Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Операционные системы

Студент: Песецкий Н.А.

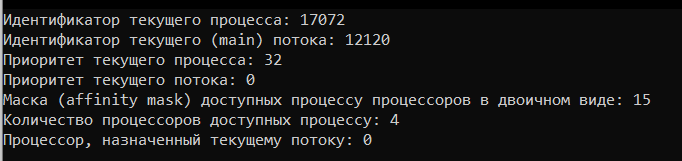
ФИТ 3 курс 5 группа

Преподаватель: Савельева М.Г.

Минск 2023

**Лабораторная работа №5**

**Задание 01**



**Задание 02**

Пункт 4

* P1: целое число, задающее маску доступности процессоров (affinity mask);
* P2: целое число, задающее класс приоритета первого дочернего процесса;
* P3: целое число, задающее класс приоритета второго дочернего процесса.

P1: 65535 - 16 потоков | 16383 - 14 | 4095 - 12 | 1023 - 10 | 255 - 8 | 63 - 6 | 15 - 4 | 3 – 2

P2\3:

64 IDLE\_PRIORITY\_CLASS;

16384 BELOW\_NORMAL\_PRIORITY\_CLASS;

32 NORMAL\_PRIORITY\_CLASS;

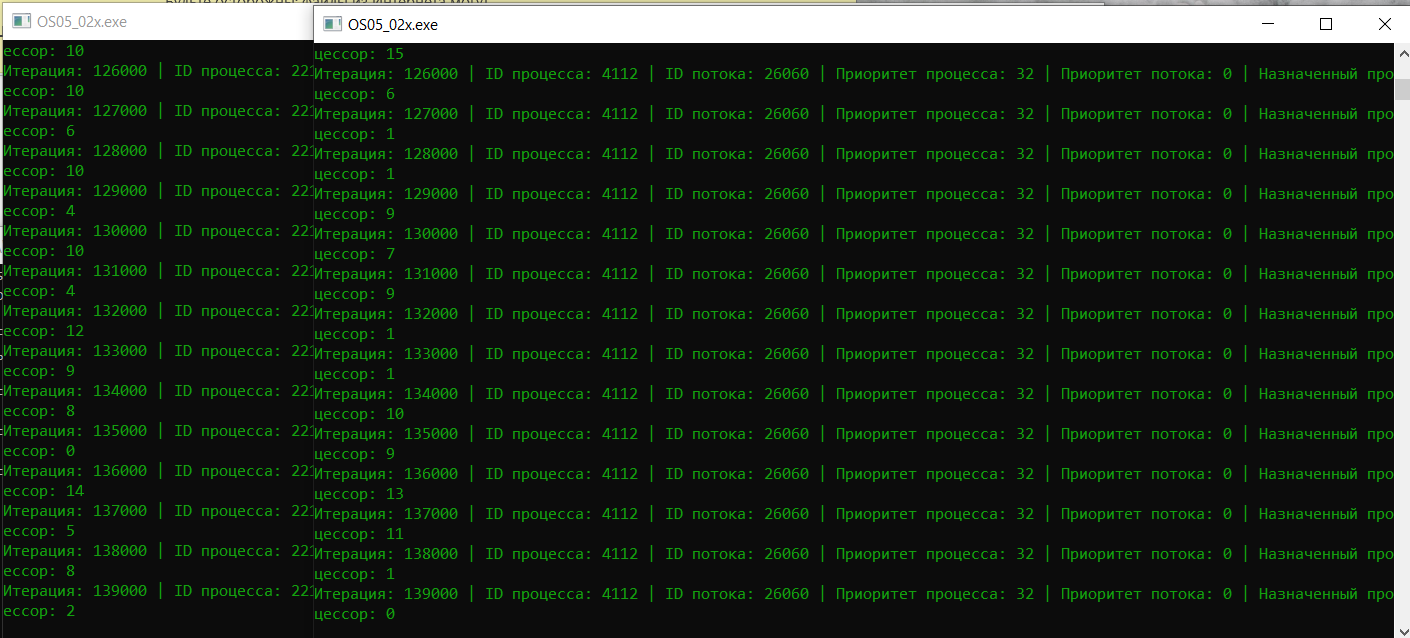
32768 ABOVE\_NORMAL\_PRIORITY\_CLASS;

