## Министерство образования Иркутской области

Государственное бюджетное профессиональное образовательное учреждение Иркутской области

«Ангарский политехнический техникум»

РАССМОТРЕНА

на заседании предметной (цикловой)

комиссии и рекомендована к применению

Председатель ПЦК

/О.Ю.Лезнова/

« » 20\_\_г.

# МЕТОДИЧЕСКАЯ РАЗРАБОТКА

## по выполнению внеаудиторной самостоятельной работы (СРС)

учебная дисциплина общепрофессионального цикла

**ОП.02. Операционные системы**

по специальности **09.02.04 Информационные системы (по отраслям)**

Преподаватель Макарова Наталья Алексеевна,

высшая квалификационная категория

Ангарск 2018

## Содержание

1. Тематический план организации СРС
2. Методические рекомендации (указания) к выполнению каждой СРС

2

# Тематический план организации самостоятельной работы

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№ п/п** | **Тема** | **Количество**  **часов** | **Цель самостоятельной**  **работы, включая ОК и ПК** | **Задания для самостоятельной работы** | **Дидактическое обеспечение и рекомендуемая литература (включая Интернет-ресурсы)** |
| 1 | Понятие операционной системы. Основные принципы построения операционной системы | 4 | Формирование умений и знаний по теме: «Тема 1.1  Общие сведения об операционных системах»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы | [Батаев А.В.](http://www.academia-moscow.ru/authors/detail/46973/) Операционные системы и среды: учебник для студ. учреждений сред. проф. образования / А.В.Батаев, Н.Ю.Налютин, С.В.Синицын. – М.: Издательский центр «Академия», 2018. – 272 с. |
| 2 | Распределение и использование ресурсов в операционной системы | 4 | Формирование умений и знаний по теме: «Тема 1.1  Общие сведения об операционных системах»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 3 | История создания операционной системы | 4 | Формирование умений и знаний по теме: «Тема 1.1  Общие сведения об операционных системах»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы.  Подготовить сообщение |
| 4 | Понятие вычислительного процесса и ресурса. | 4 | Формирование умений и знаний по теме: «Тема 1.1  Общие сведения об операционных системах»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 5 | Управление задачами и процессами.  Планирование и диспетчеризация процессов и задач | 4 | Формирование умений и знаний по теме: «Тема 2.1  Управление задачами и процессами»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 6 | Установка оборудования и драйверов для различных операционных систем | 1 | Формирование умений и знаний по теме: «Тема 2.2  Управление памятью»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 7 | Управление файлами. Основные понятия. Компоненты файловой системы. | 1 | Формирование умений и знаний по теме: «Тема 2.2  Управление памятью»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 8 | Выполнение мероприятий для защиты компьютера от вирусных атак | 1 | Формирование умений и знаний по теме: «Тема 3.1  Защищенность и отказоустойчивость операционных систем»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Ответить на контрольные вопросы |
| 9 | Эмуляторы операционных систем | 4 | Формирование умений и знаний по теме: «Тема 4.1 Принципы построения ОС. Особенности работы в конкретных ОС»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Подготовить презентацию |
| 10 | Стандартные программы ОС Windows 10 | 4 | Формирование умений и знаний по теме: «Тема 4.2 Способы организации поддержки устройств. Стандартные программы операционной системы»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Подготовить презентацию |
| 11 | Драйверы оборудования | 5 | Формирование умений и знаний по теме: «Тема 5.1 Драйверы оборудования»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Подготовить презентацию |
| 12 | Средства восстановления и защиты ОС | 5 | Формирование умений и знаний по теме: «Тема 5.1 Драйверы оборудования»  создание условий для формирования компетенций: ОК2, ОК4, ОК5, ОК 8, ОК 9 | Подготовить презентацию |

### Методические рекомендации (указания) к выполнению самостоятельной работы №1

1. Тема самостоятельной работы: *Понятие операционной системы. Основные принципы построения операционной системы*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 1.1 Общие сведения об операционных системах», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

# 1.  Основные принципы построения операционных систем.

Как комплекс системных управляющих и обрабатывающих программ, операционная система представляет собой очень сложный конгломерат взаимо­связанных программных модулей и структур данных, которые должны обеспечи­вать надежное и эффективное выполнение вычислений. Большинство потенци­альных возможностей операционной системы, ее технические и потребительские параметры — все это во многом определяется архитектурой системы — ее структу­рой и основными принципами построения.

Аналогично, и системы, реализующие мультизадачный режим работы, отличают­ся по своему строению от однозадачных систем. Если система допускает работу нескольких пользователей, то желательно иметь достаточно развитую подсистему информационной безопасности. А это, в свою очередь, налагает определенные требования и на идеологию построения операционной системы, и на выбор конкретных механизмов, помогающих реализовать защиту информационных ресурсов и ввести ограничения на доступ к другим видам ресурсов. Поскольку операционные системы помимо функций организации вычислений и организации интерфейса пользователя предоставляют интерфейсы для взаимодействия программ с операционной системой, мы в этой главе рассмотрим и интерфейсы прикладного программирования.

Среди множества принципов построения операционных систем перечислим несколь-наиболее важных: принцип модульности, принцип виртуализации, принципы

мобильности (переносимости) и совместимости, принцип открытости, принцип генерации операционной системы из программных компонентов и некоторые другие.

# 1.1. Принцип модульности.

Операционная система строится из множества программных модулей. Под *моду­лем*в общем случае понимают функционально законченный элемент системы, выполненный в соответствии с принятыми межмодульными интерфейсами. По своему определению модуль предполагает легкий способ его замены другим при наличии заданных интерфейсов. Способы обособления составных частей опера­ционной системы в отдельные модули могут быть существенно разными, но чаще всего разделение происходит именно по функциональному признаку. В значитель­ной степени разделение системы на модули определяется используемым методом проектирования системы (снизу вверх или наоборот).

Особо важное значение при построении операционных систем имеют *привилеги­рованные, повторно входимые*и *реентерабельные*модули, ибо они позволяют бо­лее эффективно использовать ресурсы вычислительной системы. Как мы уже зна­ем (см. главу 1), свойство реентерабельности может быть достигнуто различными способами, но чаще всего используются механизмы динамического выделения па­мяти под переменные для нового вычислительного процесса (задачи). В некото­рых системах реентерабельность программы получают автоматически. Этого можно достичь благодаря неизменяемости кодовых частей программ при исполнении, а также автоматическому распределению регистров, автоматическому отделению кодовых частей программ от данных и помещению последних в системную область памяти, которая распределяется по запросам от выполняющихся задач. Естествен­но, что для этого необходима соответствующая аппаратная поддержка. В других случаях это достигается программистами за счет использования специальных си­стемных модулей.

Принцип модульности отражает технологические и эксплуатационные свойства системы. Наибольший эффект от его использования достижим в случае, когда прин­цип распространен одновременно на операционную систему, прикладные програм­мы и аппаратуру. Принцип модульности является одним из основных в UNIX-системах.

Во всех операционных системах можно выделить некоторую часть наиболее важ­ных управляющих модулей, которые должны постоянно находиться в оператив­ной памяти для более скорой реакции системы на возникающие события и более эффективной организации вычислительных процессов. Эти модули вместе с некоторыми системными структурами данных, необходимыми для функционирова­ния операционной системы, образуют так называемое *ядро операционной системы,*так как это действительно ее самая главная, центральная часть, основа системы.

При формировании состава ядра требуется удовлетворить двум противоречивым требованиям. В состав ядра должны войти наиболее часто используемые систем­ные модули. Количество модулей должно быть таким, чтобы объем памяти, зани­маемый ядром, был не слишком большим. В его состав, как правило, входят мо­дули по управлению системой прерываний, средства по переводу программ из состояния счета в состояние ожидания, готовности и обратно, средства по распре­делению основных ресурсов, таких как оперативная память и процессор. В главе 1 мы уже упоминали, что операционные системы могут быть *микроядерными*и *макроядерными (монолитными).*В микроядерных операционных системах само ядро очень компактно, а остальные модули вызываются из ядра как сервисные. При этом сервисные модули могут размещаться и в оперативной памяти. В противополож­ность микроядерным в макроядерных операционных системах главная суперви-зорная часть включает в себя большое количество модулей. Более подробно о мик­роядерных и макроядерныхоперационных системах см. далее.

Помимо программных модулей, входящих в состав ядра и постоянно располагаю­щихся в оперативной памяти, может быть много других системных программных модулей, которые получают название транзитных. Транзитные программные мо­дули загружаются в оперативную память только при необходимости и в случае отсутствия свободного пространства могут быть замещены другими транзитными модулями. В качестве синонима термина «транзитный» можно использовать тер­мин «диск-резидентный».

