Оглавление

[**1. История, назначение, функции и виды операционных систем.** 1](#_Toc138339450)

[**2. Структура операционных систем. Виды ядра операционных систем.** 3](#_Toc138339451)

[**3. Микроядерная архитектура (модель клиент-сервер).** 4](#_Toc138339452)

[**4. Использование сервисных программ поддержки интерфейсов. Настройка рабочего стола.** 4](#_Toc138339453)

[**5. Настройка системы с помощью панели управления.** 5](#_Toc138339454)

[**6. Применение потоков. Классификация потоков. Реализация потоков.** 5](#_Toc138339455)

[**7. Модель процесса. Создание процесса. Завершение процесса. Иерархия процесса.** 6](#_Toc138339456)

[**8. Управление процессами с помощью команд операционной системы для работы с процессами.** 7](#_Toc138339457)

[**9. Соотношения между представляемым и истинным объёмом занятой дисковой памяти.** 13](#_Toc138339458)

[**10. Изучение влияния количества файлов на время, необходимое для их копирования.** 14](#_Toc138339459)

[**11. Управление процессами с помощью команд операционной системы для работы с процессами.** 14](#_Toc138339460)

[**12. Работа с программой «Файл-менеджер Проводник».** 20](#_Toc138339461)

[**13. Диагностика и коррекция ошибок операционной системы, контроль доступа к операционной системе.** 20](#_Toc138339462)

[**14. Работа с файловыми системами и дисками.** 22](#_Toc138339463)

[**15. Установка новых устройств.** 25](#_Toc138339464)

[**16. Использование команд работы с файлами и каталогами.** 28](#_Toc138339465)

[**17. Файловая система и ввод и вывод информации.** 29](#_Toc138339466)

[**18. Конфигурация файлов.** 31](#_Toc138339467)

[**19. Работа с операционной оболочкой.** 32](#_Toc138339468)

[**20. Управление процессами в операционной системе.** 33](#_Toc138339469)

[**21 Резервное хранение, командные файлы.** 33](#_Toc138339470)

[**22. Планирование и установка операционной системы Windows.** 34](#_Toc138339471)

[**23. Работа с текстовым редактором. Работа с Архиватором.** 36](#_Toc138339472)

## **1. История, назначение, функции и виды операционных систем.**

**История операционных систем:**

1. *Первые поколения (1940-1950)*: В это время компьютеры были огромными машинами, работающими на лампах и перфокартах. Они не имели ОС, и программы запускались прямо на "железе".
2. *Пакетная обработка (1950-1960)*: В этот период были созданы первые операционные системы, которые позволяли выполнять несколько программ пакетно. Например, UNIVAC I и IBM 704 были оборудованы операционными системами для пакетной обработки.
3. *Параллельная и разделенная обработка (1960-1970)*: В это время появились операционные системы, которые позволяли выполнять несколько программ одновременно. Некоторые известные ОС этого периода: IBM OS/360, MULTICS.
4. *Персональные компьютеры (1980-1990)*: С появлением персональных компьютеров операционные системы стали более доступными для широкой аудитории. MS-DOS и MacOS были двумя популярными ОС в этот период.
5. *Графический интерфейс пользователя (1990-2000)*: Это был период, когда графические интерфейсы пользователя (GUI) стали стандартом для операционных систем. ОС, такие как Microsoft Windows и MacOS, предоставляли пользовательский интерфейс, основанный на иконках, окнах и указателе мыши.
6. *Сетевая эра (2000-настоящее время)*: С развитием интернета и сетевых технологий операционные системы стали более ориентированными на сетевую работу. Линукс и Windows Server стали широко используемыми ОС для серверов, а Android и iOS - популярными операционными системами для мобильных устройств.

В настоящее время существует множество операционных систем, включая Windows, macOS, Linux, Android, iOS и многие другие.

**Назначение операционных систем:**

**Операционная система** – это программное обеспечение, устанавливаемое на компьютер или другое устройство, для обеспечения его работоспособности и управления ресурсами. Главное назначение ОС состоит в обеспечении взаимодействия между аппаратными компонентами компьютера и прикладным программным обеспечением.

**Функции операционных систем:**

1. *Управление ресурсами*: ОС управляет доступом к аппаратным и программным ресурсам компьютера, таким как процессор, память, жесткий диск и периферийные устройства.
2. *Планирование задач*: ОС распределяет процессорное время между запущенными задачами, определяет приоритеты и управляет выполнением процессов.
3. *Управление памятью*: ОС отвечает за управление физической и виртуальной памятью компьютера, включая выделение и освобождение памяти для программ.
4. *Управление файловой системой*: ОС обеспечивает организацию и управление файлами и каталогами на жестком диске или других носителях информации.
5. *Обеспечение безопасности*: ОС контролирует доступ к системным ресурсам и обеспечивает безопасность данных и конфиденциальность.
6. Поддержка пользовательского интерфейса: ОС предоставляет пользовательский интерфейс для взаимодействия с компьютером, например, графический интерфейс или командную строку.

**Виды операционных систем:**

1. **Пакетные системы:** Позволяют группировать задачи в пакеты и выполнять их последовательно. Характерны для больших вычислительных центров.
2. **Параллельные системы:** Позволяют параллельное выполнение нескольких задач одновременно на нескольких процессорах.
3. **Распределенные системы:** Операционные системы, работающие на нескольких компьютерах, связанных сетью, и позволяющие эффективно использовать ресурсы и обмениваться информацией.
4. **Временные системы реального времени:** Операционные системы, где выполнение задач должно быть завершено в строго заданные сроки, часто используемые в авионике, медицине и других приложениях, где требуется быстрая реакция.
5. **Операционные системы для мобильных устройств:** ОС, разработанные специально для мобильных устройств, таких как смартфоны и планшеты, с учетом ограниченных ресурсов и особенностей мобильной связи.
6. **Встроенные операционные системы**: ОС, предназначенные для использования во встроенных системах, таких как бытовая электроника, автомобили, роботы и другие устройства, где ОС интегрирована непосредственно с аппаратурой.

## **2. Структура операционных систем. Виды ядра операционных систем.**

***Структура ОС*** содержит следующие компоненты:

1. ***Ядро***– центральная часть операционной системы, обеспечивающая приложениям координированный доступ к ресурсам компьютера. Так же это наиболее часто используемые модули ОС, например:

* модуль управления системой прерываний;
* средства распределения оперативной памяти;
* средства распределения ресурсов процессора,

1. ***Резидентные программы*** – это программы, входящие в состав ядра, например, программы-драйверы, управляющие внешними устройствами (когда ПК работает они постоянно находятся в оперативной памяти),
2. ***Командный процессор***–программа, отвечающая за исполнение простейших команд, подаваемых пользователем, а также за взаимодействие этих команд с ядром ОС,
3. ***Система управления файлами***–программа дляорганизации удобного доступа к файлам.Для каждой ОС разрабатывается своя файловая система.

***Существует 4 основных типа ядер***:

**Монолитное ядро** – эта схема ОС, когда все части её ядра — это составные части одной программы (самый старый способ организации ОС).

Достоинства: высокая скорость работы, простая разработка модулей.  
Недостатки: ошибка работы одного из компонентов ядра нарушает работу всей системы.

**Модульное ядро** – схема ОС, когда все части её ядра распределены по сегментам.

Достоинства: выход из строя какой-то компонента не «ломает» работу всего ядра.  
Недостатки: ядро медленнее работает.

**Гибридное ядро** – сочетает в себе особенности монолитного и модульного ядра, а соответственно их плюсы и минусы.

## **3. Микроядерная архитектура (модель клиент-сервер).**

**Микроядро** — это когда ядро отвечает только за основные функции: *управление процессором*, *оперативной памятью*, *прерываниями и операциями ввода-вывода*. Всё остальное (драйверы, файловая система, управление оборудованием) выносится в так называемое пользовательское пространство и запускается в виде обычных процессов. Благодаря экому ядро, в отличие от монолитных систем, остаётся сравнительно компактным.

Суть этой архитектуры состоит в следующем. В привилегированном режиме остается работать только очень небольшая часть ОС, называемая микроядром. Микроядро защищено от остальных частей ОС и приложений. В его состав входят машинно-зависимые модули, а также модули, выполняющие базовые механизмы обычного ядра. Все остальные более высокоуровневые функции ядра оформляются как модули, работающие в пользовательском режиме. Так, менеджеры ресурсов, являющиеся неотъемлемой частью обычного ядра, становятся «периферийными» модулями, работающими в пользовательском режиме.

## **4. Использование сервисных программ поддержки интерфейсов. Настройка рабочего стола.**

**Сервисные программы поддержки интерфейсов** облегчают настройку, управление и обслуживание интерфейсов операционной системы. Они позволяют пользователям оптимизировать работу с устройствами и программами, улучшить качество связи и сократить время на решение проблем.

Сначала необходимо определить, какую сервисную программу следует использовать для своих задач. В Windows, например, можно использовать диспетчер устройств, где можно посмотреть список устройств, зарегистрированных в системе, и проверить их состояние.

После выбора сервисной программы следует настроить рабочий стол таким образом, чтобы получить из него максимальную отдачу. Необходимо оставить только те программы, которые реально используются для работы, и для каждой из них создать ярлык на рабочем столе. Это позволяет быстро запускать нужные программы, не тратя время на поиск их в меню Пуск.

**Также можно настроить рабочий стол**, чтобы он был более удобен для работы. Например, можно изменить размер иконок, создать отдельные папки для определенных видов файлов и настроить фоновое изображение. В результате рабочий стол будет более удобен и производительный.

В итоге, использование сервисных программ поддержки интерфейсов и настройка рабочего стола позволяют повысить производительность и удобство работы с компьютером.

## **5. Настройка системы с помощью панели управления.**

Панель Управления является частью пользовательского интерфейса Microsoft Windows. Она позволяет выполнять основные действия по настройке системы, такие, как добавление и настройка устройств, установка и деинсталляция программ, управление учётными записями, включение специальных возможностей, а также многие другие действия, связанные с управлением системой.