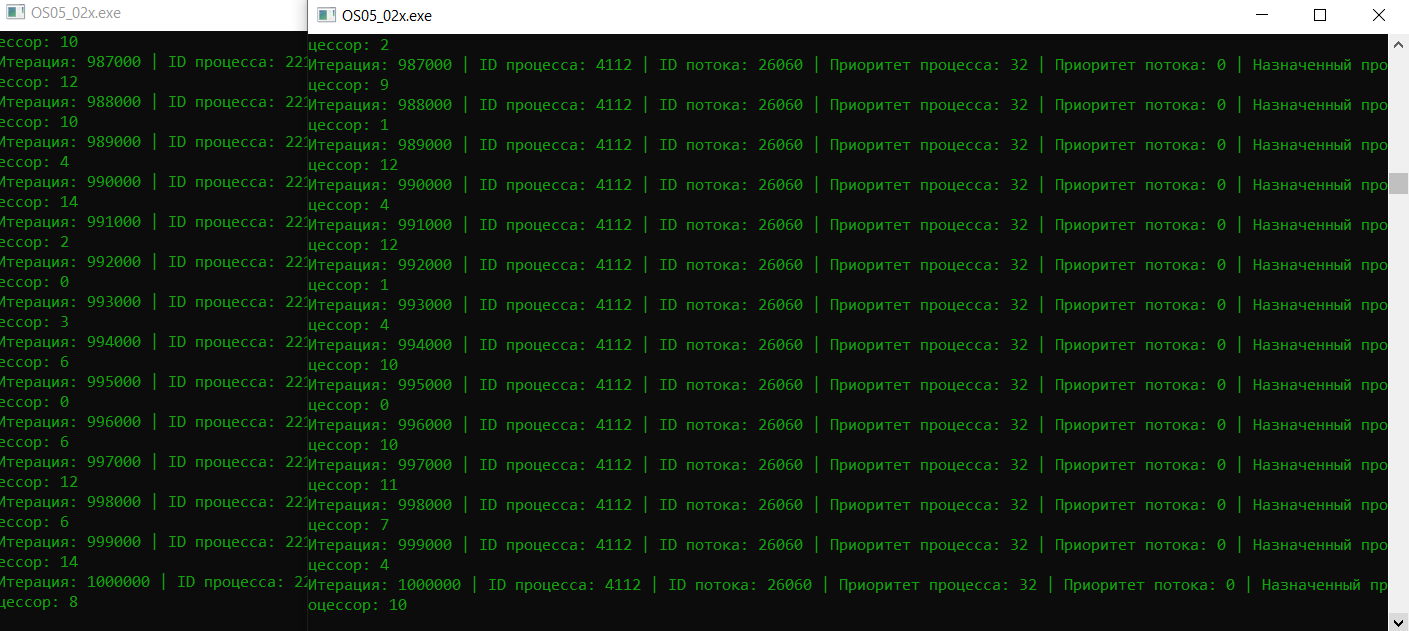
128 HIGH\_PRIORITY\_CLASS;