# 1.2. Принцип особого режима работы.

Ядро операционной системы и низкоуровневые драйверы, управляющие работой каналов и устройств ввода-вывода, должны работать в специальном режиме рабо­ты процессора. Это необходимо по нескольким причинам. Во-первых, введение специального режима работы процессора, в котором должен исполняться только код операционной системы, позволяет существенно повысить надежность выпол­нения вычислений. Это касается выполнения как управляющих функций самой операционной системы, так и прикладных задач пользователей. Категорически нельзя допускать, чтобы какая-нибудь прикладная программа могла вмешиваться (преднамеренно или в связи с появлением ошибок вычислений) в вычисления, связанные с супервизорной частью операционной системы. Во-вторых, ряд функ­ций должен выполняться исключительно централизованно, под управлением опе­рационной системы. К этим функциям мы, прежде всего, должны отнести функции, связанные с управлением процессами ввода-вывода данных. Вспомните основные принципы организации ввода-вывода (см. главу 5): *все операции ввода-вывода дан­ных объявляются привилегированными.*Это легче всего сделать, если процессор может работать, как минимум, в двух режимах: привилегированном (режим су­первизора) и пользовательском. В первом режиме процессор может выполнять все команды, тогда как в пользовательском набор разрешенных команд ограничен. Естественно, что помимо запрета на выполнение команд ввода-вывода в пользовательском режиме работы процессор не должен позволять обращаться к своим спе­циальным системным регистрам — эти регистры должны быть доступны только в привилегированном режиме, то есть исключительно супервизорному коду самой операционной системы. Попытка выполнить запрещенную команду или обратиться к запрещенному регистру должна вызывать прерывание (исключение), и центральный процессор должен быть предоставлен супервизорной части операционной системы для управления выполняющимися вычислениями.

Поскольку любая программа требует операций ввода-вывода, прикладные nporpaм мы для выполнения этих (и некоторых других) операций обращаются к суперви зорной части операционной системы (модуль супервизора иногда называют *су первизором задач)*с соответствующим *запросом.*При этом процессор должен переключиться в привилегированный режим работы. Чтобы программы не могли произвольным образом обращаться к супервизорному коду, который работает в привилегированном режиме, им предоставляется возможность обращаться к ним в строгом соответствии с принятыми правилами. Каждый запрос имеет свой идентификатор и должен сопровождаться соответствующим количеством параметров уточняющих запрашиваемую у операционной системы функцию (операцию). По этому супервизор задач при получении запроса сначала его тщательно проверяет Если запрос корректный и программа имеет право с ним обращаться, то запрос на выполнение операции, как правило, передается соответствующему модулю опера ционной системы. Множество запросов к операционной системе образует соответствующий системный *интерфейс прикладного программирования*(Application Program Interface, API).

# 1.3. Принцип виртуализации.

В наше время уже не требуется пояснять значение слова «виртуальный», ибо о виртуальных мирах, о виртуальной реальности знают даже дети. Принцип виртуализации нынче используется практически в любой операционной системе. Виртуализация ресурсов позволяет не только организовать разделение тех ресурсов между вычислительными процессами, которые не должны разделяться. Виртуализация позволяет абстрагироваться от конкретных ресурсов, максимально обобщить и: свойства и работать с некоторой абстракцией, вобравшей в себя наиболее значимые особенности. Этот принцип позволяет представить структуру системы в виде определенного набора планировщиков процессов и распределителей ресурсов (мониторов) и использовать единую централизованную схему распределения ресурсов.

Следует заметить, что сама операционная система существенно изменяет наши Представления о компьютере. Она виртуализирует его, добавляя ему функциональ­ности, удобства управления, предоставляя средства организации параллельных вычислений и т. д. Именно благодаря операционной системе мы воспринимаем компьютер совершенно иначе, чем без нее.

Наиболее законченным и естественным проявлением концепции виртуальности является понятие *виртуальной машины.*По сути, любая операционная система являясь средством распределения ресурсов и организуя по определенным прави­лам управление процессами, скрывает от пользователя и его приложений реальные аппаратные и иные ресурсы, заменяя их некоторой абстракцией. В результате пользователи видят и используют виртуальную машину как некое устройство, способное воспринимать их программы, написанные на определенном языке программирования, выполнять их и выдавать результаты на виртуальные устройства которые связаны с реально существующими в данной вычислительной системе. При таком языковом представлении пользователя совершенно не интересует, реальна конфигурация вычислительной системы, способы эффективного использования ее компонентов и подсистем. Он мыслит и работает с машиной в терминах используемого им языка.

Чаще виртуальная машина, предоставляемая пользователю, воспроизводит архи­тектуру реальной машины, но архитектурные элементы в таком представлении выступают с новыми или улучшенными характеристиками, часто упрощающими работу с системой. Характеристики могут быть произвольными, но чаще всего пользователи желают иметь собственную «идеализированную» по архитектурным характеристикам машину в следующем составе.

□ Единообразная по логике работы память (виртуальная) достаточного для вы­полнения приложений объема. Организация работы с информацией в такой памяти производится в терминах работы с сегментами данных на уровне вы­бранного пользователем языка программирования.

□  Произвольное количество процессоров (виртуальных), способных работать параллельно и взаимодействовать во время работы. Способы управления про­цессорами, в том числе синхронизация и информационные взаимодействия, реализованы и доступны пользователям с уровня используемого языка в тер­минах управления процессами.

□  Произвольное количество внешних устройств (виртуальных), способных ра­ботать с памятью виртуальной машины параллельно или последовательно, асин­хронно или синхронно по отношению к работе того или иного виртуального процессора, которые инициируют работу этих устройств. Информация, пере­даваемая или хранимая на виртуальных устройствах, не ограничена допусти­мыми размерами. Доступ к такой информации осуществляется на основе либо последовательного, либо прямого способа доступа в терминах соответствую­щей системы управления файлами. Предусмотрено расширение информаци­онных структур данных, хранимых на виртуальных устройствах.

Степень приближения к «идеальной» виртуальной машине может быть большей или меньшей в каждом конкретном случае. Чем больше виртуальная машина, реа­лизуемая средствами операционной системы на базе конкретной аппаратуры ком­пьютера, приближена к «идеальной» по характеристикам машине и, следователь­но, чем больше ее архитектурно-логические характеристики отличны от реально существующих, тем больше степень ее виртуальности.

Одним из важнейших результатов принципа виртуализации является возможность организации выполнения в операционной системе приложений, разработанных для другой операционной системы, имеющей совсем другой интерфейс прикладного программирования. Другими словами, речь идет об организации нескольких операционных сред, о чем мы уже говорили в главе 1. Реализация этого принципа позволяет операционной системе иметь очень сильное преимущество перед другими операционными системами, не имеющими такой возможности. Примером реализации принципа виртуализации может служить VDM-машина (Virtual DOS Machine) — защищенная подсистема, предоставляющая полную среду типа MS DOS консоль для выполнения DOS-приложений. Как правило, параллельно может выполняться практически произвольное число DOS-приложений, каждое в своей VDM-машине. Такие VDM-машины имеются и в операционных системах Win­dows компании Microsoft, в OS/2, в Linux.

Одним из аспектов общего принципа виртуализации является независимость про­грамм от внешних устройств, хотя иногда эту особенность выделяют особенно и на­зывают принципом. Она заключается в том, что связь программ с конкретными устройствами производится не в процессе создания программы, а в период плани­рования ее исполнения. В результате перекомпиляция при работе программы с но­вым устройством, на котором располагаются данные, не требуется. Этот принцип позволяет одинаково осуществлять операции управления внешними устройствами независимо от их конкретных физических характеристик. Например, программе, содержащей операции обработки последовательного набора данных, безразлично, на каком носителе эти данные будут располагаться. Смена носителя и данных, раз­мещаемых на них (при неизменности структурных характеристик данных), не при­внесет каких-либо изменений в программу, если в системе реализован принцип независимости программ от внешних устройств. Независимость программ от вне­шних устройств реализуется в подавляющем большинстве операционных систем общего применения. Ярким примером такого подхода являются операционные си­стемы с общим названием UNIX. Реализована такая независимость и в большин­стве современных операционных систем для персональных компьютеров. Например, в системах Windows все аппаратные ресурсы полностью виртуализи-рованы, и прямой доступ к ним со стороны прикладных (и системных обрабатыва­ющих) программ однозначно запрещен. В системах WindowsNT/2000/XP даже были введены понятия *HAL*(Hardware Abstraction Layer — уровень абстрагирова­ния аппаратуры) и *HEL*(Hardware Emulation Layer — уровень эмуляции аппара­туры), и этот шаг очень помогает в реализации идей переносимости (мобильнос­ти) операционной системы.

# 1.4.  Принцип мобильности.

Мобильность, или переносимость, означает возможность и легкость переноса опе­рационной системы на другую аппаратную платформу. Мобильная операционная система обычно разрабатывается с помощью специального языка высокого уров­ня, предназначенного для создания системного программного обеспечения. Такой язык помимо поддержки высокоуровневых операторов, типов данных и модуль­ных конструкций должен позволять непосредственно использовать аппаратные возможности и особенности процессора. Кроме этого, такой язык должен быть широко распространенным и реализованным в виде систем программирования, которые либо уже имеются на целевой платформе, либо позволяют получать программные коды для целевого компьютера. Другими словами, этот язык системно го программирования должен быть достаточно распространенным и технологи ным. Одним из таких языков является язык С, в последние годы язык C++ также стал использоваться для этих целей, поскольку идеи объектно-ориентированного программирования оказались плодотворными не только для прикладного, но и для системного программирования. Большинство современных операционных систем были созданы именно как объектно-ориентированные.

