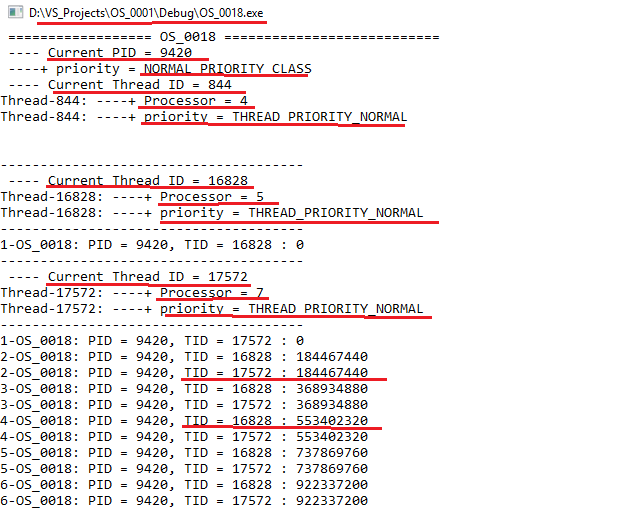




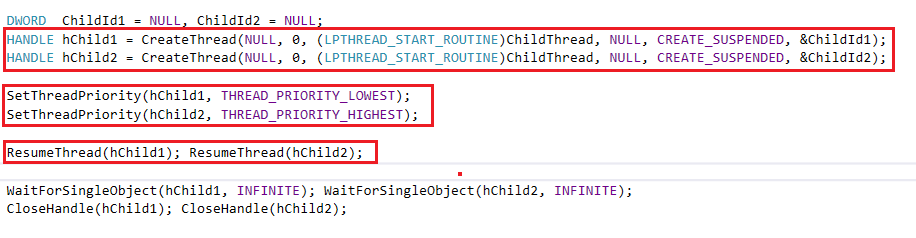
1. **OS:** Windows, CreateThread, ResumeThread, WaitForSingleObject

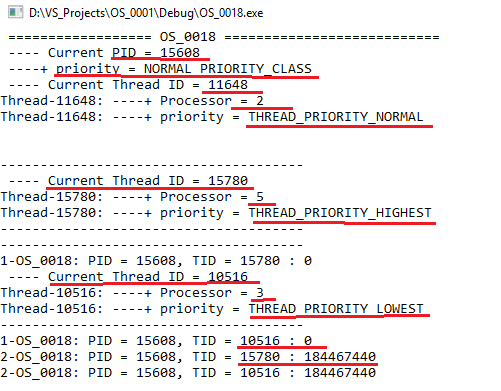






1. **OS:** Windows, CreateThread, ResumeThread, SetThreadPriority, WaitForSingleObject



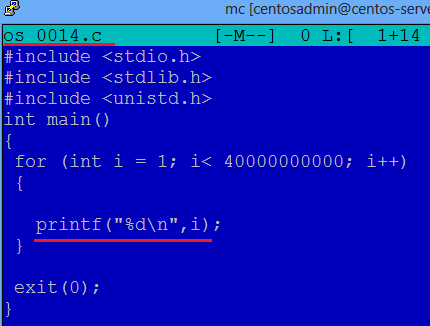


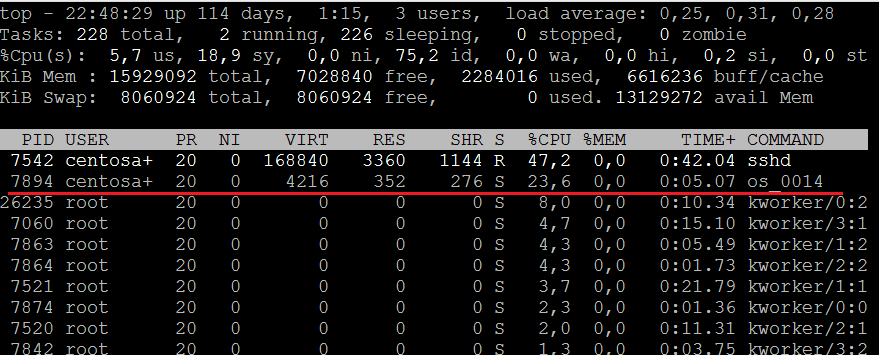
1. **OS:** Windows, только для prty = [0, 15], GetProcessPriorityBoost, SetProcessPriorityBoost, GetThreadPriorityBoost, SetThreadPriorityBoost, **TRUE – запретить boost, FALSE – разрешить boost**