256 REALTIME\_PRIORITY\_CLASS;

Пункт 6

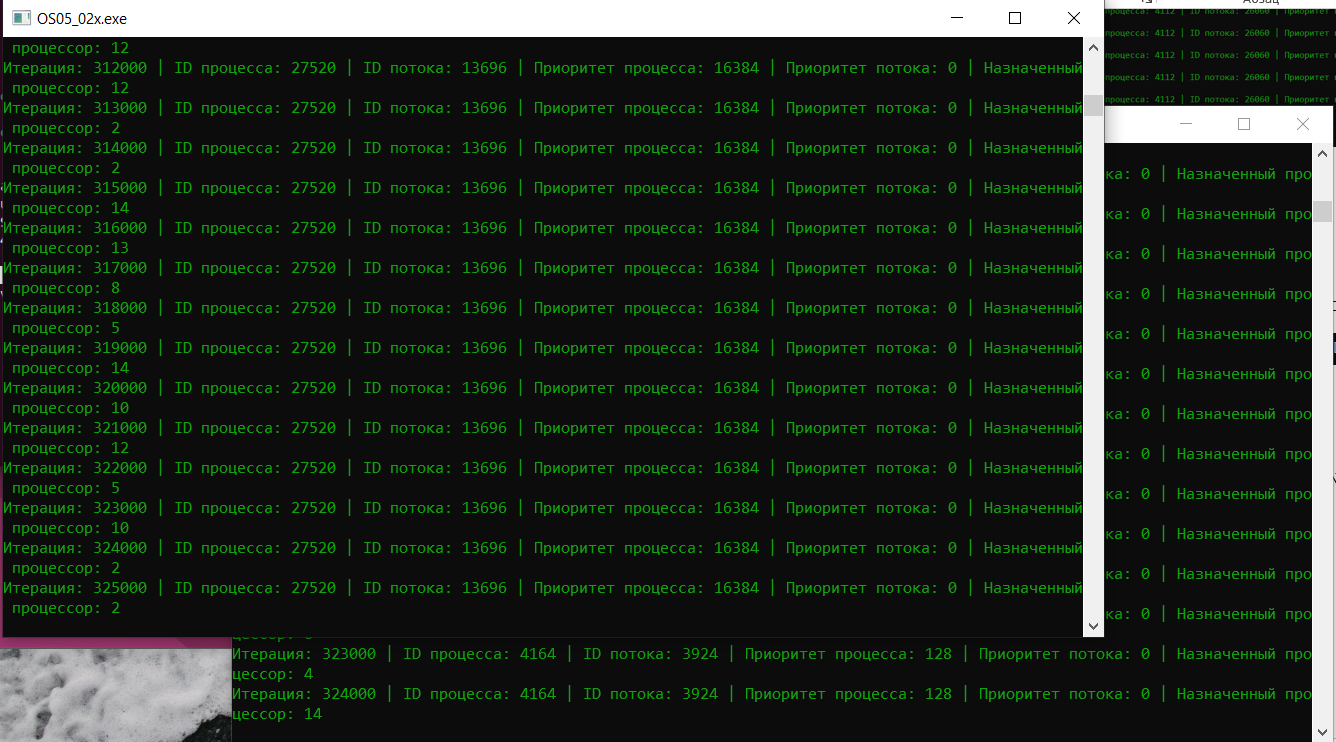
P1: доступны все процессоры P2: Normal P3: Normal