Есть 2 способа открыть панель управления:

1) Нажать комбинацию клавиш WIN+R, в появившемся окне напечатать control и нажать Enter.

2) В поисковике Windows написать Панель Управления и кликнуть по появившейся иконке.

## **6. Применение потоков. Классификация потоков. Реализация потоков.**

**Применение потоков:**

В многопоточной среде программы могут быть разделены на части, называемые потоками выполнения, которые выполняются одновременно.

Поток — это «каналы» для одновременного исполнения команд. Чем больше потоков, тем больше команд может одновременно обработать процессор, а значит, многопоточная программа будет работать быстрее.

Соответственно ***потоки применяют для увеличения производительности системы***

**Классификация потоков:**

**ПО СОСТАВУ** потоки делятся на:

Простые:

* Материальные (текущего периода, длительного периода)
* Энергетические (технологии, человека)
* Информационные (натуральные, стоимостные)

Сложные:

* Потоки рабочей силы (материальное тело, физическая энергия)

**Виды потоков:**материальные, информационные, финансовые, сервисные.

Материальный – находящееся в состоянии движения материальные ресурсы, к которым применяется логистические операции, связанные с физическим перемещением в пространстве и времени от поставщика ресурсов до конечного потребителя.

Сервисный – поток услуг обеспечивающий высокий уровень сервисного обслуживания материальных потоков.

Информационный – поток сообщений в речевой, бумажной, магнитной форме, генерированный исходной бумажной форме рассматривается в логистической системе и предназначен для реализации управленческих функций.

Финансовый – направленное движение финансовых средств, циркулирующих в логистической системе и между ней и внешней средой необходимой для движения данного материального потока.

**Реализация потоков:**

Есть 3 основных способа реализации потоков:

1) В пространстве пользователя

2) В ядре

3) Смешанная реализация

## **7. Модель процесса. Создание процесса. Завершение процесса. Иерархия процесса.**

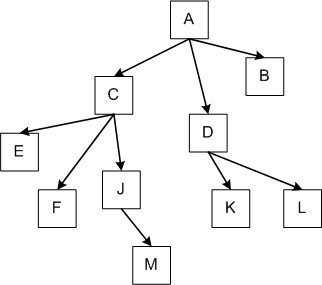
**Модель процесса**

В этой модели все функционирующее на компьютере программное обеспечение, иногда включая собственно операционную систему, организовано в виде набора последовательных процессов, или, для краткости, просто процессов. Процессом является выполняемая программа, включая текущие значения счетчика команд, регистров и переменных. С позиций данной абстрактной модели, у каждого процес­са есть собственный виртуальный центральный процессор  
**Создание процесса**

4 основ­ных события, приводящие к созданию процессов:  
\* Инициализация системы.  
\* Выполнение изданного работающим процессом системного запроса на созда­ние процесса.  
\* Запрос пользователя на создание процесса.  
\* Инициирование пакетного задания  
**Завершение процесса**

После того как процесс создан, он начинает выполнять свою работу. В конце концов он завершается одним из следующих событий:  
\* Обычный выход (преднамеренно).  
\* Выход по ошибке (преднамеренно).  
\* Выход по неисправимой ошибке (непреднамеренно).  
\* Уничтожение другим процессом (непреднамеренно).  
**Иерархия процесса**

В Windows не существует понятия иерархии процессов. Хотя можно задать специальный маркер родительскому процессу, позволяющий контролировать дочерний процесс.

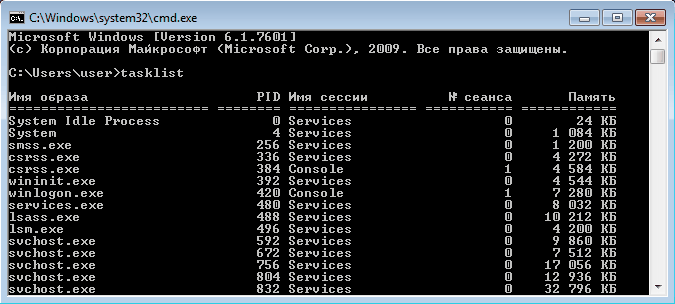
В UNIX системах заложена жесткая иерархия процессов. Каждый новый процесс, созданный системным вызовом fork, является дочерним к предыдущему процессу. Дочернему процессу достаются от родительского переменные, регистры и т.п. После вызова fork, как только родительские данные скопированы, последующие изменения в одном из процессов не влияют на другой, но процессы помнят о том, кто является родительским.

## **8. Управление процессами с помощью команд операционной системы для работы с процессами.**

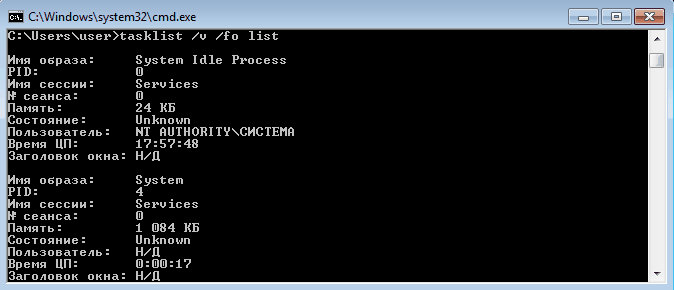
**Управление процессами из командной строки**

Для управления процессами в командной строке есть две утилиты — **tasklist** и **taskkill**. Первая показывает список процессов на локальном или удаленном компьютере, вторая позволяет их завершить. Попробуем …

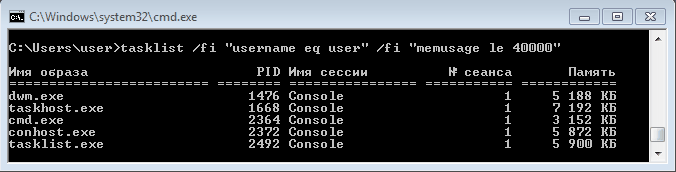
Если просто набрать команду **tasklist** в командной строке, то она выдаст список процессов на локальном компьютере.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd1.gif)

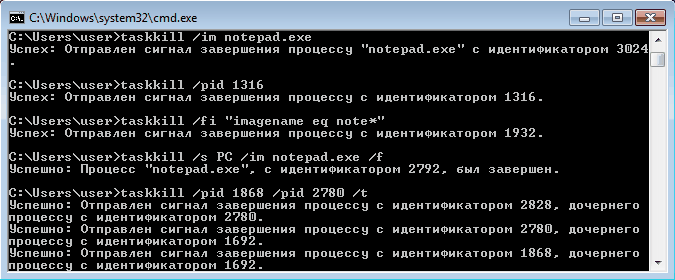
По умолчанию информация выводится в виде таблицы, однако ключ **/fo** позволяет задать вывод в виде списка или в формате CSV, а ключ **/v**показывает более подробную информацию о процессах, например команда **tasklist /v /fo list** выведет подробное описание всех процессов в виде списка.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd2.gif)

Список получится довольно большой, поэтому попробуем уточнить запрос.  Для этого используем ключ **/fi** , который позволяет использовать фильтры для вывода данных, например команда **tasklist /fi ″username eq user″ /fi ″memusage le 40000″** выводит список процессов пользователя **user**, которые потребляют не больше 40Мб памяти.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd3.gif)

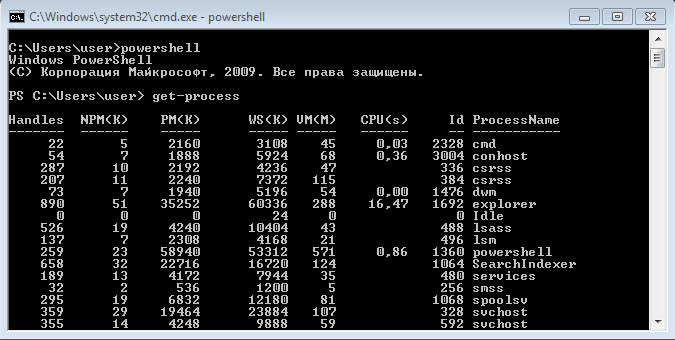
Найдя процессы, которые необходимо завершить, воспользуемся командой **taskkill.**Завершать процессы можно по имени, идентификатору процесса (PID) или задав условия с помощью фильтров. Для примера запустим несколько экземпляров блокнота (notepad.exe) и попробуем завершить его разными способами.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd5.gif)

Ключ **/f** завершает процесс принудительно, а **/t** завершает все дочерние процессы.

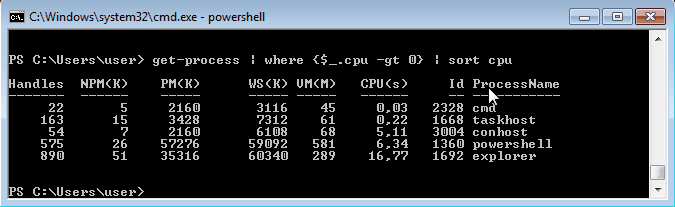
Полную справку по командам tasklist и taskkill можно получить, введя их с ключом**/?**

Теперь пустим в ход тяжелую артиллерию*—*PowerShell*.*Его можно запустить не выходя из командной строки. Для получения списка процессов используем командлет **Get-Process.**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd6.gif)

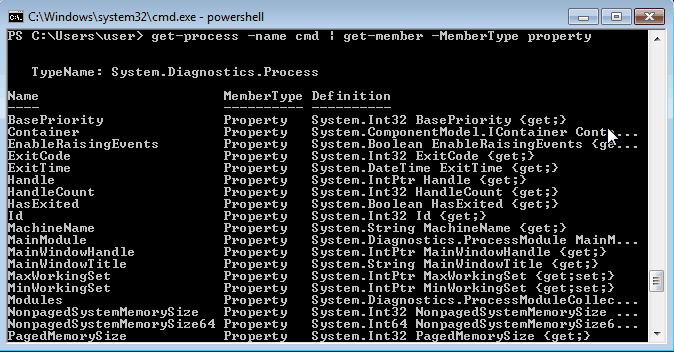
Чтобы не выводить весь список процессов можем воспользоваться командлетом **Where-Object***,*который задает фильтр для выводимой информации. Для примера выведем список процессов, которые загружают процессор и отсортируем их по возрастанию нагрузки с помощью команды:

**Get-Process | where {$\_.cpu -gt 0} | sort cpu**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd9.gif)

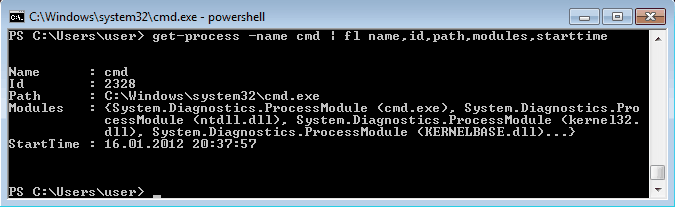
С помощью PowerShell мы можем получить любую информацию о любом процессе. В качестве примера возьмем процесс **cmd**и выведем список его свойств командой:

**Get-Process -Name cmd | Get-Member -Membertype property**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd7.gif)

Выбираем те свойства, что нам интересны ( в примере имя и ID процесса, путь к файлу, используемые модули и время запуска) и выводим их в виде списка командой:

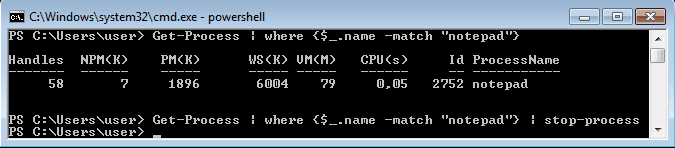
**Get-Process -Name cmd | Format-List name, id, path, modules, starttime**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd8.gif)

Таким образом мы можем посмотреть когда и кем был запущен процесс, сколько он потребляет ресурсов, где находится исполняемый файл и еще много различной информации.

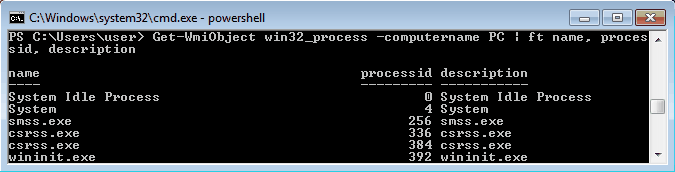
Для завершения процесса в PowerShell есть командлет **Stop-Process**. Он завершает указанный процесс по его имени или идентификатору. Однако мы поступим по другому и передадим результат выполнения командлета **Get-Process**по конвейеру:

**Get-Process | where {$\_.name -match ″notepad″}  | Stop-Process**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd10.gif)

**Get-Process** не может показать процессы на удаленном компьютере, для этого воспользуемся командлетом **Get-WmiObject** , например посмотрим процессы на удаленном компьютере PC командой:

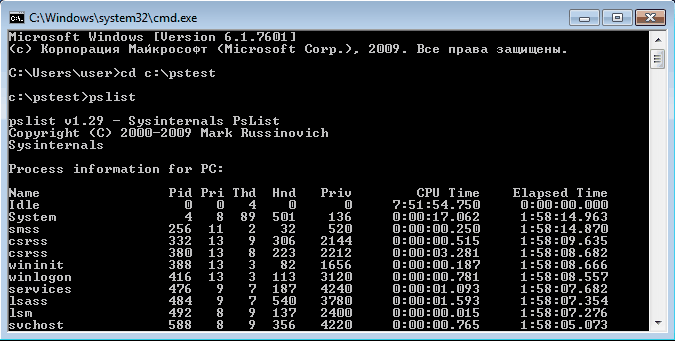
**Get-WmiObject win32\_process -computername PC | ft name, processid, description**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd11.gif)

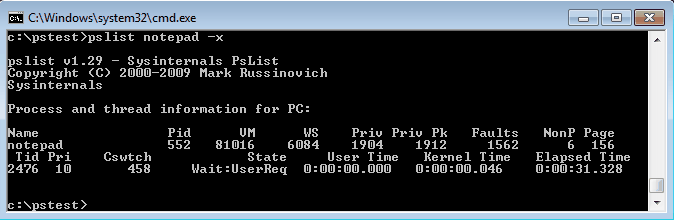
Для боле полного ознакомления с PowerShell можно воспользоваться встроенной справкой, для вызова справки нужно набрать **Get-Help** ″имя командлета″

Ну и для полноты обзора рассмотрим еще одно средство для управления процессами из командной строки. Это утилиты **Pslist** и **Pskill** входящие в состав пакета [PSTools](http://technet.microsoft.com/ru-ru/sysinternals/bb896649.aspx" \t "blank)от компании Sysinternals.

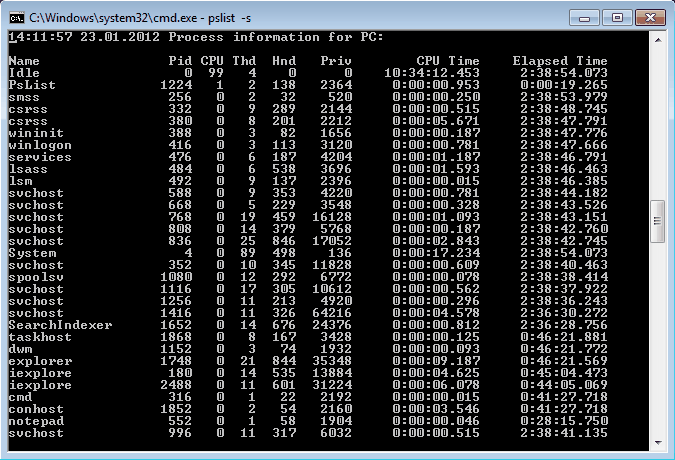
Эти утилиты не требуют специальной установки, достаточно просто скопировать их на диск. Для запуска нужно зайти в папку с утилитами и ввести в командной строке необходимую команду.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd12.gif)

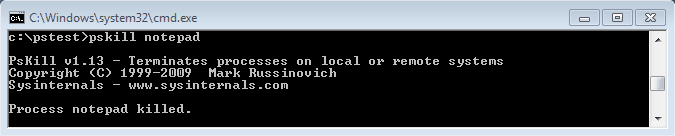
**Pslist** может выводить информацию о процессах по имени или ID, например командой **pslist notepad -x** выведем подробную информацию о нашем «многострадальном» блокноте.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd13.gif)

Особенностью утилиты **Pslist** является режим task-manager*.*В  этом режиме информация автоматически обновляется, причем можно задать время работы и интервал обновления. Запускается режим ключом **-s** , например командой **tasklist -s -r 10**запускаем режим программу в режиме task-manager*с* обновлением раз в 10 сек.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd14.gif)

Завершение процесса программой pskill предельно просто, вводим команду и имя (или ID) процесса и все.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd15.gif)

Справку по утилитам **Pslist** и **Pskill** можно посмотреть, введя команду с ключом**/?**

И еще, все манипуляции с процессами необходимо выполнять с правами администратора, для этого командную строку требуется запускать с повышением привилегий.

## **9. Соотношения между представляемым и истинным объёмом занятой дисковой памяти.**

Соотношение между представляемым и истинным объемом занятой дисковой памяти зависит от различных факторов, таких как размер кластера файловой системы, выравнивание файлов и использование дополнительных метаданных.

**Представляемый объем занятой памяти**: это объем, который отображается операционной системой и отражает размер файлов и каталогов на диске.

**Истинный объем занятой памяти**: это фактический объем, который занимает файл или каталог на диске. Он может быть больше представляемого объема из-за накладных расходов, связанных с файловой системой и использованием кластеров.

## **10. Изучение влияния количества файлов на время, необходимое для их копирования.**

Основная причина того, что для многих маленьких файлов требуется больше времени, чем для одного большого файла, заключается в том, что они копируются по одному и завершают копирование до достижения максимальной скорости, а затем начинается копирование следующего файла. Такое копирование не достигает пиковой скорости. Но для копирования большого файла требуется больше времени, и поэтому копирование успевает достичь максимальную пиковую скорости и поддерживает её до завершения копирования.

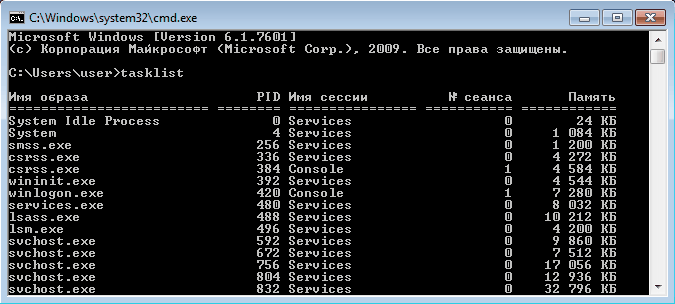
**Если простым языком**: хотя вы копируете (примерно) тот же объем данных, вы тратите дополнительные усилия на копирование файла не один раз, а тысячу раз. Это 999 дополнительных временных затрат, которые делают эту операцию намного медленнее.