Обеспечить переносимость операционной системы достаточно сложно. Дело в том, что архитектуры разных процессоров могут очень сильно различаться. У них мо­жет быть разное количество рабочих регистров, причем часть регистров может оказаться контекстно-зависимыми, как это имеет место в процессорах с архи­тектурой ia32. Различия могут быть и в реализации адресации. Более того, для операционной системы важной является не только архитектура центрального процессора, но и архитектура компьютера в целом, ибо важнейшую роль играет подсистема ввода-вывода, а она строится на дополнительных (по отношению к цен­тральному процессору) аппаратных средствах. В таких условиях сделать эффек­тивным код операционной системы при условии создания его на языке типа C/C++ невозможно. Поэтому часть программных модулей, которые более всего зависят от аппаратных особенностей процессора, от типов поддерживаемых данных, спо­собов адресации, системы команд и других важнейших моментов, разрабатывает­ся на языке ассемблера. Очевидно, что модули, написанные на языке ассемблера, при переносе операционной системы на процессор с иной архитектурой должны быть написаны заново. Зато остальная (большая) часть кода операционной систе­мы может быть просто перекомпилирована под целевой процессор. Именно по это­му принципу в свое время была создана операционная система UNIX. Относи­тельная легкость переноса этой системы на другие компьютеры позволила сделать ее одной из самых распространенных. Для обеспечения мобильности был даже создан стандарт на интерфейс прикладного программирования, названный POSIX (Portable Operating System Interface for Computer Environments — интерфейс при­кладного программирования для переносимых операционных систем). К сожалению, на самом деле далеко не все операционные системы семейства UNIX допускают относительно простую переносимость созданного для них программ­ного обеспечения, хотя сами они и поддерживают такую переносимость. Основ­ная причина тому — отход от единого стандарта API — POSIX. Очевидно, что пла­той за универсальность, прежде всего, является потеря производительности при выполнении операций ввода-вывода и вычислений, связанных с этими операция­ми. Поэтому ряд разработчиков шли и до сих пор идут на отказ от принципа мо­бильности, поскольку не всегда следование этому принципу экономически оправ­дано.

Если при разработке операционной системы сразу не следовать принципу мобиль­ности, то в последующем очень трудно обеспечить перенос на другую платформу как самой операционной системы, так и программного обеспечения, созданного для нее. Например, компания IBM потратила долгие годы на перенос своей операционной системы OS/2, созданной для персональных компьютеров с процессора

архитектуры ia32, на платформу PowerPC. Но даже если изначально в спецификации на операционную систему заложить требование легкой переносимости это не значит, что его в последующем будет просто реализовать. Подтверждение *тому*является тот же проект OS/2-Windows NT. Как известно, проект Windows NT обеспечивал работу этой операционной системы на процессорах с архитектурой ia32, MIPS, Alpha (DEC), PowerPC. Однако в последующем трудности с реализацией этого принципа привели к тому, что нынешние версии операционных систем класса Windows NT (Windows 2000/XP) уже создаются только для процессоров с архитектурой 32 и не поддерживают MIPS, Alpha и PowerPC.

# 1.5. Принцип совместимости.

Одним из аспектов совместимости является способность операционной систем выполнять программы, написанные для других систем или для более ранних версий данной операционной системы, а также для другой аппаратной платформы. Необходимо разделять вопросы двоичной совместимости и совместимости на уровне исходных текстов приложений. Двоичная совместимость достигается в том случае, когда можно взять исполняемую программу и запустить ее на выполнение в другой операционной системе. Для этого необходимы: совместимость на уровне команд процессора, совместимость на уровне системных вызовов и даже на уровне библиотечных вызовов, если они являются динамически связываемыми. Совместимость на уровне исходных текстов требует наличия соответствующего транслятора в составе системного программного обеспечения, а также совместимости на уровне библиотек и системных вызовов. При этом необходима перекомпиляция имеющихся исходных текстов в новый выполняемый модуль. Гораздо сложнее достичь двоичной совместимости между процессорами, основанными на разных архитектурах. Для того чтобы один компьютер выполнял прс граммы другого (например, программу для персонального компьютера типа IBP PC хочется выполнять на компьютере типа Маc от фирмы Apple), этот компьютер должен работать с машинными командами, которые ему изначально непонятны. Например, процессор типа PowerPC на Mac должен исполнять двоичный код, предназначенный для процессора i80x86. Процессор 80x86 имеет свои собственные де шифратор команд, регистры и внутреннюю архитектуру. Процессор PowerPC имеет другую архитектуру, он не понимает непосредственно двоичный код 80x86, поэтому должен выбрать каждую команду, декодировать ее, чтобы определить, для чего она предназначена, а затем выполнить эквивалентную подпрограмму, написанную для PowerPC. К тому же у PowerPC нет в точности таких же регистров, флагов внутреннего арифметико-логического устройства, как в 80x86, поэтому он долже эмулировать все эти элементы с использованием своих регистров или памяти. И он должен тщательно воспроизводить результаты каждой команды, что требует специально написанных подпрограмм для PowerPC, гарантирующих, что состояние эмулируемых регистров и флагов после выполнения каждой команды будет в точности таким же, как и на реальном процессоре 80x86. Выходом в таких случаях является использование так называемых прикладных сред, или эмуляторов. Учитывая, что основную часть программы, как правило, составляют *вызовы библиотечных функций,*прикладная среда имитирует библиотечные функции целиком, используя заранее написанную библиотеку функций, аналогичного назначения остальные команды эмулирует каждую по отдельности.

Одним из средств обеспечения совместимости программных и пользовательски интерфейсов является соответствие стандартам POSIX. Эти стандарты позволяют создавать программы в стиле UNIX, которые впоследствии могут легко пере­носиться из одной системы в другую.

# 1.6. Принцип генерируемости.

Согласно принципу генерируемости исходное представление центральной систем­ной управляющей части операционной системы (ее ядра и основных компонен­тов, которые должны постоянно находиться в оперативной памяти) должно обеспе­чивать возможность настройки, исходя из конкретной конфигурации конкретного вычислительного комплекса и круга решаемых задач. Под генерацией операцион­ной системы понимается ее сборка (компоновка) из отдельных программных мо­дулей. В результате генерации получают скомпонованные двоичные коды опера­ционной системы и построенные системные таблицы, отражающие конкретную конфигурацию компьютера. Эта процедура проводится редко перед достаточно протяженным периодом эксплуатации операционной системы. Процесс генерации осуществляется с помощью специальной программы-генератора и соответствую­щего входного языка для этой программы, позволяющего описывать программ­ные возможности системы и конфигурацию машины. В результате генерации получается полная версия операционной системы. Сгенерированная версия опе­рационной системы представляет собой совокупность системных наборов моду­лей и данных.

Упомянутый раньше принцип модульности положительно проявляется при гене­рации операционной системы. Он существенно упрощает ее настройку на требуе­мую конфигурацию вычислительной системы. В наши дни при использовании персональных компьютеров с принципом генерируемости операционной системы можно столкнуться разве что при работе с Linux. В этой UNIX-системе имеется возможность не только использовать какое-либо готовое ядро операционной сис­темы, но и самому сгенерировать (скомпилировать) такое ядро, которое будет оп­тимальным для данного конкретного персонального компьютера и решаемых на нем задач. Кроме генерации ядра в Linux имеется возможность указать и набор подгружаемых драйверов и служб, то есть, часть функций может реализовываться модулями, непосредственно входящими в ядро системы, а часть — модулями, име­ющими статус подгружаемых, транзитных.

В остальных современных распространенных операционных системах, в том чис­ле и для персональных компьютеров, конфигурирование системы под соответству­ющий состав оборудования осуществляется на этапе установки, причем в боль­шинстве случаев не представляется возможным серьезно вмешаться в этот процесс-В дальнейшем, при эксплуатации компьютера, можно изменить состав драйверов, служб, отдельных параметров и режимов работы. Как правило, внесение подоб­ных изменений может быть осуществлено посредством редактирования конфигурационного файла или реестра. Например, мы можем отключить ненужное устройство, заменить для какого-нибудь устройства драйвер, отключить или добавить ту или иную службу. Более того, для большей гибкости часто вводится механизм поддержки нескольких конфигураций. Например, такие популярные системы, *как*Windows 98 и Windows NT/2000/XP, предоставляют возможность создавать до девяти конфигураций. При загрузке операционной системы пользователю пре­доставляется возможность выбрать одну из имеющихся конфигураций. Таким образом, имея всего одну операционную систему, за счет нескольких различающих­ся конфигураций пользователь может получить несколько виртуальных систем, различающихся составом установленного (работающего) оборудования, драйве­ров и служб, и на выбор запускать одну из этих систем.

# 1.7. Принцип открытости.

Открытая операционная система доступна для анализа, как пользователям, так и си­стемным специалистам, обслуживающим вычислительную систему. Наращивае­мая (модифицируемая, развиваемая) операционная система позволяет не только использовать возможности генерации, но и вводить в ее состав новые модули, со­вершенствовать существующие и т. д. Другими словами, необходимо, чтобы мож­но было легко внести дополнения и изменения, если это потребуется, не нарушая целостности системы. Прекрасные возможности для расширения предоставляет подход к структурированию операционной системы по типу клиент-сервер с ис­пользованием микроядерной технологии. В соответствии с этим подходом опера­ционная система строится как совокупность привилегированной управляющей программы и набора непривилегированных служб — «серверов». Основная часть операционной системы может оставаться неизменной, в то время как добавляются новые службы или изменяются старые.