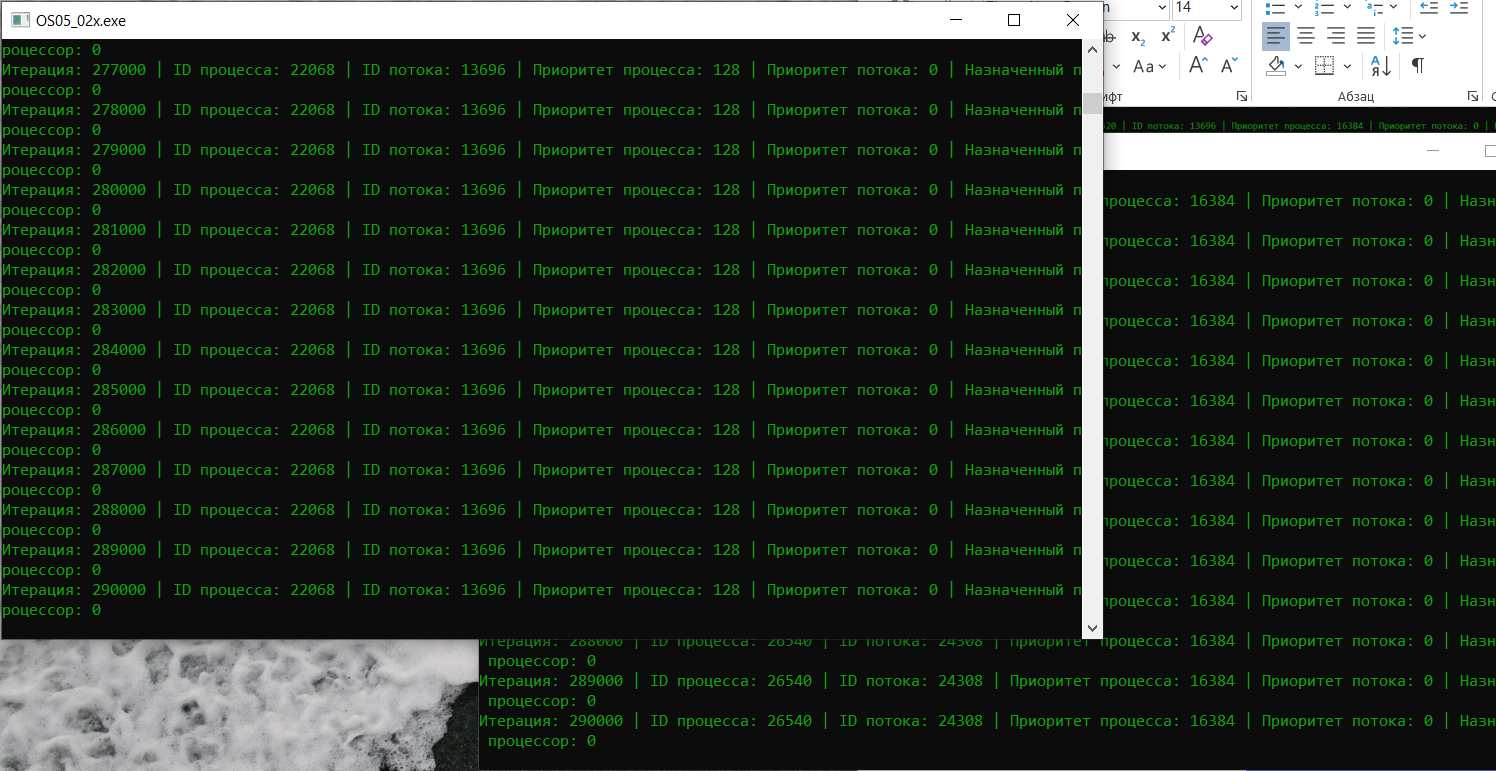
Пункт 7

P1: доступны все процессор P2: Below Normal P3: High



Пункт 8

P1: доступен один процессор P2: Below Normal P3: High



**Задание 03**

* P1: целое число, задающее маску доступности процессоров (affinity mask);
* P2: целое число, задающее класс приоритет процесса;
* P3: целое число, задающее приоритет первого дочернего потока;
* P4: целое число, задающее приоритет второго дочернего потока.

Приоритеты потоков:

THREAD\_PRIORITY\_TIME\_CRITICAL равен 15.

THREAD\_PRIORITY\_HIGHEST равен 2.

THREAD\_PRIORITY\_ABOVE\_NORMAL равен 1.

THREAD\_PRIORITY\_NORMAL равен 0.

THREAD\_PRIORITY\_BELOW\_NORMAL равен -1.

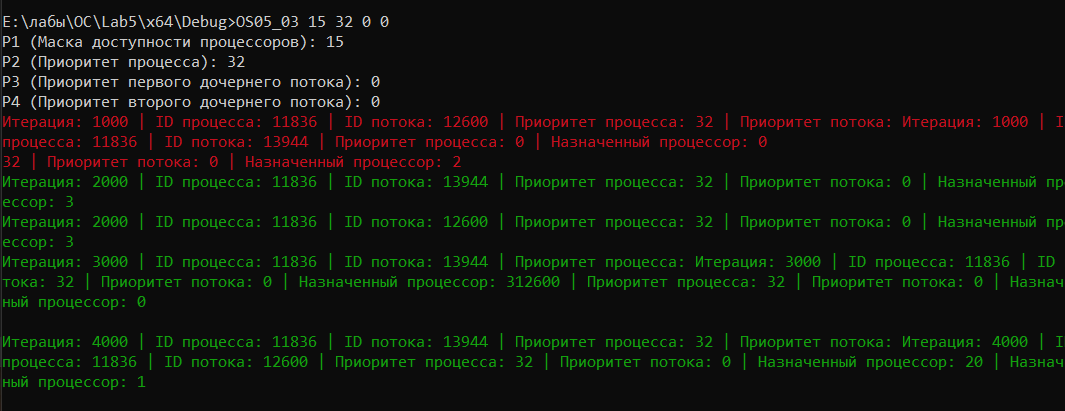
THREAD\_PRIORITY\_LOWEST равен -2.