## **11. Управление процессами с помощью команд операционной системы для работы с процессами.**

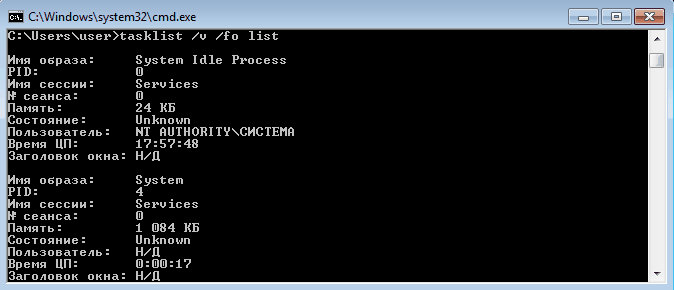
**Управление процессами из командной строки**

Для управления процессами в командной строке есть две утилиты — **tasklist** и **taskkill**. Первая показывает список процессов на локальном или удаленном компьютере, вторая позволяет их завершить. Попробуем …

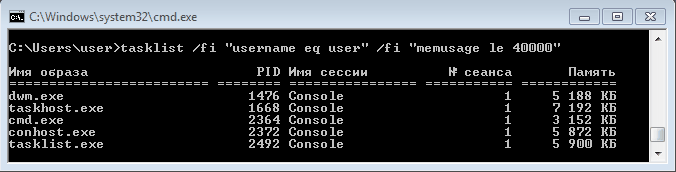
Если просто набрать команду **tasklist** в командной строке, то она выдаст список процессов на локальном компьютере.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd1.gif)

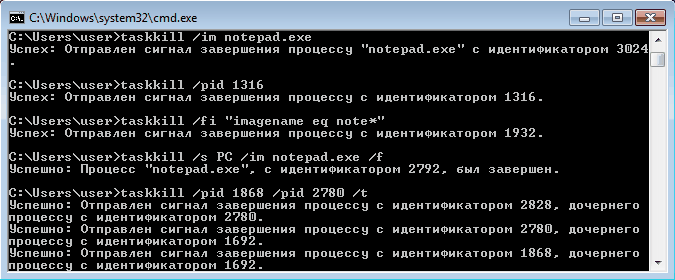
По умолчанию информация выводится в виде таблицы, однако ключ **/fo** позволяет задать вывод в виде списка или в формате CSV, а ключ **/v**показывает более подробную информацию о процессах, например команда **tasklist /v /fo list** выведет подробное описание всех процессов в виде списка.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd2.gif)

Список получится довольно большой, поэтому попробуем уточнить запрос.  Для этого используем ключ **/fi** , который позволяет использовать фильтры для вывода данных, например команда **tasklist /fi ″username eq user″ /fi ″memusage le 40000″** выводит список процессов пользователя **user**, которые потребляют не больше 40Мб памяти.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd3.gif)

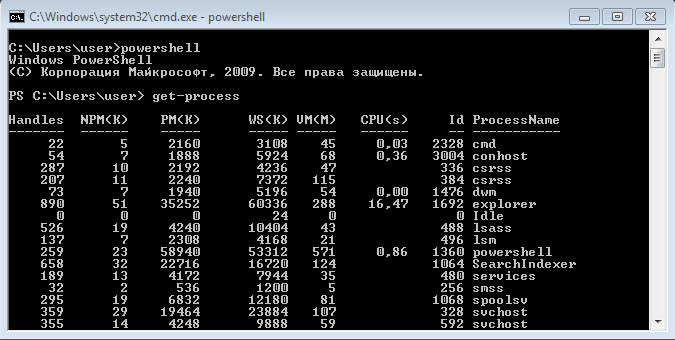
Найдя процессы, которые необходимо завершить, воспользуемся командой **taskkill.**Завершать процессы можно по имени, идентификатору процесса (PID) или задав условия с помощью фильтров. Для примера запустим несколько экземпляров блокнота (notepad.exe) и попробуем завершить его разными способами.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd5.gif)

Ключ **/f** завершает процесс принудительно, а **/t** завершает все дочерние процессы.

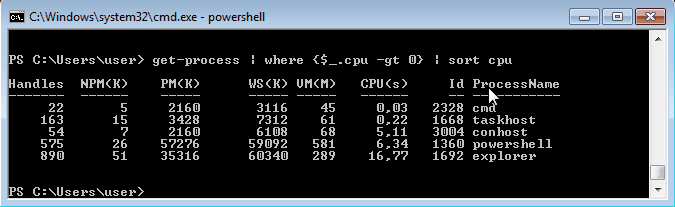
Полную справку по командам tasklist и taskkill можно получить, введя их с ключом**/?**

Теперь пустим в ход тяжелую артиллерию*—*PowerShell*.*Его можно запустить не выходя из командной строки. Для получения списка процессов используем командлет **Get-Process.**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd6.gif)

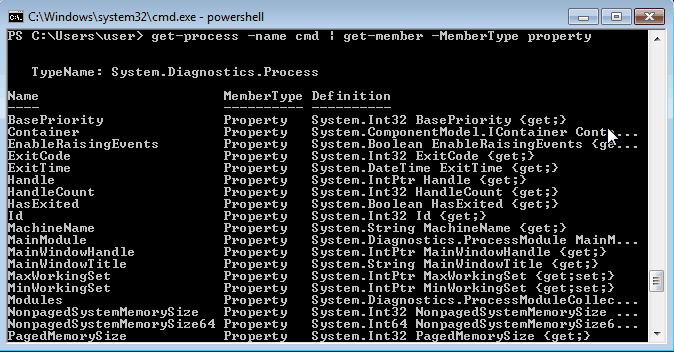
Чтобы не выводить весь список процессов можем воспользоваться командлетом **Where-Object***,*который задает фильтр для выводимой информации. Для примера выведем список процессов, которые загружают процессор и отсортируем их по возрастанию нагрузки с помощью команды:

**Get-Process | where {$\_.cpu -gt 0} | sort cpu**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd9.gif)

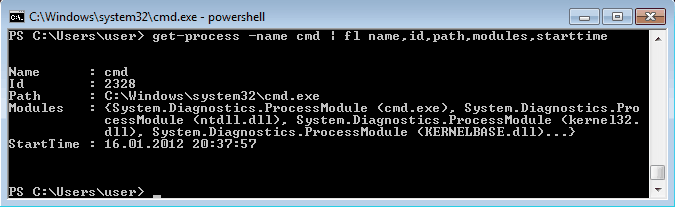
С помощью PowerShell мы можем получить любую информацию о любом процессе. В качестве примера возьмем процесс **cmd**и выведем список его свойств командой:

**Get-Process -Name cmd | Get-Member -Membertype property**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd7.gif)

Выбираем те свойства, что нам интересны ( в примере имя и ID процесса, путь к файлу, используемые модули и время запуска) и выводим их в виде списка командой:

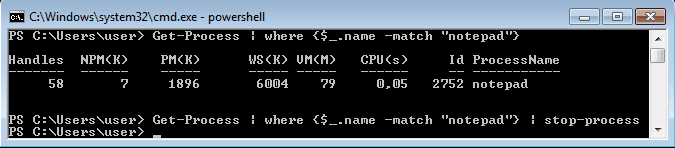
**Get-Process -Name cmd | Format-List name, id, path, modules, starttime**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd8.gif)

Таким образом мы можем посмотреть когда и кем был запущен процесс, сколько он потребляет ресурсов, где находится исполняемый файл и еще много различной информации.

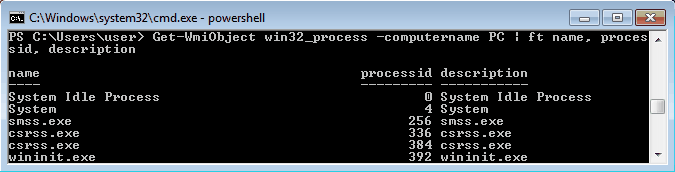
Для завершения процесса в PowerShell есть командлет **Stop-Process**. Он завершает указанный процесс по его имени или идентификатору. Однако мы поступим по другому и передадим результат выполнения командлета **Get-Process**по конвейеру:

**Get-Process | where {$\_.name -match ″notepad″}  | Stop-Process**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd10.gif)

**Get-Process** не может показать процессы на удаленном компьютере, для этого воспользуемся командлетом **Get-WmiObject** , например посмотрим процессы на удаленном компьютере PC командой:

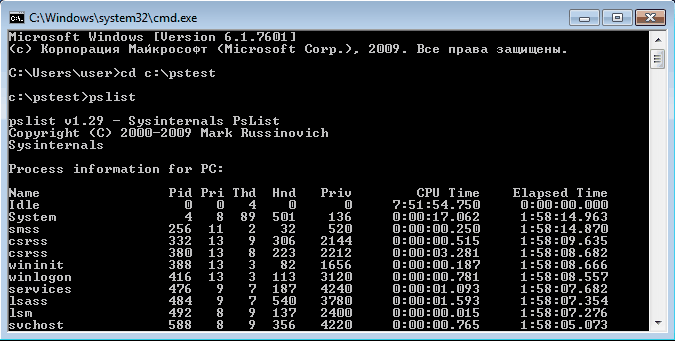
**Get-WmiObject win32\_process -computername PC | ft name, processid, description**

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd11.gif)

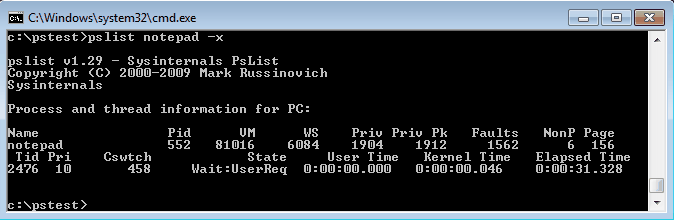
Для боле полного ознакомления с PowerShell можно воспользоваться встроенной справкой, для вызова справки нужно набрать **Get-Help** ″имя командлета″

Ну и для полноты обзора рассмотрим еще одно средство для управления процессами из командной строки. Это утилиты **Pslist** и **Pskill** входящие в состав пакета [PSTools](http://technet.microsoft.com/ru-ru/sysinternals/bb896649.aspx" \t "blank)от компании Sysinternals.

