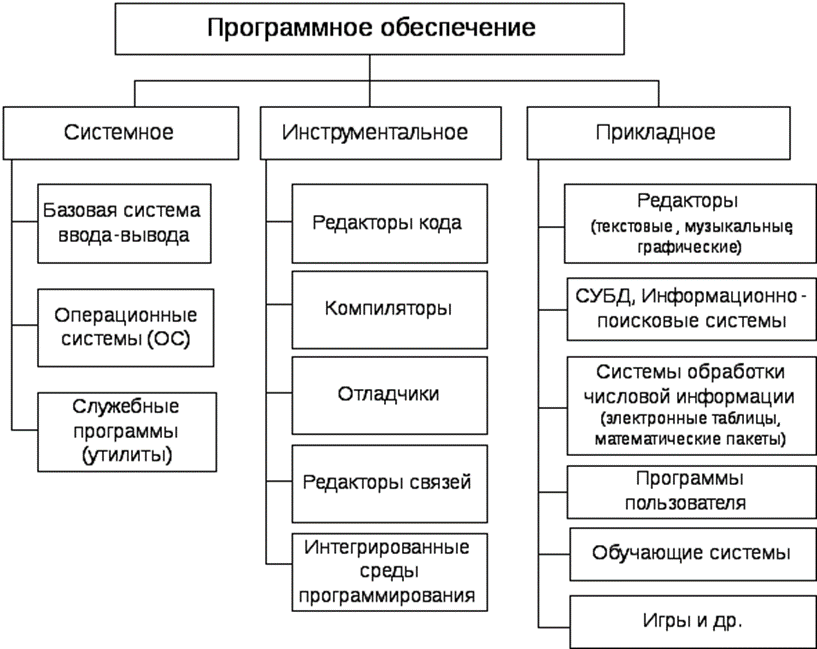
**Системное программное обеспечение (СПО)**

**Аппаратно-программное обеспечение ЭВМ можно разделить на:**

1. Аппаратное обеспечение (напр: процессор, память, устройства ввода-вывода).

2. Системное ПО (напр: операционная система, драйверы, утилиты).

3. Прикладное и инструментальное ПО (напр: текстовые редакторы, браузеры, игры, среды разработки)



В данном курсе нас интересует *Системное Программное Обеспечение (СПО)*

**Системное программное обеспечение (СПО)** – это комплекс программ, обеспечивающих управление аппаратными ресурсами ЭВМ, взаимодействие с пользователем и выполнение прикладных программ. СПО включает в себя операционные системы, утилиты, драйверы устройств, связующее ПО (например, СУБД), и системы программирования

**Основные функции СПО:**

- Управление аппаратными ресурсами (процессор, память, устройства ввода-вывода и тд)

- Организация взаимодействия между пользователем и компьютером

- Обеспечение работы прикладного программного обеспечения

**Классификация системных программ:**

1. Операционные системы (ОС) – управляют ресурсами компьютера (Windows, Linux, macOS).

2. Загрузчики – программы, загружающие ОС в память (GRUB, NTLDR).

3. Системы программирования (инструментальные программы):

Программы, созданные для разработки другого программного обеспечения. Они включают:

* Редакторы кода: Для написания исходного кода программ.
* Компиляторы и интерпретаторы: Для преобразования кода, написанного человеком, в машинный код.
* Отладчики: Для поиска и исправления ошибок в программах.

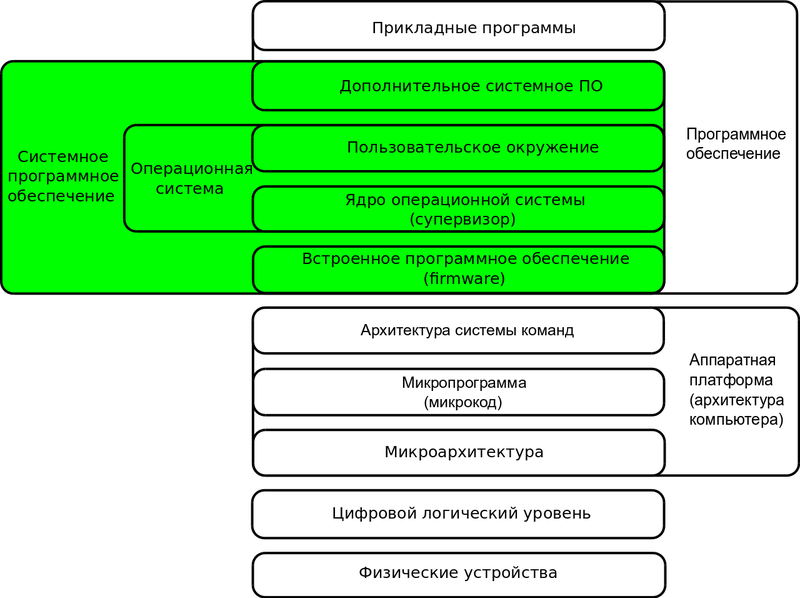
4. Связзующее программное обеспечение:

программы, которые обеспечивают взаимодействие между различными программными компонентами. Сюда относятся:

* Системы управления базами данных (СУБД): программы для создания, хранения и управления базами данных.
* Программные оболочки и интерфейсы: программы, которые предоставляют удобный интерфейс для взаимодействия пользователя с операционной системой.

5. Утилиты – вспомогательные программы для обслуживания системы (антивирусы, архиваторы). Они могут быть предназначены для диагностики, обслуживания дисков, управления файлами, защиты от вирусов и других функций.

6.Драйвера устройств - программы, которые позволяют операционной системе взаимодействовать с конкретным аппаратным устройством (например, принтером, сетевой картой, видеокартой)



**Организация взаимодействия с ЭВМ**

Системное программное обеспечение взаимодействует с ЭВМ следующим образом:

1. Загрузка и инициализация:

При включении компьютера BIOS (базовая система ввода-вывода) инициализирует аппаратное обеспечение и запускает операционную систему.

2. Управление ресурсами:

Операционная система распределяет процессорное время, оперативную память и доступ к устройствам между различными прикладными программами.

3. Интерфейс между аппаратным и программным обеспечением:

Драйверы устройств позволяют операционной системе и прикладным программам "общаться" с аппаратным обеспечением, например, с принтером или сетевой картой.

4. Работа с данными:

СПО управляет хранением, обработкой и передачей данных, обеспечивая их сохранность и доступность.

5. Безопасность и стабильность:

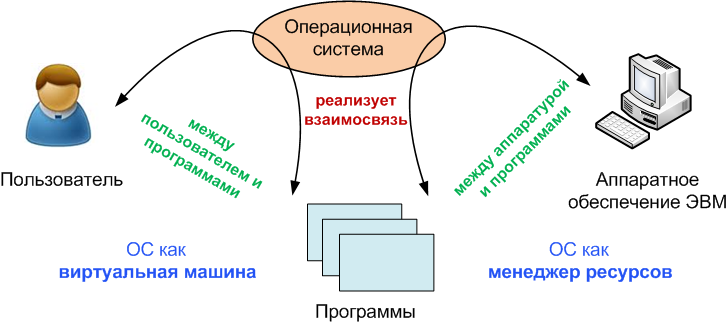
СПО обеспечивает защиту системы от несанкционированного доступа и сбоев, а также координирует работу всех компонентов, предотвращая конфликты и обеспечивая стабильную работу.

6. Пользовательский интерфейс:

Операционная система предоставляет пользователю средства для взаимодействия с компьютером, например, графический интерфейс или командную строку.

Ключевым системным ПО, координирущим работу всех компонентов системы, является **операционная система (ОС).**

Она выполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.ОС обычно хранится во внешней памяти компьютера — на диске. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ. Этот процесс называется загрузкой операционной системы. Интерфейс ОС обеспечивает взаимодействие между программами и ядром.



**Интерфейс ОС** – набор правил и методов взаимодействия между программами и ОС. Основная задача интерфейса операционной системы — обеспечить эффективное и комфортное взаимодействие между человеком и компьютером, позволяя пользователю: вводить и получать информацию, управлять приложениями, осуществлять обмен данными с другими устройствами

Виды интерфейсов:

1. Пользовательский интерфейс

* **GUI (Graphical User Interface)** – графический, где управление осуществляется через окна, кнопки и меню
* **CLI (Command Line Interface)** – командная строка, где команды вводятся текстом в консоли.

GUI делает работу с системой наглядной и интуитивно понятной простому пользователю, тогда как CLI часто используется для специализированных задач, например, программистами.