THREAD\_PRIORITY\_IDLE равен -15.

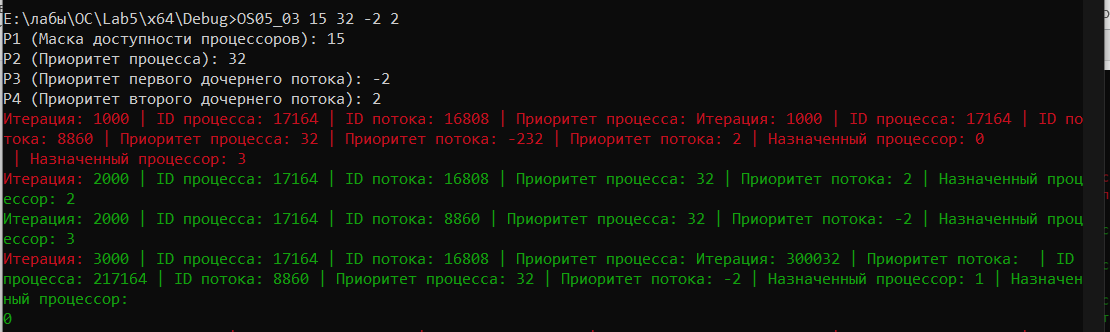
THREAD\_MODE\_BACKGROUND\_BEGIN равен 16.

THREAD\_MODE\_BACKGROUND\_END равен 15.

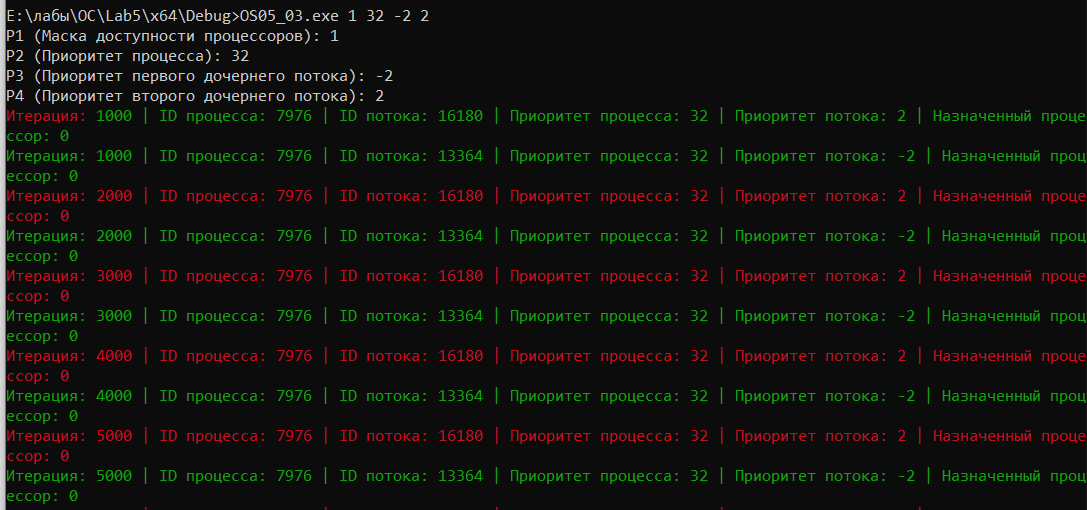
Пункт 13:



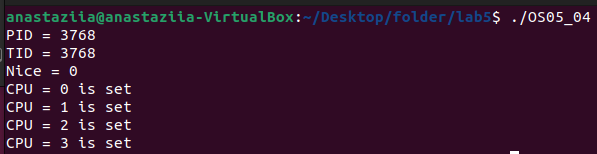
Пункт 14:



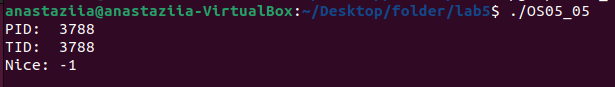
Пункт 15:



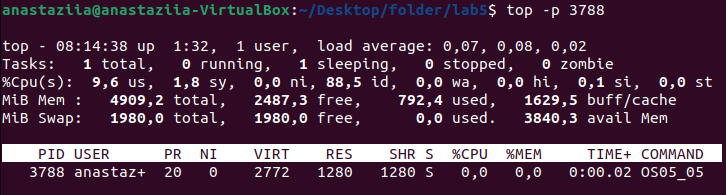
**Задание 04**

****

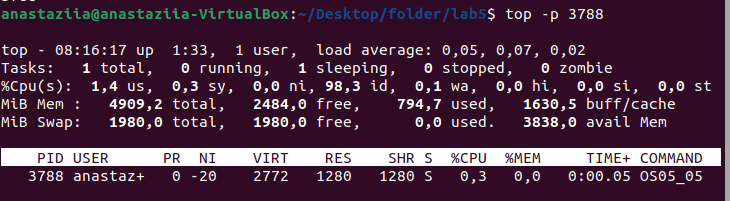
**Задание 05**

****

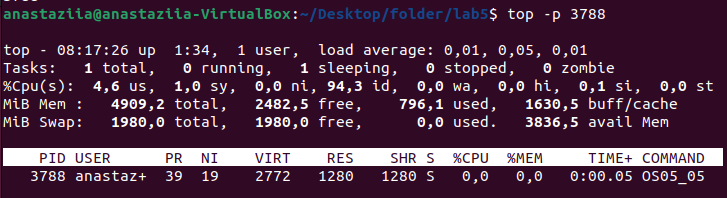
****



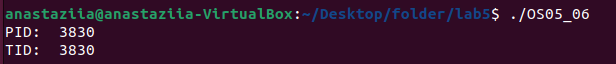








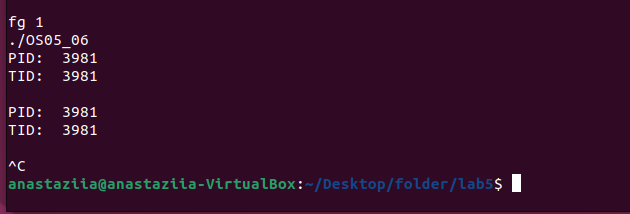
**Задание 06**

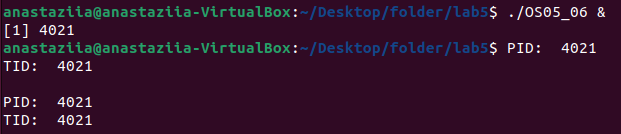
****





****

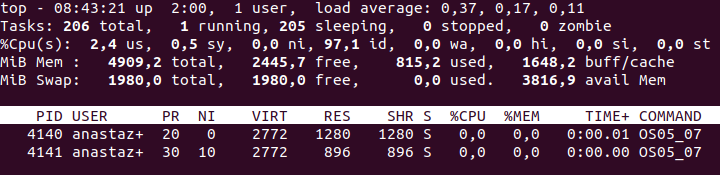
****

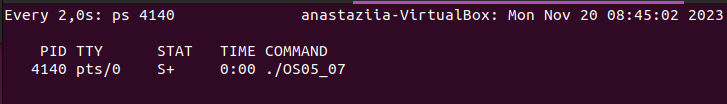
****

****

****

**Задание 07**

****

****

**Задание 08.**

1. **Поясните понятие «мультизадачная OS с вытеснением»**

**Мультизадачная OS с вытеснением** - это операционная система, которая поддерживает выполнение нескольких процессов или потоков (задач) одновременно и использует алгоритмы планирования, которые могут вытеснять текущий процесс или поток с ЦП, даже если он еще не завершил свою работу. Это обеспечивает равномерное распределение ресурсов процессора между задачами и гарантирует, что ни одна задача не блокирует выполнение других задач.

1. **Поясните понятие «циклическое планирование».**

**Циклическое планирование** - это метод планирования, при котором каждая задача или процесс получает фиксированный квант времени для выполнения на процессоре. По истечении кванта времени процесс переходит в конец очереди планирования, и следующий процесс начинает выполнение. Этот метод обеспечивает справедливое распределение процессорного времени между задачами.