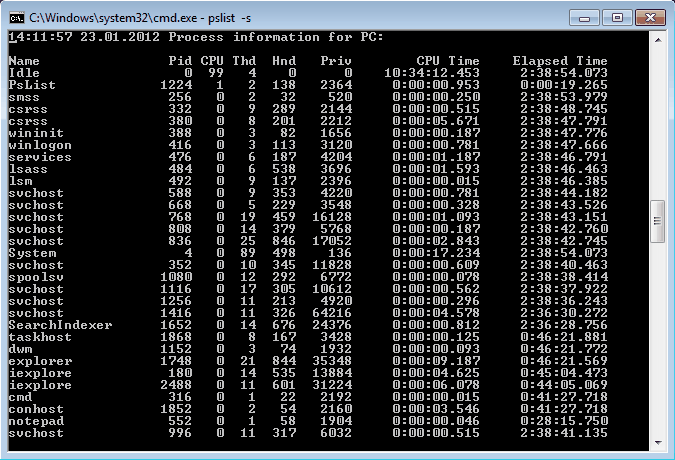
Эти утилиты не требуют специальной установки, достаточно просто скопировать их на диск. Для запуска нужно зайти в папку с утилитами и ввести в командной строке необходимую команду.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd12.gif)

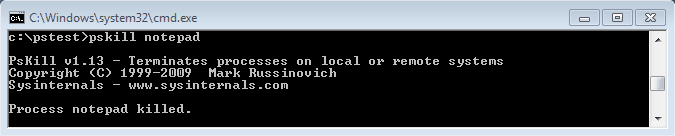
**Pslist** может выводить информацию о процессах по имени или ID, например командой **pslist notepad -x** выведем подробную информацию о нашем «многострадальном» блокноте.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd13.gif)

Особенностью утилиты **Pslist** является режим task-manager*.*В  этом режиме информация автоматически обновляется, причем можно задать время работы и интервал обновления. Запускается режим ключом **-s** , например командой **tasklist -s -r 10**запускаем режим программу в режиме task-manager*с* обновлением раз в 10 сек.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd14.gif)

Завершение процесса программой pskill предельно просто, вводим команду и имя (или ID) процесса и все.

[](https://windowsnotes.ru/wp-content/uploads/2012/01/cmd15.gif)

Справку по утилитам **Pslist** и **Pskill** можно посмотреть, введя команду с ключом**/?**

И еще, все манипуляции с процессами необходимо выполнять с правами администратора, для этого командную строку требуется запускать с повышением привилегий.

## **12. Работа с программой «Файл-менеджер Проводник».**

**Файл**-**менеджер** **Проводник** – это служебная программа для работы с файловой структурой компьютера ОС Windows. Проводник предоставляет возможность увидеть все содержимое ПК, позволяя запускать прикладные программы, копировать и перемещать файлы, осуществлять поиск файлов, создавать папки, открывать документы и выполнять другие задачи.

Рабочее поле Проводника разделено на два окна. В левом окне Проводника показана файловая структура (оглавление или дерево дисков и папок). Знак «>» напротив диска или папки говорит о том, что структура скрыта. Знак «˅» – структура видна. Отсутствие какого-либо знака означает, что в данной папке нет вложенных папок. В левом окне Проводника диски и папки открываются одинарным щелчком по их значкам.

В правом окне Проводника показано содержимое открытого объекта – вложенные папки и файлы. В этом окне Проводника диски, папки и файлы открываются двойным щелчком.

Способы открыть Проводник:

1. С помощью команды Windows + E
2. Нажав на него в списке программ через поисковую строку Windows
3. Нажав на него в панели задач

## **13. Диагностика и коррекция ошибок операционной системы, контроль доступа к операционной системе.**

**Диагностика неполадок**

1. Зайти в панель управления, после **запустить апплет «устранение неполадок».**
2. Открывается вкладка. В ней несколько подкатегорий: *программы* – с помощью этого инструмента осуществляется выполнение приложений, в которых возникли проблемы с несовместимостью. При запуске и поиске утилита выполняется сканирование системы и отобразит список с установленными в ней приложениями.
3. Устранение неполадок: **В число средств входят настройка устройства, звук, сеть, принтер, проигрывание видео*.***

Для устранения проблемы следует использовать предлагаемое: **Для пользовательского устранения неполадок функционирует окно с высвечивающимися ошибками и пути решения проблемы.**

На Windows 10 появилась функция: **«Поиск/Устранение неполадок»,** которая позволяет обнаружить проблему еще до запуска компьютера. **Перезагрузите компьютер с зажатой клавишей Shift**. После перезагрузки кликаем на экране **«Поиск/Устранение неполадок»**.

**Контроль операционной системы** включает в себя**:**

- контроль полномочий доступа к компьютерной технике, программ, программной документации, файлов данных, данных на выходе

- контроль “электронной подписи”, отчеты системы о дате, времени и месте несанкционированного доступа

**Физическая защита и контроль доступа**

Элементами физической защиты являются:

- ограничение физического доступа в помещения, оборудованных компьютерами, и к библиотекам данных (закрытые двери, карточки-ключей, охрана)

- регистрация доступа к компьютерным системам (пароли, карточки-ключи)

- предварительное разрешение руководства на работу вне расписания

- физический контроль помещений перед закрытием

**Прикладной контроль** – он направлен на обеспечение обоснованной уверенности в полноте, своевременности, точности и правильности автоматической обработки данных.

Прикладной контроль включает:

- Контроль ввода

- Контроль обработки

- Контроль вывода

1. **Основные средства контроля ввода**:

- контроль полномочий на ввод информации;

- исправление ошибок;

- использование контрольных цифр;

1. **Контроль проверки информации**:

- сверка номеров счетов с данными файла-перечня;

- тест разрядов, которые содержат коды самопроверки;

- проверку остатка на определенных счетах (например, сумма основных фондов, которые не амортизированы полностью, меньше нуля);

- проверка на буквенные символы в цифровом поле;

И другие;

1. **Контроль вывода** выполняется по принципам:

- соответствие сумм ввода и вывода (проверяется автоматически или вручную);

- поэлементная проверка данных ввода и вывода в особенно важных случаях;

- контроль доступа к данным на выходе.

## **14. Работа с файловыми системами и дисками.**

Файловая система отвечает за оптимальное логическое распределение информационных данных на конкретном физическом носителе.

Основными функциями файловой системы являются:

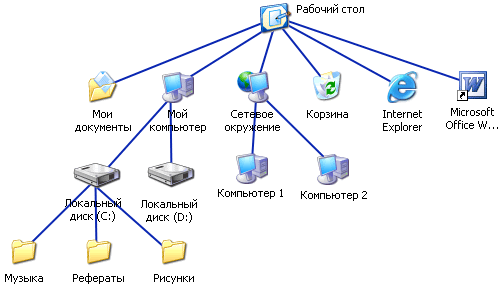
* размещение и упорядочивание на носителе данных в виде файлов;
* определение максимально поддерживаемого объема данных на носителе информации;
* создание, чтение и удаление файлов;
* назначение и изменение атрибутов файлов (размер, время создания и изменения, владелец и создатель файла, доступен только для чтения, скрытый файл, временный файл, архивный, исполняемый, максимальная длина имени файла и т.п.);
* определение структуры файла;
* поиск файлов;
* организация каталогов для логической организации файлов;
* защита файлов при системном сбое;

Функции файловой системы нацелен на решение следующих задач:

* присвоение имен файлам;
* программный интерфейс работы с файлами для приложений;
* отображение логической модели файловой системы на физическую организацию хранилища данных;
* поддержка устойчивости файловой системы к сбоям питания, ошибкам аппаратных и программных средств;
* содержание параметров файла, необходимых для правильного взаимодействия с другими объектами системы (ядро, приложения и пр.).

**Файловые системы Windows**

* Исходный код файловой системы, получившей название FAT, был разработан владельцом Microsoft Билла Гейтса . Основной задачей FAT была работа с данными в операционной системе Microsoft . Файловая система FAT претерпела несколько модификаций . Основным отличием каждой версии является преодоление ограниченного объема доступной для хранения информации. В дальнейшем были разработаны еще две более совершенные системы обработки и хранения данных – NTFS и ReFS.



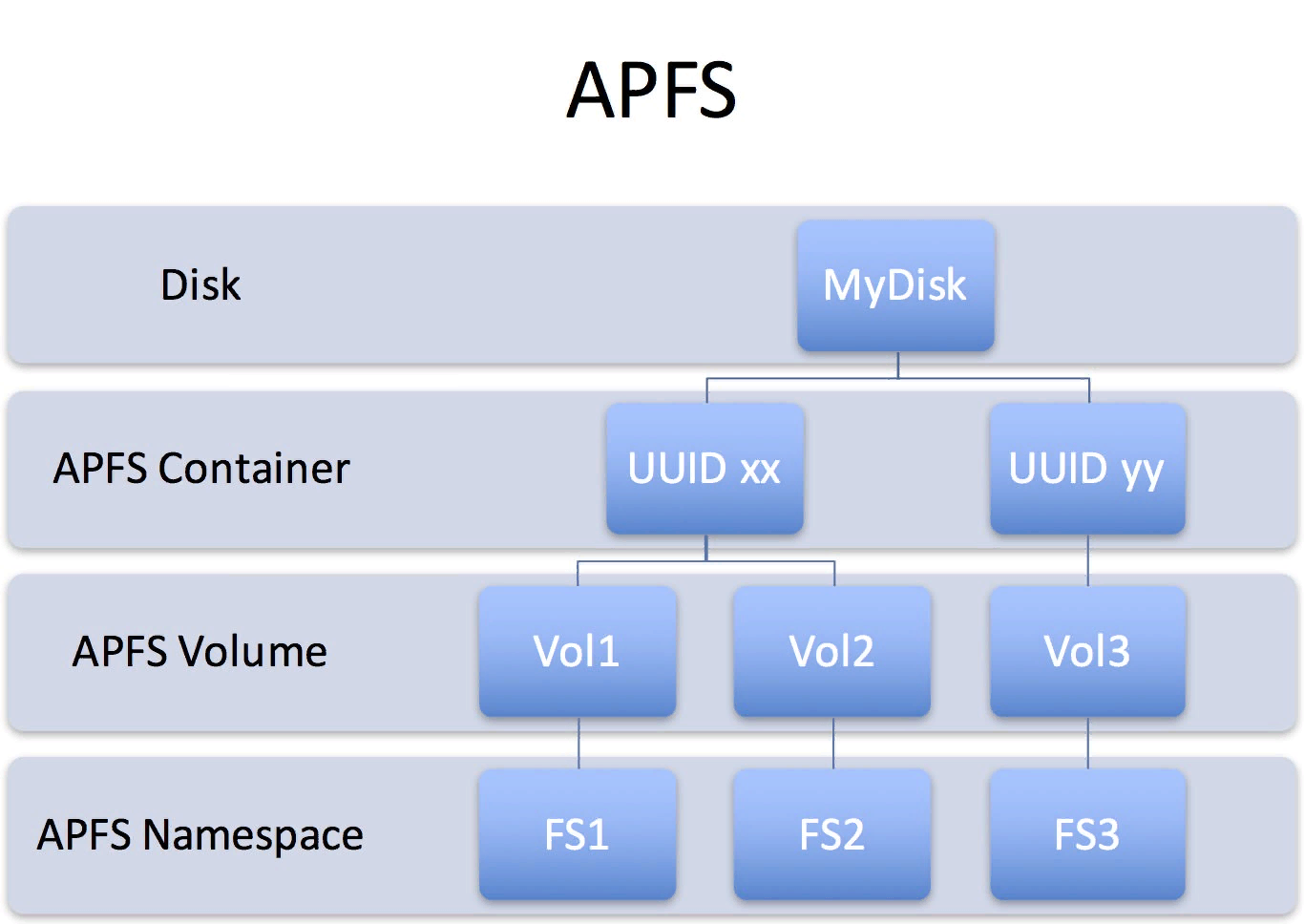
**NTFS (файловая система новой технологии)**