Этот принцип иногда трактуют как расширяемость системы. К открытым операционным системам, прежде всего, следует отнести UNIX-систе­мы и, естественно, системы Linux.

# 1.8. Принцип обеспечения безопасности вычислений.

Обеспечение безопасности при выполнении вычислений является желаемым свой­ством для любой многопользовательской системы. Правила безопасности опреде­ляют такие свойства, как защита ресурсов одного пользователя от других и уста­новление квот по ресурсам для предотвращения захвата одним пользователем всех системных ресурсов (таких как память).

Обеспечение защиты информации от несанкционированного доступа является обя­зательной функцией многих операционных систем. Для решения этой проблемы чаще всего используется механизм учетных записей. Он предполагает проведение *аутентификации*пользователя при его регистрации на компьютере и последующую *авторизацию,*которая определяет уровень полномочий (прав) пользователя (об аутенти­фикации и авторизации пользователей см. главу 1). Каждая учетная запись может входить в одну или несколько групп. Встроенные группы, как правило, определяют права пользователей, тогда как создаваемые администратором группы (их называют группами безопасности) используются для определения разрешений в доступе пользователей к тем или иным ресурсам. Имеющиеся учетные записи хранятся в специальной базе данных, которая бывает доступна только для самой системы. Для этого файл базы данных с учетными записями открывается системой в монопольном режиме, доступ к нему со стороны любого пользователя становится невозможным. Делает это для того, чтобы нельзя было получить базу данных с учетными записями. Если получить файл с учетными записями, то посредством его анализа можно было бы узнать пароль пользователя, по которому осуществляется аутентификация. Во многих современных операционных системах гарантируется степень безопас­ности данных, соответствующая уровню С2 в системе стандартов США. Основы стандартов в области безопасности были заложены в документе «Критерии оцен­ки надежных компьютерных систем». Этот документ, изданный в США в 1983 году. Национальным центром компьютерной безопасности (National Computer Security Center), часто называют Оранжевой книгой.

В соответствии с требованиями Оранжевой книги безопасной считается система, которая «посредством специальных механизмов защиты контролирует доступ к ин­формации таким образом, что только имеющие соответствующие полномочия лица или процессы, выполняющиеся от их имени, могут получить доступ на чтение, за­пись, создание или удаление информации».

Иерархия уровней безопасности, приведенная в Оранжевой книге, помечает низ­ший уровень безопасности как D, а высший — как А.

В класс D попадают системы, оценка которых выявила их несоответствие требова­ниям всех других классов.

Основными свойствами, характерными для систем класса С, являются наличие подсистемы учета событий, связанных с безопасностью, и избирательный конт­роль доступа. Класс (уровень) С делится на два подуровня: уровень С1 обеспечи­вает защиту данных от ошибок пользователей, но не от действий злоумышленни­ков. На более строгом уровне С2 должны присутствовать:

□  средства секретного входа, обеспечивающие идентификацию пользователей путем ввода уникального имени и пароля перед тем, как им будет разрешен доступ к системе;

□  избирательный контроль доступа, требуемый на этом уровне, позволяет вла­дельцу ресурса определить, кто имеет доступ к ресурсу и что он может с ним делать, (владелец делает это путем предоставления разрешений доступа пользователю или группе пользователей);

□  средства аудита (auditing) обеспечивают обнаружение и запись важных со тий, связанных с безопасностью, или любых попыток создать системные ресурсы, получить доступ к ним или удалить их; □  защита памяти заключается в том, что память перед ее повторным использованием должна инициализироваться.

На этом уровне система не защищена от ошибок пользователя, но поведение его может быть проконтролировано по записям в журнале, оставленным средствами аудита.

Системы уровня В основаны на помеченных данных и распределении пользователей по категориям, то есть реализуют мандатный контроль доступа. Каждому пользователю присваивается рейтинг защиты, и он может получать доступ к данным только в соответствии с этим рейтингом. Этот уровень в отличие от уровня С защищает систему от ошибочного поведения пользователя.

Уровень А является самым высоким уровнем безопасности, он требует в дополне­ние ко всем требованиям уровня В выполнения формального математически обо­снованного доказательства соответствия системы требованиям безопасности. Различные коммерческие структуры (например, банки) особо выделяют необхо­димость учетной службы, аналогичной той, что предлагают государственные ре­комендации С2. Любая деятельность, связанная с безопасностью, может быть от­слежена и тем самым учтена. Это как раз то, что требует стандарт для систем класса С2 и что обычно нужно банкам. Однако коммерческие пользователи, как правило, не хотят расплачиваться производительностью за повышенный уровень безопасности. Уровень безопасности А занимает своими управляющими механиз­мами до 90 % процессорного времени, что, безусловно, в большинстве случаев не­приемлемо. Более безопасные системы не только снижают эффективность, но и существенно ограничивают число доступных прикладных пакетов, которые соот­ветствующим образом могут выполняться в подобной системе. Например, для опе­рационной системы Solaris (версия UNIX) есть несколько тысяч приложений, а для ее аналога уровня В — только около ста.

# 2. Контрольные вопросы.

1.      Основные  принципы построения  операционных  систем.

2.      Принцип  модульности.

3.      Принцип  виртуализации.

4.      Принцип  мобильности.

5.      Принцип открытости.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования/Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №2

1. Тема самостоятельной работы: *Распределение и использование ресурсов в операционной системы*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 1.1 Общие сведения об операционных системах», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

**Оптимальное распределение ресурсов**- такое распределение ресурсов, которое обеспечивает наилучшее, наиболее эффективное их использование. Основой оптимального распределения ресурсов является их ограниченность, что требует их использования (соответственно распределения) с учетом критерия оптимальности. Проблема оптимального распределения ресурсов решается с помощью экономико-математических моделей (линейного и нелинейного программирования и т. д.). При этом все экономико-математические модели направлены на то, чтобы обеспечить минимум затрат либо максимум эффекта при ограничениях по объему ресурсов и потребности в них.

К задачам оптимального распределения относятся такие широко распространенные задачи, как транспортная задача линейного программирования, задача о назначениях и др.

**Взаимоблокировка процессов**может происходить, когда несколько процессов борются за один ресурс.

Ресурсы бывают выгружаемые и невыгружаемые, аппаратные и программные.

**Выгружаемый ресурс**- этот ресурс безболезненно можно забрать у процесса (например: память).

**Невыгружаемый ресурс**- этот ресурс нельзя забрать у процесса без потери данных (например: принтер).

Проблема взаимоблокировок процессов возникает при борьбе за невыгружаемый ресурсы.

**Условия необходимые для взаимоблокировки:**

1. Условие взаимного исключения - в какой-то момент времени, ресурс занят только одним процессом или свободен.
2. Условие удержания и ожидания - процесс, удерживающий ресурс может запрашивать новые ресурсы.
3. Условие отсутствия принудительной выгрузки ресурса.
4. Условие циклического ожидания - должна существовать круговая последовательность из процессов, каждый, из которого ждет доступа к ресурсу, удерживаемому следующим членом последовательности.

**Виды ресурсов персонального компьютера:**

Аппаратные ресурсы (Hardware), файловые ресурсы, программные ресурсы (Software), сетевые ресурсы.

* Аппаратные ресурсы – это системный блок, периферийные устройства, любое оборудование, подключенное к компьютеру.
* Файловые ресурсы – это файлы и папки, а также вся файловая система.
* Программные ресурсы – это все программы установленные в компьютере. Часто называют программным обеспечением (ПО). Программное обеспечение подразделяется на два вида: системное и прикладное ПО.
* Сетевые ресурсы – ресурсы доступные по средствам ЛВС. Как правило, это ресурсы других компьютеров доступные по локальной или глобальной сети.

Сетевыми ресурсами могут быть:

* Оборудование (т.е. аппаратные ресурсы другого ПК или сетевые устройства), например сетевой принтер.
* Информация (т.е. файлы и папки другого компьютера), например информация в Интернете, или на сервере.
* Программное обеспечение (установленное на другом компьютере).

Операционная система не только предоставляет пользователям и программистам удобный интерфейс к аппаратным средствам ком­пьютера, но и является механизмом, распределяющим ресурсы ком­пьютера. К числу основных ресурсов современных вычислительных систем могут быть отнесены процессоры, основная память, таймеры, наборы данных, диски, накопители на магнитных лентах, принте­ры, сетевые устройства и некоторые другие. Ресурсы распределяют­ся между процессами (кроме процессоров, которые распределяются между потоками).

Управление ресурсами вычислительной системы с целью наибо­лее эффективного их использования является назначением операци­онной системы. Например, мультипрограммная операционная систе­ма организует одновременное выполнение сразу нескольких процес­сов на одном компьютере, поочерёдно переключая процессор с одно­го процесса на другой, исключая простои процессора, вызываемые обращениями процессов к вводу-выводу ОС также отслеживает и разрешает конфликты, возникающие при обращении нескольких про­цессов к одному и тому же устройству ввода-вывода или к одним и тем же данным.

Критерий эффективности, в соответствии с которым ОС орга­низует управление ресурсами компьютера, может быть различным. Например, в одних системах важен такой критерий, как пропускная способность вычислительной системы, в других — время её реак­ции. Соответственно выбранному критерию эффективности опера­ционные системы по-разному организуют вычислительный процесс.