2. Программный интерфейс (**API, интерфейс программирования приложения, application programming interface**) – программный интерфейс, то есть описание способов взаимодействия одной компьютерной программы с другими (в противоположность пользовательскому интерфейсу, используемому для взаимодействия конечного пользователя с программой)

API упрощает процесс программирования при создании приложений, абстрагируя базовую реализацию и предоставляя только объекты или действия, необходимые разработчику. API для ввода/вывода файлов может дать разработчику функцию, которая копирует файл из одного места в другое, не требуя от разработчика понимания операций файловой системы, происходящих «внутри ящика»

3. Аппаратный интерфейс для взаимодействия с устройствами (драйверы)

**Системные вызовы**

**Системный вызов** – запрос программы к ядру ОС для выполнения привелигированных операций (например, чтение файла).

Ядро ОС исполняется в привилегированном режиме работы процессора. Для выполнения межпроцессной операции или операции, требующей доступа к оборудованию, программа обращается к ядру, которое, в зависимости от полномочий вызывающего процесса, исполняет либо отказывает в исполнении такого вызова.

**Привилегии**

**Привилегии** при системном вызове — это разрешения, которыми обладают процессы для доступа к ресурсам ядра ОС (например, файлам, устройствам, памяти) через системный вызов. Операционная система (ядро) проверяет эти привилегии, и если у процесса их нет, доступ к запрашиваемым ресурсам запрещается, предотвращая несанкционированные действия.

**Как работают привилегии:**

1. Программа запрашивает действие:

Приложение инициирует системный вызов, чтобы выполнить действие, например, открыть файл.

1. Переключение в режим ядра:

Управление передается ядру операционной системы.

2. Проверка привилегий:

Ядро определяет, имеет ли процесс необходимые права для выполнения этого вызова.

3. Выполнение или отказ:

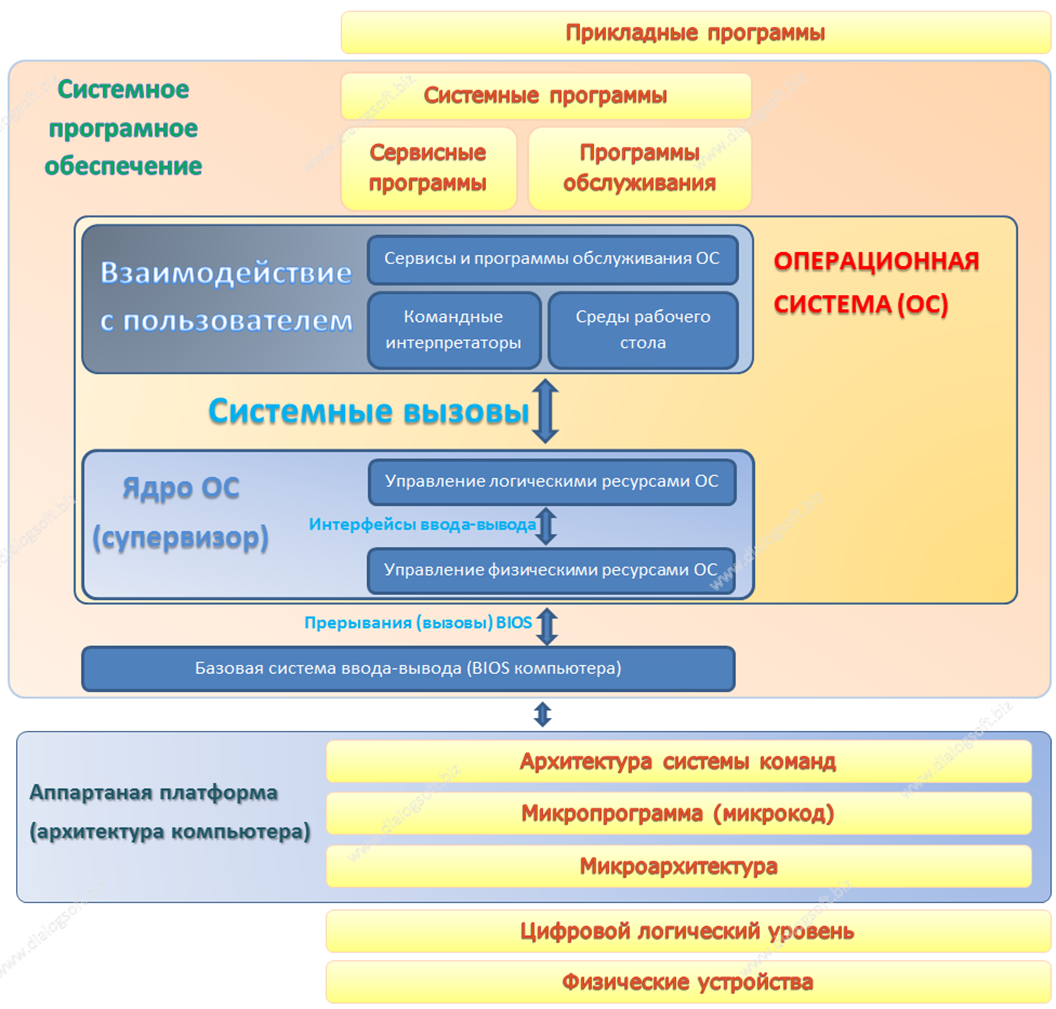
• Если привилегии есть, ядро выполняет запрошенное действие.