Стандарт NTFS разработан с целью устранения недостатков, присущих более ранним версиям ФС. Система NTFS расширила допустимый предел размера файлов до шестнадцати гигабайт, поддерживает разделы диска до 16 Эб (эксабайт, 1018 байт). Файловая система позволяет использовать расширенные имена файлов, включая поддержку многоязычности в стандарте юникода UTF, в том числе в формате кириллицы. Встроенное приложение проверки жесткого диска или внешнего накопителя на ошибки файловой системы [chkdsk](https://timeweb.com/go?url=https%3A%2F%2Fdocs.microsoft.com%2Fru-ru%2Fwindows-server%2Fadministration%2Fwindows-commands%2Fchkdsk&hash=e970cd170edaca0da388c33ce82f1276fa1ad373" \t "_blank) повышает надежность работы харда, но отрицательно влияет на производительность.

**Файловые системы macOS**

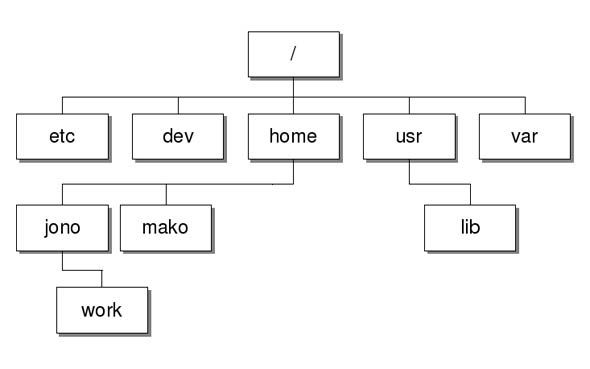
Для операционной системы macOS компания Apple использует собственные разработки файловых систем:

HFS+, которая является усовершенствованной версией HFS, и ее более соверешенный аналог APFS. Стандарт HFS+ используется во всех устройствах под управлением продуктов Apple, включая компьютеры Mac, iPod, а также Apple X Server.



**Файловые системы Linux**

В отличие от ОС Windows и macOS, ограничивающих выбор файловой системы предустановленными вариантами, Linux предоставляет возможность использования нескольких ФС, каждая из которых оптимизирована для решения определенных задач. Файловые системы в Linux используются не только для работы с файлами на диске, но и для хранения данных в оперативной памяти или доступа к конфигурации ядра во время работы системы. Все они включены в ядро и могут использоваться в качестве корневой файловой системы.

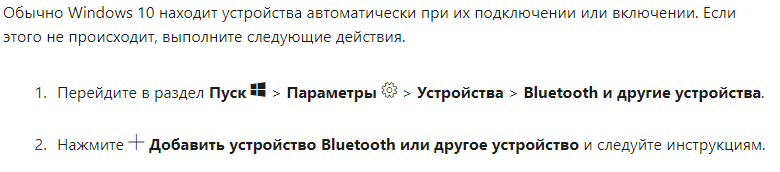


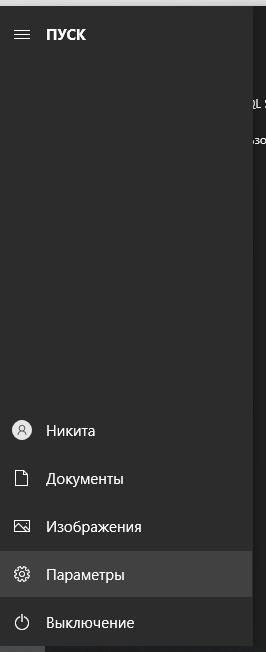
Основные файловые системы, используемые в дистрибутивах Linux:

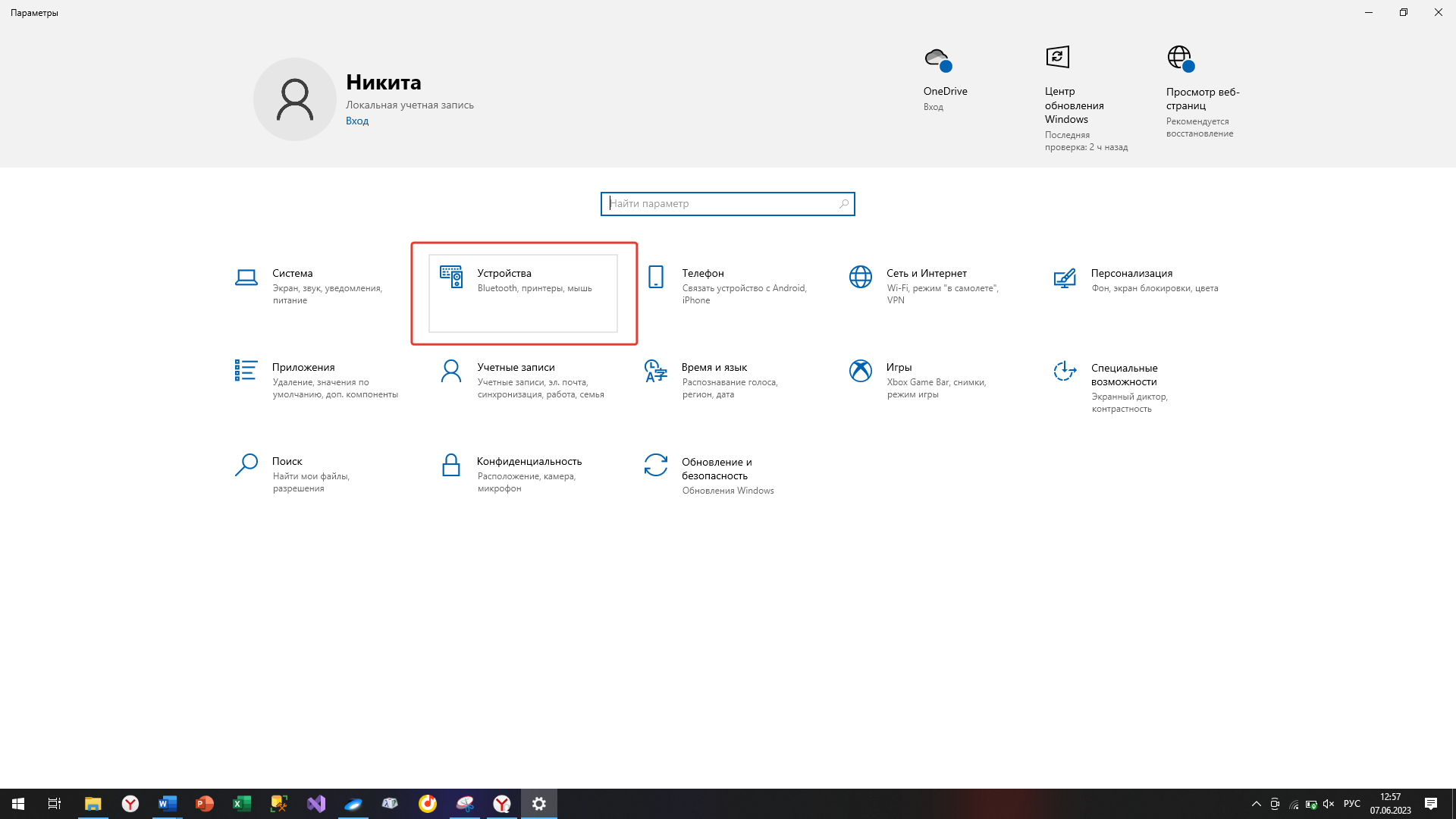
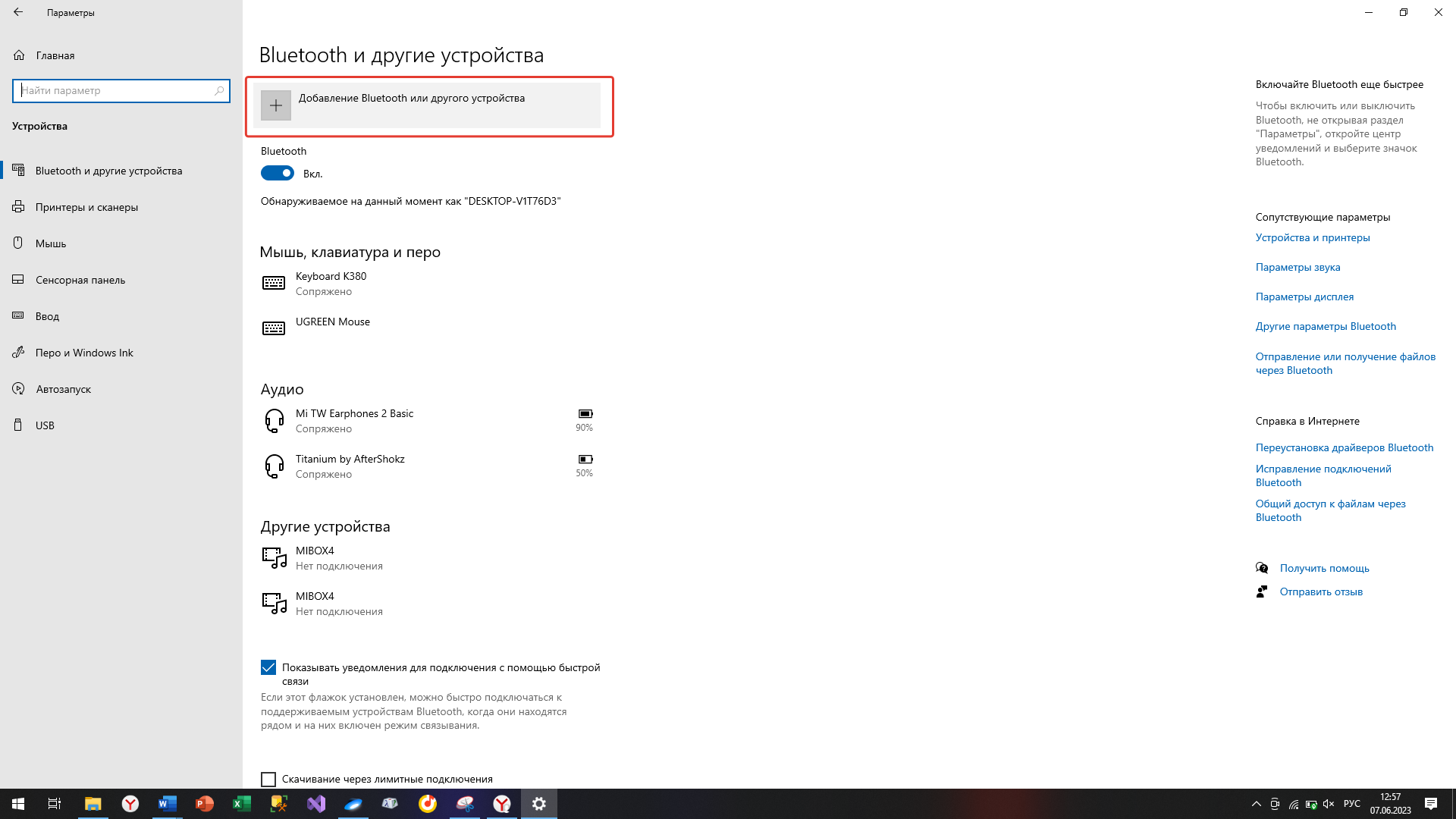
* Ext2;
* Ext3;
* Ext4;