Управление ресурсами включает решение следующих общих, не зависящих от ресурса задач:

- планирование ресурса, то есть определение, какому процессу, когда и в каком количестве (если ресурс может выделяться частями) следует выделить ресурс;

- удовлетворение запросов на ресурсы;

- отслеживание состояния и учёт использования ресурса, то есть поддержание оперативной информации о том, занят или свободен ресурс и какая доля ресурса уже распределена;

- разрешение конфликтов между процессами.

Задача организации эффективного совместного использования ресурсов несколькими процессами является весьма сложной, и слож­ность эта порождается в основном случайным характером возник­новения запросов на потребление ресурсов. В мультипрограммной системе образуются очереди заявок от одновременно выполняемых программ к разделяемым ресурсам компьютера: процессору, страни­цам памяти, к принтерам, к дискам. Операционная система органи­зует обслуживание этих очередей по разным алгоритмам: в порядке поступления, на основе приоритетов, кругового обслуживания и т. д.

**Контрольные вопросы.**

1.      Виды ресурсов.

2.      Оптимальное распределение ресурсов.

3.      Взаимоблокировка процессов.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №3

1. Тема самостоятельной работы: *История создания операционной системы*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы. Подготовить сообщение
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 1.1 Общие сведения об операционных системах», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

**Поколение №0:** 1740-ые первая механическая счетная машина. Электронные машины появились после появления триггеров. Некоторый прогресс в создании цифровых вычислительных машин произошел после второй мировой войны. 1944 в США, на них не было ОС. Устройства ввода/вывода – бумажные носители. В первых ВМ не было ОС, пользователи имели полный доступ к машинному языку, все программы писались непосредственно в машинных командах.

**Поколение №1:** 50-е годы. Основные направления: а) Автоматизация ввода/вывода (перфокарты, перфоленты) зарождение BIOS. б) Экономия времени (сателлит – малые машины для ввода/вывода рядом с большой вычислительной машиной). Появление операторов ЭВМ. Первая ОС – начало 50-х годов: General Motors для IBM 701. ОС были разработаны с целью ускорения и упрощения перехода с задачи на задачу.  Назначение ОС – обеспечение пакетного режима (пакетный обработкой, который предусматривал объединение отдельных задач в группы или пакеты). Выделение повторяющихся команд в отдельные процедуры. Создание stdio, стандартов ввода/вывода.

**Поколение №2:**60-е годы. Основные направления: а) Многозадачная пакетная логика. б) Многопользовательские режимы. в) Чисто многозадачный режим. Бурное развитие ОС и различных теорий ОС. Развиваются ОС пакетной обработки, многозадачные ОС, многопользовательские ОС, ОС с транзакциями. Появились методы, обеспечивающие независимость программирования от внешних устройств (ВУ). Появление многопроцессорных машин и соответственно ОС. Появление ОС реального времени. Пример глобальной сложной системы: Multics (проект провален). К большой машине стыковались по последовательному порту терминалы (символьные). К концу 60-х годов появление распределенных машин (сбор информаций из многих мест, например: сбор информации об авиабилетах).

**Поколение №3:** 70-е годы. Появление научной дисциплины проектирование программ, создание совместимых систем, создание эмуляторов, начало специализирования ОС. Разработка модульного процедурного программного обеспечения. Совместимость ОС и программных средств (обеспечения) впервые организовала фирма IBM. Было много фирм, их машины и системы команд были разными. Очень сложно было переходить к вычислительной машине другого класса. IBM создала линейку вычислительных машин совместимых снизу вверх: IBM System 360 и ОС, совместимую со всеми машинами этой линейки (OS/360), её аналог в СССР – ЕС-ЭВМ. Для облегчения перехода с других машин на эту линейку фирма IBM создала эмуляторы других систем на своей системе. (Аналогично в свое время поступит Win95, потребовала написания драйверов под эту систему). Позднее будут созданы DEC-PDP и ОС UNIX. Сложность проектирования отладки и поддержки ОС породила технологию конструирования программ – модульность, совместимость и мобильность ОС.

**Поколение №4:** 80-е годы. ОС ещё более функционально насыщены. ОС общего назначения становятся сетевыми. Распределенные вычислительные системы. Развиваются средства управления базами данных. Данные стали рассматриваться как объект управления. С точки зрения совместимости широко развивается концепция виртуальной машины. Удобство для пользователя (системы управления при помощи меню и так далее). Появление первых персональных компьютеров. Удобство программирования (интегрированная оболочка программирования). В 70-е годы появление языка C, языка высокого уровня. В 80-е годы C++ – объектно-ориентированный язык программирования. В эти годы произошло разделение производителей и появление большого количества независимых фирм, разрабатывающих ОС.

**Поколение №5:** 90-е годы. Принцип распределенной обработки данных перешел на принцип распределенных сетевых ОС, в которых основные функции реализуются на распределенных в пределах сети процессорах. Появление глобальной сети. Интерфейс становится более дружественным (интуитивно понятный интерфейс). Для программиста создание среды визуального программирования. Новый класс ОС, специализированных для процессоров ЦОС.

**Вывод:** ОС на современном этапе развиваются в направлении: 1) повышения функциональной сложности, насыщенности и производительности. 2) повышение степени абстрагирования от аппаратуры (мобильности). 3) повышение степени дружественности к оператору и распределенности обработки

**Контрольные вопросы.**

1. Поколения операционных систем.
2. Развитие современных ОС.
3. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

**Методические рекомендации (указания) к выполнению**

**самостоятельной работы №4**

1. Тема самостоятельной работы: *Понятие вычислительного процесса и ресурса.*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 1.1 Общие сведения об операционных системах», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

Понятие *«вычислительный процесс»* (или просто – *«процесс»*) является одним из основных при рассмотрении операционных систем. Как понятие процесс яв­ляется определенным видом абстракции, и мы будем придерживаться следующе­го неформального определения, приведенного в работе [37]. Последовательный процесс (иногда называемый *«задачей»*[[1]](#footnote-1)1) – это выполнение отдельной программы с её данными на последовательном процессоре. Концептуально процессор рас­сматривается в двух аспектах: во-первых, он является носителем данных и, во-вторых, он (одновременно) выполняет операции, связанные с их обработкой.

В качестве примеров можно назвать следующие процессы (задачи): прикладные программы пользователей, утилиты и другие системные обрабатывающие про­граммы. Процессами могут быть редактирование какого-либо текста, трансляция исходной программы, её компоновка, исполнение. Причем трансляция какой-ни­будь исходной программы является одним процессом, а трансляция следующей исходной программы – другим процессом, поскольку, хотя транслятор как объ­единение программных модулей здесь выступает как одна и та же программа, но данные, которые он обрабатывает, являются разными.

Определение концепции процесса преследует цель выработать механизмы распре­деления и управления ресурсами. Понятие ресурса, так же как и понятие процесса, является, пожалуй, основным при рассмотрении операционных систем. Термин *ресурс* обычно применяется по отношению к повторно используемым, относи­тельно стабильным и часто недостающим объектам, которые запрашиваются, ис­пользуются и освобождаются процессами в период их активности. Другими сло­вами, ресурсом называется всякий объект, который может распределяться внутри системы.

Ресурсы могут быть разделяемыми, когда несколько процессов могут их исполь­зовать одновременно (в один и тот же момент времени) или параллельно (в те­чение некоторого интервала времени процессы используют ресурс поперемен­но), а могут быть и неделимыми (рис. 1.1).

Ресурсы

Делимые

Неделимые

Используемые одновременно

Используемые параллельно

**Рис. 1.1.** Классификация ресурсов

При разработке первых систем ресурсами считались процессорное время, память, каналы ввода/вывода и периферийные устройства [49, 89]. Однако очень скоро понятие ресурса стало гораздо более универсальным и общим. Различного рода программные и информационные ресурсы также могут быть определены для сис­темы как объекты, которые могут разделяться и распределяться и доступ к кото­рым необходимо соответствующим образом контролировать. В настоящее время понятие ресурса превратилось в абстрактную структуру с целым рядом атрибу­тов, характеризующих способы доступа к этой структуре и её физическое пред­ставление в системе. Более того, помимо системных ресурсов, о которых мы сейчас говорили, как ресурс стали толковать и такие объекты, как сообщения и синхросигналы, которыми обмениваются задачи.

В первых вычислительных системах любая программа могла выполняться только после полного завершения предыдущей. Поскольку эти первые вычислительные системы были построены в соответствии с принципами, изложенными в извест­ной работе Яноша Джон фон Неймана, все подсистемы и устройства компьютера управлялись исключительно центральным процессором. Центральный процессор осуществлял и выполнение вычислений, и управление операциями ввода/выво­да данных. Соответственно, пока осуществлялся обмен данными между опера­тивной памятью и внешними устройствами, процессор не мог выполнять вычис­ления. Введение в состав вычислительной машины специальных контроллеров позволило совместить во времени (распараллелить) операции вывода получен­ных данных и последующие вычисления на центральном процессоре. Однако все равно процессор продолжал часто и долго простаивать, дожидаясь завершения очереднойоперации ввода/вывода. Поэтому было предложено организовать так называемый *мультипрограммный* (мультизадачный) режим работы вычислитель­ной системы. Суть его заключается в том, что пока одна программа (один вычис­лительный процесс или задача, как мы теперь говорим) ожидает завершения очереднойоперации ввода/вывода, другая программа (а точнее, другая задача) может быть поставлена на решение (рис. 1.2).