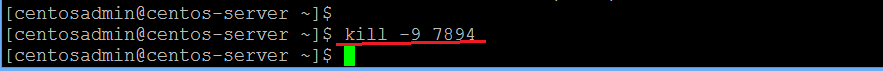


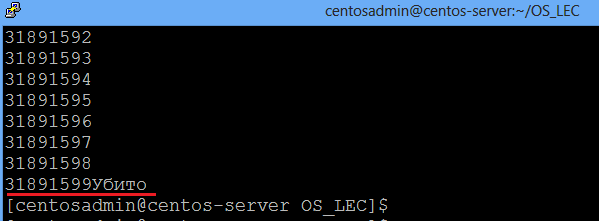


1. **OS:** Linux – мультизадачная OS с вытеснением (приоритетами).
2. **OS:** Linux, top, NI - nice, kill

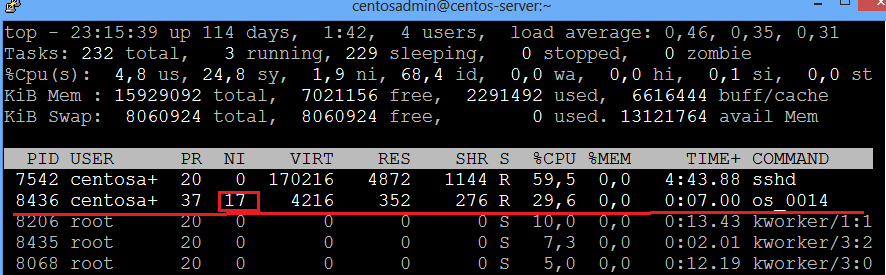




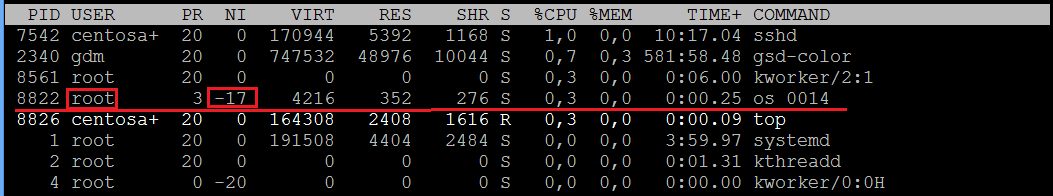






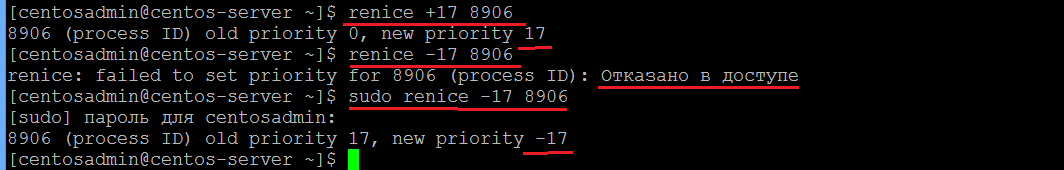


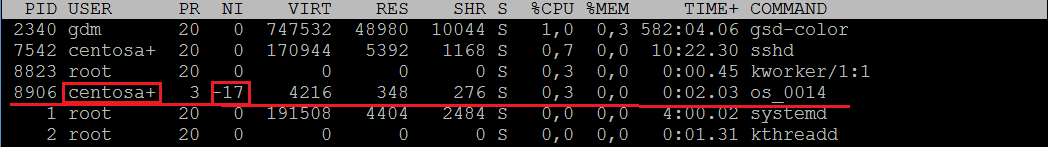




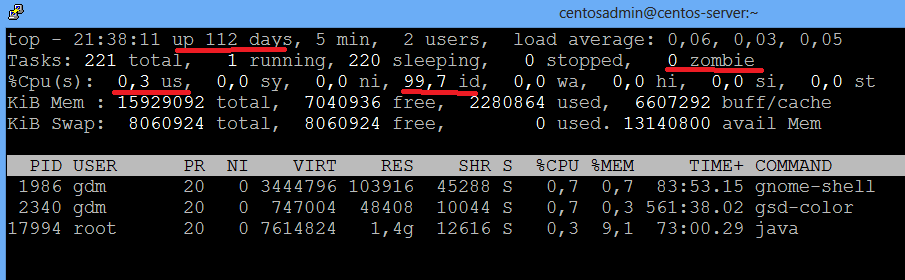


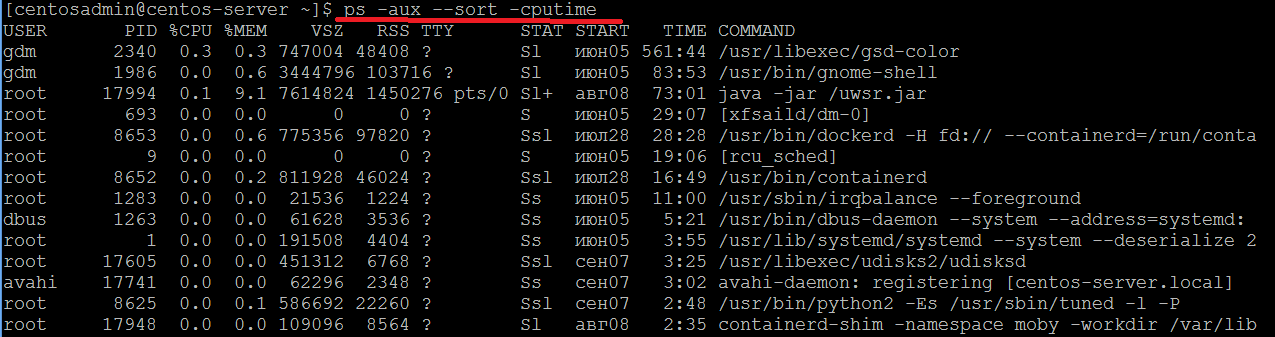
1. **OS:** Linux, top, renice



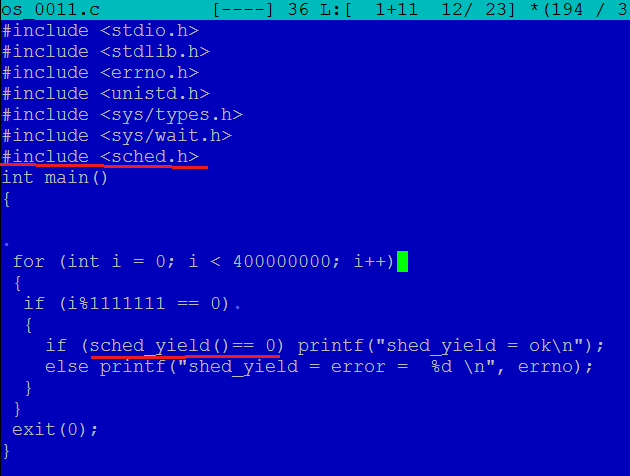


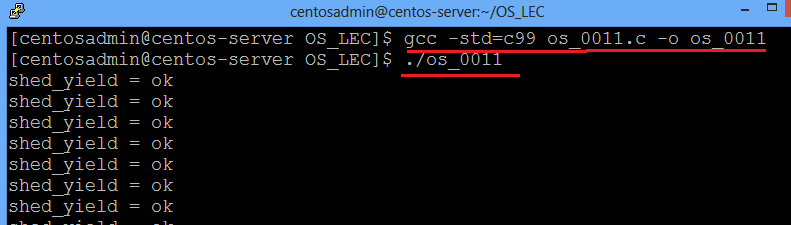