1. **Поясните понятие «приоритетное планирование».**

**Приоритетное планирование** - это метод планирования, при котором каждая задача или процесс имеет приоритет, который определяет, насколько часто она получает доступ к процессору. Задачи с более высоким приоритетом будут исполняться чаще, чем задачи с более низким приоритетом.

1. **Поясните понятие «кооперативное планирование».**

**Кооперативное планирование** - это метод планирования, при котором задачи или процессы добровольно освобождают процессор, позволяя другим задачам выполняться. Это требует активного участия задачи в управлении временем на процессоре.

1. **Поясните понятие «OS реального времени».**

**OS реального времени** - это операционная система, спроектированная для выполнения задач с жесткими временными ограничениями. Она гарантирует, что задачи будут выполнены в строго определенные моменты времени, что делает ее подходящей для систем, где важна точность и надежность выполнения задач.

1. **Поясните понятие «приоритет процесса».**

**Приоритет процесса** - это числовое значение, которое определяет, насколько важен процесс для операционной системы. Процессы с более высоким приоритетом получают больше процессорного времени и выполняются чаще.

1. **Поясните выражение «поток уступает процессор другому потоку».**

**Поток уступает процессор другому потоку** - это означает, что поток, который в данный момент выполняется на процессоре, добровольно переходит в состояние ожидания, чтобы другой поток мог получить доступ к процессору. Это осуществляется с целью справедливого распределения процессорного времени между потоками.

1. **Windows: как поток может уступить процессор?**

**Windows: как поток может уступить процессор**? - В Windows поток может уступить процессор, вызвав функцию Sleep(0) или SwitchToThread(), которые позволяют другим потокам выполниться.

1. **Windows: что такое базовый приоритет потока, как он вычисляется и диапазон его изменения?**

**Windows: что такое базовый приоритет потока, как он вычисляется и диапазон его изменения?** - Базовый приоритет потока - это числовое значение, которое определяет, насколько важен поток для операционной системы. В Windows базовый приоритет потока находится в диапазоне от 1 (самый низкий приоритет) до 15 (самый высокий приоритет). Он может быть установлен с помощью функции SetThreadPriority и рассчитывается на основе приоритета процесса и других факторов.

1. **Windows: поясните назначение и принцип применения системного вызова SetThreadIdealProcessor.**

**Windows: поясните назначение и принцип применения системного вызова SetThreadIdealProcessor** - Функция SetThreadIdealProcessor в Windows используется для установки предпочитаемого процессора для потока. Это позволяет программе указать, на каком процессоре должен выполняться поток. Это может быть полезно для оптимизации производительности в многопроцессорных системах.

1. **Windows: поясните назначение и принцип применения системного вызова ResumeThread.**

**Windows: поясните назначение и принцип применения системного вызова ResumeThread -** Функция ResumeThread используется для возобновления выполнения потока, который был приостановлен с помощью SuspendThread. Она позволяет продолжить выполнение приостановленного потока.

1. **Windows: поясните назначение и принцип применения системного вызова WaitForSingleObject.**

**Windows: поясните назначение и принцип применения системного вызова WaitForSingleObject** - Функция WaitForSingleObject используется для ожидания завершения выполнения объекта, такого как поток, процесс, событие или мьютекс. Она блокирует текущий поток до тех пор, пока объект не завершит свою работу.

1. **Windows: поясните назначение и принцип применения системных вызовов GetProcessPriorityBoost, GetThreadPriorityBoost, SetProcessPriorityBoost, SetThreadPriorityBoost.**

**Windows: поясните назначение и принцип применения системных вызовов GetProcessPriorityBoost, GetThreadPriorityBoost, SetProcessPriorityBoost, SetThreadPriorityBoost** - Эти системные вызовы используются для управления приоритетом и "приоритетным ускорением" процессов и потоков в Windows. Они позволяют включать или выключать функцию "приоритетного ускорения", которая может повышать приоритет задач в определенных случаях.

В Windows приоритетное ускорение (Priority Boost) - это механизм, который используется операционной системой Windows для временного увеличения приоритета выполнения процесса или потока.