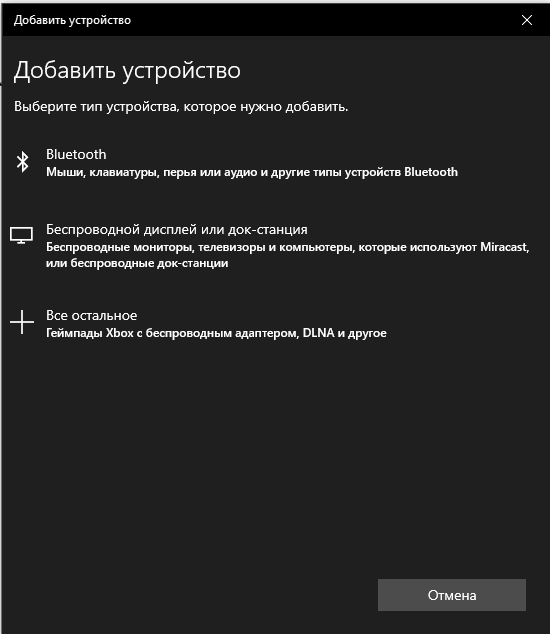
**Ext2, Ext3, Ext4** или **Extended Filesystem**– стандартная файловая система, первоначально разработанная еще для Minix. Содержит максимальное количество функций и является наиболее стабильной в связи с редкими изменениями кодовой базы. Начиная с ext3 в системе используется функция журналирования. Сегодня версия ext4 присутствует во всех дистрибутивах Linux.

## **15. Установка новых устройств.**

**Пошаговая инструкция:**







## **16. Использование команд работы с файлами и каталогами.**

Команды работы с каталогами:

**dir** – просмотр содержимого каталога.

Общий вид: ***dir \диск\маршрут\имя файла***

**cd –** смена каталога.

Общий вид: ***cd \маршрут***

**md**– создание каталога.

Общий вид: ***md \маршрут***

**rd**– удаление пустого каталога.

Общий вид: ***rd \маршрут***

Команды работы с файлами:

**copy con**– создание текстового файла.

Общий вид: ***copy con \имя файла.***

**opy** – команда копирования файла (группы файлов).

Общий вид:

***copy \диск\маршрут\имя файла\диск\маршрут\имя файла***

**Т.е.: от сюда вот сюда**

**del**– удаление файла.

Общий вид: ***del \диск\маршрут\имя файла.***

**move** – команда перемещения файла (группы файлов).

Общий вид: **move** ***\диск\маршрут\имя файла\диск\маршрут***

***Т.е.: ЭТО от сюда вот сюда***

**type** – вывод содержимого файла на экран.

Общий вид: ***type \диск\маршрут\имя файла.***

**ren** – переименование файла.

Общий вид: ren \диск\маршрут\ имя файла старое имя файла новое.

## **17. Файловая система и ввод и вывод информации.**

**Файловая система:**

**Файл** – это именованная область внешней памяти, в которую можно записать и из которой можно прочитать данные.

Назначением файлов является:

· Долговременное и надежное хранение информации

· Совместное использование информации несколькими приложениями (пользователями).

**Файловая система** – это часть ОС, включающая совокупность всех файлов на диске, наборы служебных структур данных (каталоги, дескрипторы файлов, списки свободных кластеров диска) и комплекс программных средств для операций с файлами (создания, чтения, удаления и т.д.).

Задачи, которые должна решать ФС , зависят от типа ОС. В наиболее простой (однозадачной) ОС, например, MS DOS, эти задачи включают:

· Именование файлов

· Программный интерфейс для приложений

· Отображение логической модели ФС на физическую организацию носителя

· Устойчивость ФС к сбоям питания, ошибкам аппаратных и программных средств ВС.

· Использование файла как разделяемого ресурса

Все файлы делятся на:

1. **Обычные файлы** – содержащие произвольную информацию. ОС не контролирует их содержание. Единственные файлы, которые ОС должна распознавать и контролировать – это файлы с исполняемыми модулями данной ОС.

2. **Каталоги** – это особые файлы фиксированной структуры, создаваемые самой ОС, и содержащие информацию о других файлах (обычных или каталогах), зарегистрированных в данном каталоге. Каталоги устанавливают соответствие между именами файлов и их характеристиками.

3. **Специальные файлы** – это фиктивные файлы, ассоциированные с УВВ и используемые для унификации механизмов доступа к файлам и ВУ. Специальные файлы позволяют пользователям выполнять команды в/в на УВВ в форме чтения или записи данных в файл.

**Ввод/Вывод:**

Основными компонентами подсистемы в/в являются драйверы, управляющие внешними устройствами. С некоторой условностью, к ней можно отнести и диспетчер прерываний, так как источником прерываний, в основном, являются внешние устройства.

От подсистемы В/В требуется так спланировать работу задействованных ВУ (точнее их контроллеров), чтобы с одной стороны, обеспечит максимальную пропускную способность системы в целом, с другой – чтобы обеспечить приемлемое время реакции каждого драйвера на независимые события контроллера. Кроме того, необходимо минимизировать загрузку процессора задачами в/в, освободив его для пользовательских процессов.

Отметим, что подсистема в/в работает в системе реального времени

Устройства ввода-вывода можно разделить на 2 категории:

**Блочные** устройства ввода-вывода хранят информацию в виде блоков фиксированной длины. Важным свойством таких устройств является то, что к ним можно применять операцию позиционирования и считывать или записывать каждый блок независимо от других (произвольный доступ). Все блоки адресуются. Например: жесткий диск, магнитные ленты, флешки.

**Символьные** устройства ввода-вывода принимают или выдают поток символов, а не блоки. К такой информации невозможно применять операции позиционирования и произвольный доступ невозможен, так как данные не адресуются. Например: принтер, мышь (в качестве устройства указателя).

**Способы осуществления ввода-вывода**

**Программный ввод-вывод**

В данном варианте ввода-вывода вся работа возлагается на процессор. Способ работы применяемый здесь - активное ожидание. Такой вариант работы применяется в встроенных системах, так как там нет других процессов, которые требуют времени, а частое использование прерываний снизило бы производительность. Также это самый простой для реализации вариант работы.

**Ввод-вывод, управляемый прерываниями**

В данном варианте ввода-выводе используется помощь аппаратуры, т.е. прерываний, а не только программные методы. После инициирования записи очередного блока данных мы можем вызвать планировщик и дать поработать другому процессу некоторое время, пока устройство ввода-вывода не вызовет прерывание, тем самым оповещая о готовности принять новый блок данных для записи.

**Ввод-вывод с помощью DMA-контроллера**

В данном варианте ввода-вывода вся работа по записи данных возлагается с процессора на DMA контроллер и снижается количество прерываний.

## **18. Конфигурация файлов.**

**Конфигурирование файлов в Windows** — это процесс настройки параметров и настроек системных файлов, которые используются для работы операционной системы Windows. Эти файлы могут включать конфигурационные файлы, реестр Windows, файлы системных драйверов и другие системные файлы.  
  
Для конфигурирования файлов в Windows необходимо выполнить следующие шаги:  
  
1. Открыть "Панель управления" и выбрать "Система и безопасность".  
  
2. Выбрать "Администрирование".  
  
3. Выбрать "Службы".  
  
4. Выбрать нужную службу и нажать правой кнопкой мыши на нее.  
  
5. Выбрать "Свойства".  
  
6. На вкладке "Общие" выбрать тип запуска службы (автоматический, ручной или отключенный).  
  