1. **OS:** Linux, потоки в Linux – единица выполнения процесса; ядро Linux не знает ничего о потоках, для ядра поток – это процесс; для ядра нет разницы между двумя изолированными процессами и двумя потоками в рамках одного процесса; два потока в одном процессе для ядра – это два процесса использующих общие ресурсы ядра.
2. **OS:** process scheduler – планировщик процессов в Linux, компонент ядра, выбирает процесс для выполнения, квантует процессорное время, ˜20мс., O(1)-планировщик.
3. **OS:** Linux/Centos 7**,** top –i, ps aus –sort -cputime





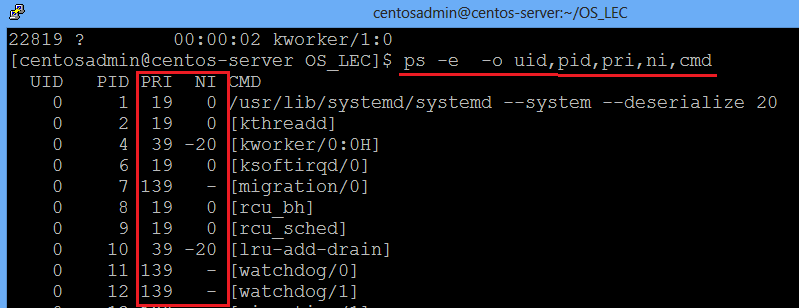
1. **OS:**Linux, уступить процессор **shed\_yield(),** эффект ping/pong(аппаратный кэш, L1, L2…)