1. **Linux: поясните принцип идентификации процессов и потоков и поясните почему он такой.**

**Linux: поясните принцип идентификации процессов и потоков и поясните почему он такой -** В Linux идентификация процессов и потоков выполняется с использованием идентификаторов процессов (PID) и идентификаторов потоков (TID). Каждый процесс имеет уникальный PID, и каждый поток внутри процесса имеет уникальный TID, но PID и TID могут совпадать в разных процессах. Это позволяет операционной системе однозначно идентифицировать процессы и потоки.

1. **Linux: Поясните понятие «планировщик потоков».**

**Linux: Поясните понятие "планировщик потоков"** - Планировщик потоков (или планировщик задач) в Linux - это часть операционной системы, отвечающая за распределение процессорного времени между процессами и потоками. Он решает, какие процессы или потоки будут выполняться, и в каком порядке. В Linux существует несколько политик планирования, таких как SCHED\_FIFO, SCHED\_RR и SCHED\_OTHER, которые определяют алгоритмы планирования и приоритеты.

1. **Linux: поясните принцип использования значения nice –процесса, диапазон его изменения, для какого режима работы планировщика это значение применяется?**

**Linux: поясните принцип использования значения nice – процесса, диапазон его изменения, для какого режима работы планировщика это значение применяется** - Значение nice в Linux используется для определения относительного приоритета процесса. Значение nice находится в диапазоне от -20 (наивысший приоритет) до +19 (наименьший приоритет). Значение nice положительное, когда приоритет низкий, и отрицательное, когда приоритет высокий. Значение nice применяется в планировщике потоков SCHED\_OTHER в Linux.

1. **Linux: перечислите политики планирования, какая действует по умолчанию?**

**Linux: перечислите политики планирования, какая действует по умолчанию? -** В Linux существует несколько политик планирования, включая:

**SCHED\_OTHER** (по умолчанию) - политика для обычных процессов

**SCHED\_FIFO** - политика с жесткими временными ограничениями

**SCHED\_RR** - политика с жесткими временными ограничениями и квази-раунд-робин планированием

**SCHED\_BATCH** - политика для процессов, выполняющих задачи с низким приоритетом

**SCHED\_IDLE** - политика с наименьшим приоритетом

1. **Linux: как выяснить действующую политику планирования для процесса с помощью файловой системы proc?**

**Linux: как выяснить действующую политику планирования для процесса с помощью файловой системы proc?** - Вы можете узнать текущую политику планирования для процесса, просмотрев файл cat **/proc/[PID]/sched** или **/proc/[PID]/status**, где **[PID]** - это идентификатор процесса.

1. **Linux: с помощью какого системного вызова поток может уступить процессор.**

**Linux: с помощью какого системного вызова поток может уступить процессор** - Поток может уступить процессор с помощью системного вызова sched\_yield().

1. **Linux: чем отличается системный вызов nice от вызова setpriority**

**Linux: чем отличается системный вызов nice от вызова setpriority** - Системный вызов nice используется для изменения приоритета процесса и принимает значение от -20 (высший приоритет) до +19 (низший приоритет).

nice работает с использованием относительных приоритетов и принимает целое число, которое указывает, насколько сместить текущий приоритет процесса.

setpriority позволяет устанавливать приоритеты в более широком диапазоне, чем nice. Он использует абсолютные значения, где -20 всегда является самым высоким приоритетом, а 19 - самым низким.

1. **Linux: поясните понятие «планировщик ввода вывода», каким образом можно выяснить какие планировщики ввода/ вывода доступны?**

Планировщик ввода-вывода (I/O scheduler) в операционной системе Linux является частью ядра (kernel), ответственной за управление операциями ввода и вывода на диске и других устройствах хранения данных.

Просмотр доступных планировщиков:

cat /sys/block/sda/queue/scheduler

1. **Linux: перечислите известные вам планировщики ввода/ вывода, кратко охарактеризуйте их.**

**Linux: перечислите известные вам планировщики ввода/вывода, кратко охарактеризуйте их** -

**CFQ** (Completely Fair Queuing) - обеспечивает справедливое распределение ресурсов диска между процессами.

**Deadline** - ставит приоритет на соблюдение временных ограничений для операций ввода/вывода.

**NOOP** - простой планировщик без приоритетов, хорош для ненагруженных систем.

1. **Linux: каким образом можно выяснить тип планировщика действующего для блокового устройства?**

cat /sys/block/sda/queue/scheduler