Из рис. 1.2, на котором в качестве примера изображена такая гипотетическая си­туация, видно, что благодаря совмещению во времени выполнения двух программ общее время выполнения двух задач получается меньше, чем если бы мы выпол­няли их по очерёди (запуск одной только после полного завершения другой). Из этого же рисунка видно, что время выполнения каждой задачи в общем слу­чае становится больше, чем если бы мы выполняли каждую из них как единст­венную.

При мультипрограммировании повышается пропускная способность системы, но отдельный процесс никогда не может быть выполнен быстрее, чем если бы он выполнялся в однопрограммном режиме (всякое разделение ресурсов замедляет работу одного из участников за счёт дополнительных затрат времени на ожида­ние освобождения ресурса).



**Рис. 1.2.** Пример выполнения двух программ: а – однопрограммный режим;

б – мультипрограммный режим

Как мы уже отмечали, операционная система поддерживает *мультипрограмми­рование* (многопроцессность) и старается эффективно использовать ресурсы пу­тём организации к ним очерёдей запросов, составляемых тем или иным способом. Это требование достигается поддерживанием в памяти более одного процесса, ожидающего процессор, и более одного процесса, готового использовать другие ресурсы, как только последние станут доступными. Общая схема выделения ре­сурсов такова. При необходимости использовать какой-либо ресурс (оператив­ную память, устройство ввода/вывода, массив данных и т. п.) задача обращается к супервизору операционной системы – её центральному управляющему моду­лю, который может состоять из нескольких модулей, например: супервизор вво­да/вывода, супервизор прерываний, супервизор программ, диспетчер задач и т. д. – посредством специальных вызовов (команд, директив) и сообщает о своём тре­бовании. При этом указывается вид ресурса и, если надо, его объём (например, количество адресуемых ячеек оперативной памяти, количество дорожек или сек­торов на системном диске, устройство печати и объём выводимых данных и т. п.).

Директива обращения к операционной системе передаёт ей управление, перево­дя процессор в привилегированный режим работы (см. раздел «Прерывания», глава 1), если такой существует. Не все вычислительные комплексы имеют два (и более) режима работы: привилегированный (режим супервизора), пользова­тельский, режим эмуляции какого-нибудь другого компьютера и т. д.

Ресурс может быть выделен задаче, обратившейся к супервизору с соответствую­щим запросом, если:

♦ он свободен и в системе нет запросов от задач более высокого приоритета к этому же ресурсу;

♦ текущий запрос и ранее выданные запросы допускают совместное использо­вание ресурсов;

♦ ресурс используется задачей низшего приоритета и может быть временно ото­бран (разделяемый ресурс).

Получив запрос, операционная система либо удовлетворяет его и возвращает управление задаче, выдавшей данный запрос, либо, если ресурс занят, ставит за­дачу в очередь к ресурсу, переводя её в состояние ожидания (блокируя). очередь к ресурсу может быть организована несколькими способами, но чаще всего это осуществляется с помощью списковой структуры.

После окончания работы с ресурсом задача опять с помощью специального вы­зова супервизора (посредством соответствующей директивы) сообщает операци­онной системе об отказе от ресурса, или операционная система забирает ресурс сама, если управление возвращается супервизору после выполнения какой-либо системной функции. Супервизор операционной системы, получив управление по этому обращению, освобождает ресурс и проверяет, имеется ли очередь к освободившемуся ресурсу. Если очередь есть – в зависимости от принятой *дисциплины обслуживания* (правила обслуживания)[[2]](#footnote-2)1 и приоритетов заявок он выводит из состояния ожидания задачу, ждущую ресурс, и переводит её в состояние го­товности к выполнению. После этого управление либо передаётся данной задаче, либо возвращается той, которая только что освободила ресурс.

При выдаче запроса на ресурс задача может указать, хочет ли она владеть ресур­сом монопольно или допускает совместное использование с другими задачами. Например, с файлом можно работать монопольно, а можно и совместно с други­ми задачами.

Если в системе имеется некоторая совокупность ресурсов, то управлять их ис­пользованием можно на основе определенной стратегии. Стратегия подразуме­вает четкую формулировку целей, следуя которым можно добиться эффективно­го распределения ресурсов.

При организации управления ресурсами всегда требуется принять решение о том, что в данной ситуации выгоднее: быстро обслуживать отдельные наиболее важ­ные запросы, предоставлять всем процессам равные возможности либо обслужи­вать максимально возможное количество процессов и наиболее полно использо­вать ресурсы

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

[[3]](#footnote-3)

### Методические рекомендации (указания) к выполнению

### самостоятельной работы №5

1. Тема самостоятельной работы: *Управление задачами и процессами. Планирование и диспетчеризация процессов и задач*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 2.1 Управление задачами и процессами», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

Оперативная память – это важнейший ресурс любой вычислительной системы, поскольку без неё (как, впрочем, и без центрального процессора) невозможно выполнение ни одной программы. Мы уже отмечали, что память является разделяемым ресурсом. От выбранных механизмов распределения памяти между выполняющимися процессорами очень сильно зависит и эффективность исполь­зования ресурсов системы, и её производительность, и возможности, которыми могут пользоваться программисты при создании своих программ. Способы рас­пределения времени центрального процессора тоже сильно влияют и на скорость выполнения отдельных вычислений, и на общую эффективность вычислитель­ной системы.

Понятие процесса (задачи) нам уже известно. В данной главе мы не будем стараться разделять понятия процесс (process) и поток (thread), вместо этого используя как бы обобщающий термин *task* (задача). В других разделах, если специально это не оговаривается, под задачей или процессом следует понимать практически одно и то же. Сейчас же мы будем говорить о разделении ресурса центрального процессора, поэтому термин задача может включать в себя и поня­тие треда (потока).

Итак, операционная система выполняет следующие основные функции, связан­ные с управлением задачами:

♦ создание и удаление задач;

♦ планирование процессов и диспетчеризация задач;

♦ синхронизация задач, обеспечение их средствами коммуникации.

Система управления задачами обеспечивает прохождение их через компьютер. В зависимости от состояния процесса ему должен быть предоставлен тот или иной ресурс. Например, новый процесс необходимо разместить в основной па­мяти – следовательно, ему необходимо выделить часть адресного пространства. Новый порожденный поток текущего процесса необходимо включить в общий список задач, конкурирующих между собой за ресурсы центрального процессора.

Создание и удаление задач осуществляется по соответствующим запросам от поль­зователей или от самих задач. Задача может породить новую задачу. При этом между процессами появляются «родственные» отношения. Порождающая задача называется «предком», «родителем», а порожденная – «потомком», «сыном» или «дочерней задачей». «Предок» может приостановить или удалить свою дочер­нюю задачу, тогда как «потомок» не может управлять «предком».

Основным подходом к организации того или иного метода управления процесса­ми, обеспечивающего эффективную загрузку ресурсов или выполнение каких-либо иных целей, является организация очередей процессов и ресурсов.

Очевидно, что на распределение ресурсов влияют конкретные потребности тех за­дач, которые должны выполняться параллельно. Другими словами, можно столк­нуться с ситуациями, когда невозможно эффективно распределять ресурсы с тем, чтобы они не простаивали. Например, всем выполняющимся процессам требует­ся некоторое устройство с последовательным доступом. Но поскольку, как мы уже знаем, оно не может распределяться между параллельно выполняющимися процессами, то процессы вынуждены будут очень долго ждать своей очерёди. Таким образом, недоступность одного ресурса может привести к тому, что дли­тельное время не будут использоваться и многие другие ресурсы.

Если же мы возьмем набор таких процессов, которые не будут конкурировать между собой за неразделяемые ресурсы при параллельном выполнении, то, ско­рее всего, процессы смогут выполниться быстрее (из-за отсутствия дополнитель­ных ожиданий), да и имеющиеся в системе ресурсы будут использоваться более эффективно. Итак, возникает задача подбора такого множества процессов, что при выполнении они будут как можно реже конфликтовать из-за имеющихся в системе ресурсов. Такая задача называется *планированием вычислительных про­цессов.*

Задача планирования процессов возникла очень давно – в первых пакетных ОС при планировании пакетов задач, которые должны были выполняться на компьютере и оптимально использовать его ресурсы. В настоящее время актуальность этой задачи не так велика. На первый план уже очень давно вышли задачи дина­мического (или краткосрочного) планирования, то есть текущего наиболее эф­фективного распределения ресурсов, возникающего практически при каждом со­бытии. Задачи динамического планирования стали называть *диспетчеризацией[[4]](#footnote-4)1 .*

Очевидно, что планирование осуществляется гораздо реже, чем задача текущего распределения ресурсов между уже выполняющимися процессами и потоками. Основное отличие между долгосрочным и краткосрочным планировщиками за­ключается в частоте запуска: краткосрочный планировщик, например, может за­пускаться каждые 30 или 100 мс, долгосрочный – один раз за несколько минут (или чаще; тут многое зависит от общей длительности решения заданий пользо­вателей).