7. На вкладке "Вход в систему" выбрать учетную запись, от имени которой будет запускаться служба.  
  
8. Нажать "ОК".  
  
Также можно конфигурировать файлы Windows, используя командную строку или редактор реестра. Однако, при работе с этими инструментами необходимо быть осторожным, чтобы не повредить системные файлы и не нарушить работу операционной системы.

## **19. Работа с операционной оболочкой.**

**Операционная оболочка (ОБ)** – программу-надстройку к операционной системе, обеспечивающую доступ пользователя к командам и ресурсам операционной системы посредством более удобного интерфейса и реализующую дополнительные функции распределения ресурсов вычислительной системы управления файлами.

Операционная оболочка Windows в основном ориентирована на использование мыши. Мышь управляет указателем на экране. Когда вы передвигаете ее по коврику, указатель перемещается по экрану. На разных участках окна программы указатель может принимать разную форму. После нажатия кнопки мыши указатель активизируется.

Основные возможности большинства оболочек ОС:

* просмотр содержимого папки (диска) в различных форматах;
* вывод информации о скрытых файлах и папках;
* сортировка и фильтрация информации о содержимом папки (диска);
* просмотр структуры диска (дерева папок);
* переименование, копирование, пересылка, удаление и быстрый поиск файлов;
* просмотр, создание, сравнение и синхронизация каталогов;
* просмотр, создание и редактирование текстовых файлов;
* архивация, обновление и разархивация архивных файлов, просмотр архивов;
* слияние файлов;
* работа с сетью;
* запуск программ и создание пользовательских меню запуска.

Примерами оболочек ОС являются следующие программы: DOS Navigator, Windows Explorer (Проводник), Total Commander

## **20. Управление процессами в операционной системе.**

**Процесс** – это выполнение программы, которая выполняет действия, указанные в этой программе. Его можно определить как исполнительный модуль, в котором выполняется программа. ОС помогает вам создавать, планировать и завершать процессы, используемые процессором. Процесс, созданный основным процессом, называется дочерним процессом.

**Архитектура процесса:**

* **Стек:** Стек хранит временные данные, такие как параметры функции, адреса возврата и локальные переменные.
* **Куча:** Распределяет память, которая может быть обработана во время выполнения.
* **Данные:** содержит переменную.
* **Текст:** текстовый раздел включает текущее действие, которое представлено значением счетчика программы

**Управление процессами**

* Создание и завершение процессов
* Планирование и диспетчеризация процессов
* Переключение процессов
* Синхронизация и поддержка обмена информацией между процессами
* Организация управляющих блоков процессов

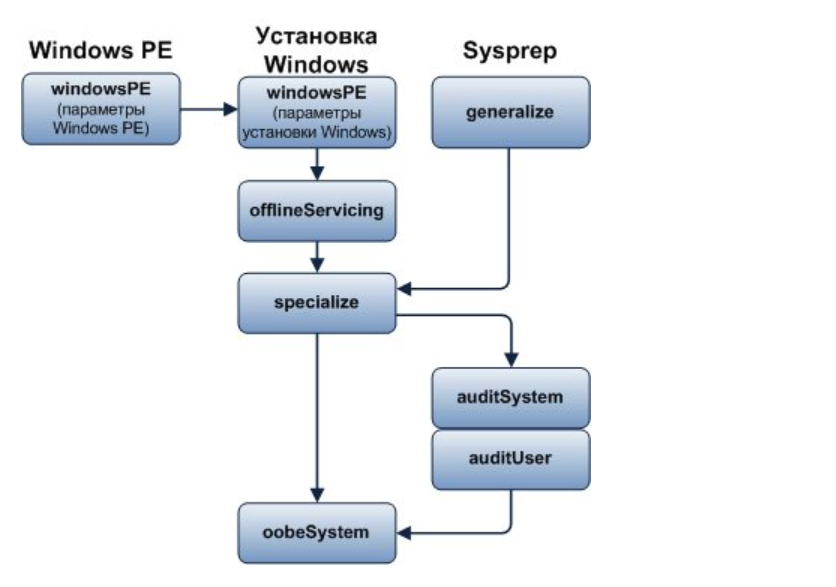
## **21 Резервное хранение, командные файлы.**

* **Резервное копирование** — создание копий файлов, папок или систем на дополнительных носителях информации.
* Резервное копирование необходимо для формирования архива данных, защищенного от изменений и повреждений, а также восстановления в случае повреждения или сбоев в первоисточнике.
* Резервное хранение данных:  
  backup – сохранение файловой системы, восстановление данных  
  cpio – архивирование данных  
  dump – дамп указанных частей объектных файлов (ext2/ext3)  
  restore – восстановление данных (пара dump/restore)  
  tar – архивирование данных  
  *Утилиты*:  
  - gzip, gunzip  
  -bzip2, bunzip2  
  - zip
* Программирование командных файлов (в оболочке Shell):
* **Командный файл** — текстовый файл с правами на запуск.
* В командном файле можно записать последовательность команд на исполнение. Кроме этого, можно написать небольшой скрипт.  
  Некоторые возможности по написанию скрипта:  
  • Условный оператор "IF"  
  • Оператор вызова ("CASE")  
  • Оператор цикла с перечислением ("FOR")  
  • Оператор цикла с истинным условием ("WHILE")  
  • Оператор цикла с ложным условием ("UNTIL")  
  • Можно писать свои функции.

## **22. Планирование и установка операционной системы Windows.**

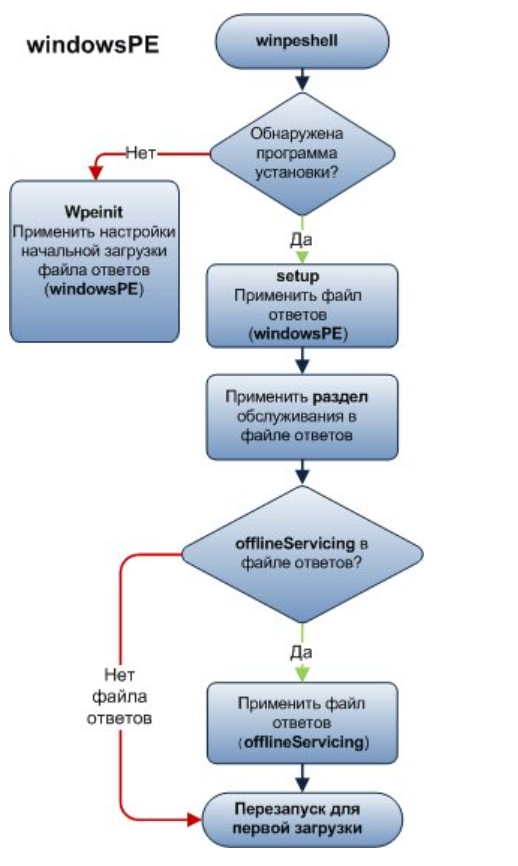
* Первую свою команду компьютер получает от ПЗУ – микросхемы, которая расположена на материнской плате, питается от батарейки, и поэтому записанные в ней программы не стираются после выключения компьютера. Именно к ПЗУ обращается процессор в момент включения и делает это всегда и автоматически. В ПЗУ находятся программы тестирования компьютера BIOS. Работа BIOS отображается на экране белыми бегущими строками. В этот момент компьютер проверяет свои устройства – оперативную память, жесткий диск и дисководы других дисков, наличие клавиатуры и др. устройств. Если что-то не работает, BIOS докладывает о неисправности, иначе заканчивает свою работу и дает команду загрузить с жесткого диска в оперативную память специальную программу.
* Эта программа находится в специальном загрузочном секторе диска и называется Master Boot (загрузчик ОС). Она очень маленькая и ее основное назначение – считать в ОЗУ операционную систему с системного диска.
* После окончания загрузки ОС управление передается командному процессору и на экране появляется графический интерфейс. Отныне все, что мы делаем с компьютером, происходит под управлением операционной системы.

**Этапы установки Windows**



**WindowsPE** – это проход программы установки Windows, во время которого сначала запускается среда предустановки, а затем и сама программа установки, производящая первоначальную конфигурацию, такую как разбиение дисков на разделы, определение местоположение образа (локальный или сетевой), выбор редакции и разрядности операционной системы, копирование образа на жесткий диск компьютера.

* **Если программа установки Windows запущена из среды WindowsPE** (т.е. использовалось загрузочное съемное устройство), то **применяются параметры как для WindowsPE, так и для программы установки Windows.**
* Если установка запущена из предыдущей версии операционной системы, то **применяются параметры только для программы установки Windows.**



***offlineServicing***

**В ходе этого этапа настройки можно добавить драйверы в образ перед непосредственным запуском Windows.** Это позволяет установить и обработать драйверы от изготовителя оборудования во время работы программы установки Windows.

Этот этап настройки также используется для применения обновлений к образу Windows в сценариях обслуживания.

* **offlineServicing запускается в промежутке между этапом windowsPE и первой перезагрузкой системы.**

***specialize***

**Этот этап настройки используется для создания и настройки сведений образа Windows, а также сведений об оборудовании, на которое устанавливается образ.**

После первой загрузки образа Windows запускается этап настройки specialize. В ходе этого этапа создаются уникальные идентификаторы безопасности (SID). Кроме того, в ходе этого этапа можно настроить большинство компонентов Windows, включая параметры сети, региональные параметры и сведения о домене.

## **23. Работа с текстовым редактором. Работа с Архиватором.**

**Работа с текстовым редактором** — это процесс создания, редактирования и форматирования текстовых документов на компьютере.

Текстовый редактор позволяет пользователю: создавать документы, вносить изменения в уже существующие, форматировать текст, добавлять изображения и таблицы, а также сохранять документы в различных форматах.

**Работа с архиватором** — это процесс сжатия файлов для уменьшения их размера и упаковки в один файл для более удобного хранения и передачи.

Архиватор позволяет пользователю: создавать архивы, извлекать из них файлы, добавлять новые файлы в уже существующие архивы, а также защищать архив паролем. Распаковка архива происходит путем его открытия и извлечения содержимого на компьютер.