• Если привилегий нет, система отклоняет запрос, предотвращая доступ к защищенным ресурсам.

Современные операционные системы предусматривают:

* разделение времени между выполняющимися вычислительными процессами (многозадачность)
* разделение полномочий, препятствующее обращению исполняемых программ к данным других программ и оборудованию: приложения обычно ограничены своим адресным пространством таким образом, что они не могут получить доступ или модифицировать другие приложения, исполняемые в операционной системе, либо саму операционную систему, и обычно не могут напрямую получать доступ к системным ресурсам (жёсткие диски, видеокарта, сетевые устройства и т. д.).

Архитектура современных процессоров предусматривает использование защищённого режима с несколькими уровнями привилегий. Таким образом, для взаимодействия с системными ресурсами приложения используют системные вызовы, которые дают возможность операционной системе обеспечить безопасный доступ к ним. Системные вызовы передают управление ядру операционной системы, которое определяет, предоставлять ли приложению запрашиваемые ресурсы. Если ресурсы доступны, то ядро выполняет запрошенное действие, затем возвращает управление приложению.



В Windows интерфейс системных вызовов — это часть Native API в библиотеке ntdll.dll; это недокументированный API, используемый реализациями обычного Windows API и напрямую используемый некоторыми системными приложениями Windows.

**Категории системных вызовов**

Системные вызовы могут быть сгруппированы в пять больших категорий:

1.Управление процессами

load,execute,end (exit),создание процесса, завершение процесса

get/set process attributes, wait, signal события,allocate, free memory

2.Работа с файлами

create file, delete file,open, close,read, write, reposition,get/set file attributes

3.Управление устройствами

request device, release device,read, write

4.Работа с информацией

get/set time or date,get/set system data,get/set process, file, or device attributes

5.Связь, коммуникация

create, delete communication connection

send, receive messages,transfer status information\

**32- и 64-разрядные интерфейсы**

32-разрядные и 64-разрядные интерфейсы относятся к архитектуре процессора и операционной системы. Основное отличие заключается в объеме информации, который процессор может обрабатывать за один такт (32 или 64 бита).

Объем оперативной памяти (ОЗУ):

* 32-битные системы могут использовать не более 4 ГБ ОЗУ(2 в 32 степени бита)
* 64-битные системы не имеют этого ограничения и могут работать с гораздо большими объемами памяти, (теоретически — до 16 миллионов терабайт, но это практически недостижимо для современных компьютеров)

Производительность и обработка данных:

* 64-разрядная: Более эффективно обрабатывает большие объемы информации и выполняет вычисления, что приводит к лучшей производительности, особенно в требовательных приложениях.
* 32-разрядная: Менее эффективно справляется с обработкой больших данных, что снижает производительность в многозадачных сценариях.

Совместимость:  
для работы 32- и 64-разрядных программ операционная система использует разные каталоги ресурсных файлов. 64-разрядная ОС без проблем будет работать с 32-битными приложениями. Сказать это же о 32-битных ОС нельзя — они несовместимы с программами на 64-битной архитектуре.

Совместимость оборудования и ОС:

64-разрядная: Требует 64-битный процессор и может работать только на компьютерах с таким процессором.

32-разрядная: Может быть установлена как на 32-битный, так и на 64-битный процессор, но 64-разрядные версии ОС установить на 32-битный процессор нельзя.

Вопросы для самопроверки:

1. Какие программы относятся к системному ПО?

2. Что такое операционная система?

3. Что такое системный вызов и зачем он нужен?

4. В чем отличие 32 и 64 разрядных систем?