****

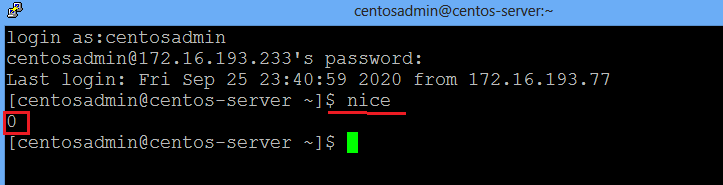


1. **OS:**Linux, значения приоритетов убывающие:

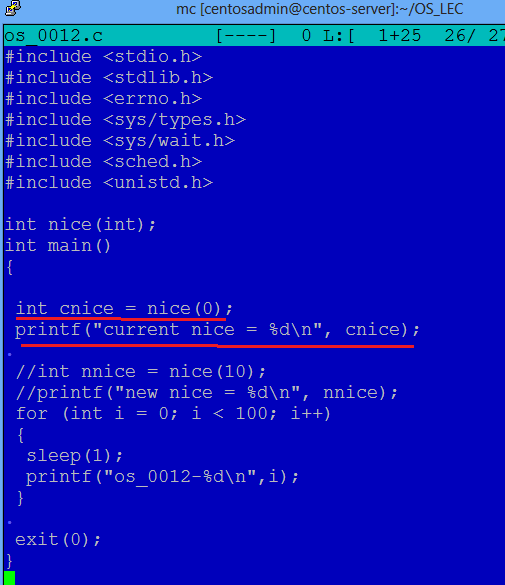
* nice – диапазон приоритетов [-20, 19], default = 0;
* real time – диапазон приоритетов, default =[0,99].

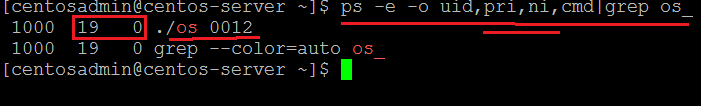


1. **OS:**Linux: nice – значение любезности, чем меньше, тем выше приоритет.
2. **OS:** Linux, nice по умолчанию

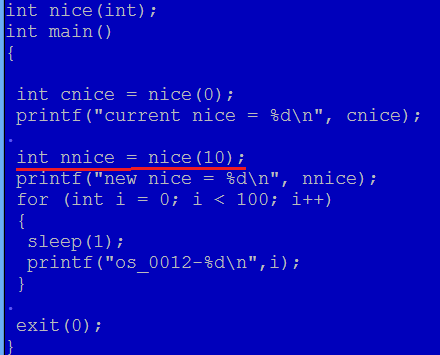


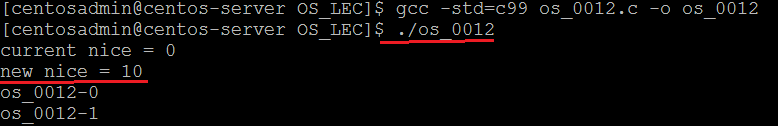
1. **OS:** Linux: функция nice

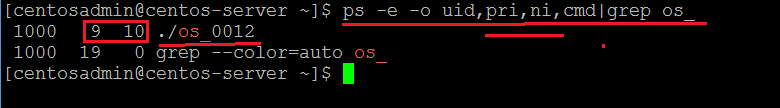


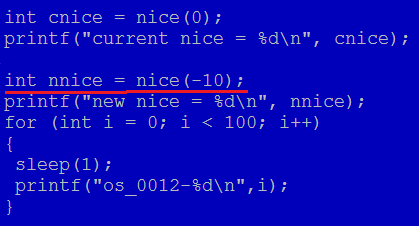


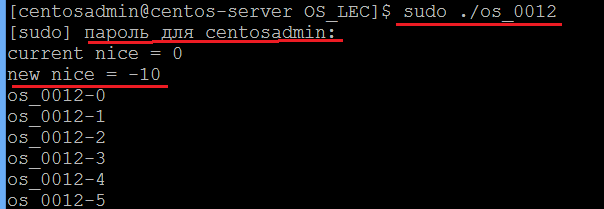
1. **OS:** Linux: nice(10), nice(-10) – необходима привилегия

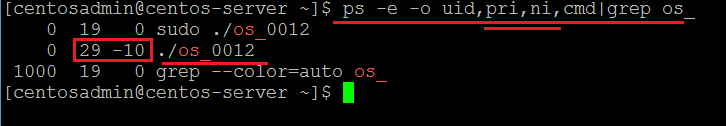












**ПРОДОЛЖЕНИЕ**

1. https://habr.com/ru/company/clrium/blog/488260/