Долгосрочный планировщик решает, какой из процессов, находящихся во вход­ной очереди, должен быть переведен в очередь готовых процессов в случае осво­бождения ресурсов памяти. Он выбирает процессы из входной очерёди с целью создания неоднородной мультипрограммной смеси. Это означает, что в очерёди готовых к выполнению процессов должны находиться – в разной пропорции – как процессы, ориентированные на ввод/вывод, так и процессы, ориентирован­ные на преимущественную работу с центральным процессором.

Краткосрочный планировщик решает, какая из задач, находящихся в очерёди го­товых к выполнению, должна быть передана на исполнение. В большинстве со­временных операционных систем, с которыми мы сталкиваемся, долгосрочный планировщик отсутствует.

Контрольные вопросы и задачи

* 1. Перечислите и поясните основные функции ОС, которые связаны с управлени­ем задачами.
  2. Какие дисциплины диспетчеризации задач вы знаете? Опишите их.
  3. Что такое «гарантия обслуживания»? Как её можно реализовать?
  4. Сравните механизмы диспетчеризации задач в ОС Windows NT и OS/2. В чём заключаются основные различия?
  5. Опишите механизм динамической диспетчеризации, реализованный в UNIX-системах.
  6. Что такое «виртуальный адрес», «виртуальное адресное пространство»? Чем (в общем случае) определяется максимально возможный объём виртуального адресного пространства программы?
  7. Что такое «фрагментация памяти»? Какой метод распределения памяти по­зволяет добиться минимальной фрагментации?
  8. Что такое «уплотнение памяти»? Когда оно применяется?
  9. Объясните сегментный способ организации виртуальной памяти. Что пред­ставляет собой (в общем случае) дескриптор сегмента?
  10. Сравните сегментный и страничный способы организации виртуальной памя­ти. Перечислите достоинства и недостатки каждого.
  11. Какие дисциплины применяются для решения задачи замещения страниц? Ка­кие из них являются наиболее эффективными и как они реализуются?
  12. Что такое «рабочее множество»? Что позволяет разрешить реализация этого понятия?
  13. Изложите принципы распределения памяти в MS-DOS.
  14. Что означает термин «плоская модель памяти»? В чём заключаются достоин­ства (и недостатки, если они есть) использования этой модели?

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования /Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №6

1. Тема самостоятельной работы: *Установка оборудования и драйверов для различных операционных систем*
2. Количество часов: 1ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 2.2 Управление памятью», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

Драйвер – набор служебных программ, позволяющих операционной системе (ОС) работать с тем или иным устройством компьютера. Его задача – обрабатывать запросы, поступающие от прикладных и системных программ, переводить их на язык, понятный физическому устройству, управлять процессами его инициализации, настройки параметров, обмена данными, переключением из одного состояния в другое и т.п. Драйвер позволяет операционной системе взаимодействовать с конкретным устройством через общий интерфейс, не учитывающий особенности данного устройства. Другими словами, драйвер транслирует запросы высокого уровня в запросы низкоуровневого машинного языка, непосредственно обращаясь к аппаратным ресурсам компьютера.

Не для каждого устройства требуется драйвер. Если существует строгий стандарт, описывающий набор команд, последовательность и временные параметры операций и другие особенности работы с данным классом устройств, драйвер может и не понадобиться, так как операционная система уже имеет в своем составе все необходимые для этого процедуры. В принципе, это можно назвать и встроенным драйвером. Примеры – клавиатура, таймер, коммуникационные порты, модем (внешний). Но если устройство может быть заменено на другое, отличное по своим функциональным возможностям, то драйвер для него нужно будет устанавливать обязательно.

Драйвер может содержаться и в дистрибутиве операционной системы. Тогда вопрос его поиска отпадает сам по себе. Однако устройства, появившиеся после официального выхода ОС, потребуют установки отдельного драйвера, разработанного производителем. Кроме того, набор драйверов в комплекте с ОС невелик и охватывает только небольшую часть наиболее распространенных или полностью стандартных устройств

**Драйверы и windows**

В операционных системах microsoft windows драйвер состоит из нескольких файлов, хранящихся обычно в каталогах **system**, **system32**и их подкаталогах. Ядро драйвера хранится в файлах с расширениями .**vxd**, .**drv**, .**sys**и некоторых других, а дополнительные процедуры собраны в динамические библиотеки **.dll**. Кроме того, в состав драйвера могут входить файлы справки, утилиты, модули деинсталляции и т.д

Последовательность операций по установке и удалению драйвера хранится в специальном информационном файле **.inf**. С его помощью windows определяет тип, производителя, модель устройства, класс драйвера, необходимые ресурсы и файлы. В этом файле также описываются операции распаковки, запуска, копирования, удаления, переименования файлов, добавления и удаления ключей в реестре и т.д. Все .inf-файлы хранятся в каталоге inf, причем устанавливаемые драйверы не-microsoft'овского происхождения (не поставляемые в комплекте с ОС) откладываются в отдельный подкаталог **inf/other**.

windows умеет автоматически находить драйвер для устройства. Для этого она использует технологию **plug&play**, точнее, ее часть, отвечающую за самоидентификацию устройства. В частности, pci-устройства обнаруживаются bios и заносятся в виде списка в специальную область **escd**(extended system configuration data). windows может использовать ее, а может и самостоятельно опросить шину pci и узнать у каждого устройства коды его производителя, модели и версии, необходимые ресурсы и другую информацию. Далее проверяется база данных (файлы **drvdata.bin**и **drvidx.bin**) по всем известным устройствам и находится необходимый .inf-файл. Если в каталоге .inf имеются новые файлы, они будут автоматически проиндексированы и внесены в базу данных.

Следует также помнить, что операционные системы windows 98 se и windows 2000, равно как и их потомки, поддерживают новую модель драйверов, получившую название wdm (windows driver model). Это – попытка реализовать полную поддержку **plug&play**и acpi, то есть дать возможность загружать и выгружать драйверы «на ходу», без перезагрузки системы, подключать их в виде фильтров-расширений к стандартным драйверам microsoft, более гибко управлять энергосбережением и конфигурацией устройств и т.п. wdm-драйверы хранятся в каталоге system32/drivers. В частности, интерфейсы нового поколения usb и ieee-1394 (firewire) работают только под управлением wdm-драйверов.

**«Референс» или «фирменные»?**

Как правило, функциональность каждого устройства компьютера определяется его контроллерами. Контроллеры имеют вид интегральных микросхем, установленных на печатной плате. Взаимодействие устройства с остальными компонентами системы сводится к обмену данными и командами между контроллером устройства и центральным процессором (или другим контроллером, например, арбитром шины, контроллером dma и т.п.). Можно сказать, что драйвер – это программа, позволяющая процессору «общаться» с контроллером

Очень часто разработчики микросхем (так называемого чипсета) сами пишут драйверы для того устройства, которым будет управлять их контроллер. Обычно такие драйверы называют референсными ("reference"). Они могут быть либо общедоступными и выкладываться в Интернет на сайте разработчика, либо предоставляться исключительно производителям оборудования на доработку и адаптацию. Очевидно, что в первом случае для устройства можно и нужно устанавливать референсные драйверы, которые обновляются гораздо чаще «фирменных» и содержат исправления ошибок и новые возможности. Однако при этом можно потерять доступ к некоторым функциям, специфичным для конкретного устройства.

«Фирменные» драйверы производителя оборудования могут быть изрядно устаревшими, но при этом учитывать какие-то особенности данного устройства, о которых не знают разработчики чипсета. В любом случае всегда имеет смысл попробовать сначала «референс»-драйвер (если он доступен для скачивания), а если он по какой-то причине не подойдет, вернуться к «фирменным».

**Версии драйверов**

Как и любое другое программное обеспечение, драйверы имеют свои версии. Формат версии свободный – каждый разработчик сам решает, сколько ему нужно цифр для нумерации и в какой последовательности они должны стоять. В некоторых случаях все-таки есть система: windows-драйверы для наиболее интересных с нашей точки зрения устройств вроде игровых видео- и звуковых карт, поддерживающих directx, нумеруются определенным образом. Первая цифра – номер версии операционной системы windows. 4 = win9x/winnt, 5 = win2000. Далее после точки идет версия directx. 12 = dx7, 13 = dx8. Последние цифры – номер версии самого драйвера. Например, представленный на скриншоте драйвер для видеокарт nvidia имеет версию 4.13.01.1241, что означает предназначен для win9x и directx8, его версия – 1241

Чтобы узнать версию драйвера, распакуйте его до состояния отдельных .**vxd**и **.dll**-файлов. Далее нужно в «Проводнике» щелкнуть любой из файлов (лучше с расширением **.vxd**) правой кнопкой мыши, выбрать «Свойства», потом закладку «Версия». Версия драйвера будет выделена цветом

Иногда производители сопровождают драйвер своим инсталлятором (setup.exe, install.exe и т.п.). Тогда увидеть файлы драйвера не представляется возможным. Поищите файл **readme.txt**, **version.txt**, **release.txt**или что-нибудь подобное, так как в них часто тоже указаны версии файлов или всего комплекта

Также стоит упомянуть о так называемых бета-драйверах. Разработчики чипсетов и устройств часто стараются стимулировать интерес к себе, выкладывая на своих сайтах экспериментальные драйверы. Другой известный прием – предоставление драйверов владельцам различных тематических Интернет-сайтов под видом украденных или случайно оставленных на внутренних корпоративных страницах (так называемые "leaked"-драйверы). Конечно, разработчики заранее открещиваются от них и не обещают технической поддержки в случае возникновения проблем, но, как мне кажется, они все равно анализируют поток писем от пользователей и тем самым экономят на процессе поиска сбоев и отладки

Чтобы предотвратить сбои в системе из-за бета-драйверов, microsoft ввела специальную сертификацию. Драйверы тестируются в лаборатории microsoft windows hardware quality lab (whql), и после успешного прохождения всех процедур они получают сертификат. Он имеет вид файла типа "security catalog" (расширение .cat), в котором перечислены все сертифицированные компоненты драйвера. Если такого файла в комплекте драйвера нет, то очень велика вероятность, что это бета-версия

Кстати, проверяйте **.cat-файлы**, запуская их двойным щелчком мыши, так как они могут быть фиктивными (пустыми).

**Как определить производителя и модель устройства**

Конечно, лучше покупать комплектующие известных производителей, благо сегодня такая возможность есть. Тогда не будет возникать вопрос о том, какой драйвер нужно искать. Но иногда пользователь не имеет понятия о модели или даже производителе устройства – например, если компьютер не очень новый и собирал его кто-то другой. Перед тем, как начинать поиск, придется определить точное название устройства или его чипсета.

Все чипы обязательно имеют маркировку, где указано название производителя, модель, серия, версия и ревизия чипа, дата выпуска. Микросхемы чипсета обычно имеют самое сложную структуру, у них большой корпус и много контактов, поэтому они больше других чипов по размерам. По маркировке, пользуясь поисковиками http://www.google.com/, http://www.altavista.com/, http://www.yandex.ru/ и другими, можно достаточно быстро отыскать сайт производителя чипсета и "reference"-драйверы.

Производителя самого устройства найти будет легче. Во-первых, вся необходимая информация должна быть нанесена краской на печатной плате. Даже если вы не нашли названия, есть шанс определить его по fcc-номеру. Ищите строку "fcc id:", в которой будет указан регистрационный код, присвоенный изготовителю устройства. После этого нужно зайти на сайт fcc и ввести код в строку поиска.

Во-вторых, где-нибудь обязательно будет написана модель устройства, по которой в Интернет можно попытаться найти сайт производителя.

В-третьих, если устройство устанавливается в слот pci или agp, то с помощью утилит можно определить код производителя этого устройства или чипсета. Могу порекомендовать sandra, powerstrip. Принцип, положенный в основу идентификации, базируется на анализе информации, выдаваемой pci-устройством. Как минимум можно получить коды производителя (vendor id) и модели (device id), которые поддаются расшифровке с помощью таблиц. В частности, в таблице sandra содержится более 6 тыс. кодов различных устройств

**Где взять драйвер**

Конечно, можно не напрягаться и установить все драйверы прямо с компакт-диска, идущего в комплекте с устройством. Такое решение вполне оправданно, но только на первое время. Драйверы (особенно для внутренних устройств) имеют тенденцию часто обновляться, поэтому полезно иногда поискать в Интернет новые версии. Иначе велика вероятность столкнуться с несовместимостью устройства с новым программным обеспечением или другим комплектующими.

Если вы точно знаете доменное имя сайта производителя (или догадываетесь – www.производитель.com, www.производитель.com.tw), то можно начинать поиск. На титульной странице отыщите кнопку "download", "drivers", "support" или аналогичную. Далее предстоит выбрать из списка именно то устройство, которое у вас имеется. Вполне возможно, что вы увидите несколько устройств с совпадающими названиями, отличия которых будут заключаться только в суффиксах или даже номерах ревизий. Обращайте на это внимание, иначе драйвер может не подойти. Далее, по возможности скачивайте драйверы не только для той операционной системы, которая у вас установлена, но и для других ОС: во время инсталляции новой windows будет неприятно обнаружить факт отсутствия необходимого драйвера.

Если вам никак не удается найти нужный сайт, обращайтесь на специализированные драйверные сайты (http://www.windrivers.com/, http://www.driverhq.com/, http://www.drv.ru/ и т.д.). Скачивать оттуда непосредственно файлы драйвера можно только в одном случае: фирма-разработчик уже прекратила свое существование и ее сайта физически нет в Сети. Гораздо лучше воспользоваться поиском не драйвера, а сайта производителя или его страницы технической поддержки.

**Как установить**

Установка драйвера заключается в следующем: windows обнаруживает .inf-файл, отыскивает в нем строки идентификации **plug&play**-устройств, и если они совпадают с информацией, выданной самим устройством, система выполняет предписанные действия по копированию файлов, добавлению записей в реестр и т.д. Вы должны указать местоположение файла в ответ на запрос мастера обновления оборудования. Мастер запускается либо автоматически, после появления сообщения о найденном устройстве, либо вручную. В последнем случае зайдите в «Свойства системы» (нажать я+pause), выбирайте закладку «Управление устройствами», потом свойства нужного устройства, закладку «Драйвер», «Обновить драйвер».

Если в комплекте с драйвером есть свой инсталлятор, то лучше воспользоваться им. Для этого нужно после появления сообщения о найденном устройстве нажать «esc», загрузить windows без драйвера и запустить инсталлятор. Он скопирует все файлы драйвера в папку windowsinfother, где драйвер и будет найден после перезагрузки. Кроме того, будут установлены дополнительные программы, идущие в комплекте. Немаловажно также и то, что в большинстве случаев инсталлятор может корректно убрать драйвер, не оставив следов своего пребывания («Панель управления»/«Установка и удаление программ»).

**Как удалить вручную**

Если установленный вами драйвер не справляется со своими функциями и даже вызывает ошибки и сбои, его нужно удалить. Просто нажать «del» в списке устройств, а потом дождаться сообщения о новом устройстве?.. Иногда это помогает, иногда – нет. Дело в том, что windows может следующий раз и не спросить у вас драйвер, а просто проинформировать о найденном устройстве и тут же вернуть все на место. Для удаления информации о драйвере потребуется дополнительно выйти в dos, зайти в каталог windows/inf и найти там .inf-файл нужного (точнее, уже ненужного) драйвера и удалить его. Вслед за ним можно отправить и .vxd-файлы, которые обычно находятся в windows/system, но тут нужно проявить аккуратность, так как можно и перестараться. Теперь после перезагрузки windows обновит свою базу данных и не найдет старого драйвера, а значит, можно устанавливать новый или возвращать на место старый, хорошо работавший до того, как вы начали ремонтировать то, что не сломалось.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №7

1. Тема самостоятельной работы: *Управление файлами. Основные понятия. Компоненты файловой системы.*
2. Количество часов: 1ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 2.2 Управление памятью», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

### Общая структура файловой системы

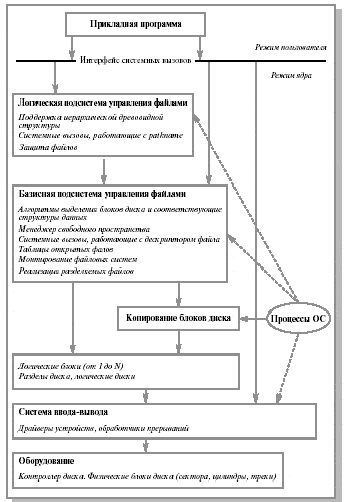
Система хранения данных на дисках может быть структурирована следующим образом ([рис.1](https://www.intuit.ru/studies/courses/2192/31/lecture/990?page=1#image.12.1)).

**Нижний уровень** - оборудование. Это в первую *очередь* *магнитные диски с подвижными головками* - основные *устройства внешней памяти*, представляющие собой пакеты магнитных пластин (поверхностей), между которыми на одном рычаге двигается пакет магнитных головок. Шаг движения пакета головок является дискретным, и каждому положению пакета головок логически соответствует цилиндр магнитного диска. Цилиндры делятся на дорожки (треки), а каждая дорожка размечается на одно и то же количество блоков (секторов) таким образом, что в каждый блок можно записать по максимуму одно и то же число байтов. Следовательно, для обмена с магнитным диском на уровне аппаратуры нужно указать номер цилиндра, номер поверхности, номер блока на соответствующей дорожке и число байтов, которое нужно записать или прочитать от начала этого блока. Таким образом, диски могут быть разбиты на блоки фиксированного размера и можно непосредственно получить *доступ* к любому блоку (организовать *прямой* *доступ* к файлам).

Непосредственно с устройствами (дисками) взаимодействует часть ОС, называемая **системой ввода-вывода** (см. лекцию 13). Система ввода-вывода предоставляет в распоряжение более высокоуровневого компонента ОС - файловой системы - используемое дисковое *пространство* в виде **непрерывной последовательности блоков фиксированного размера**. Система ввода-вывода имеет дело с **физическими** блоками диска, которые характеризуются адресом, например *диск* 2, цилиндр 75, сектор 11. *Файловая система* имеет дело с **логическими** блоками, каждый из которых имеет номер (от 0 или 1 до N). Размер логических блоков файла совпадает или является кратным размеру физического блока диска и может быть задан равным размеру страницы виртуальной памяти, поддерживаемой аппаратурой компьютера совместно с операционной системой.

В структуре *системы управления файлами* можно выделить базисную подсистему, которая отвечает за выделение дискового пространства конкретным файлам, и более высокоуровневую логическую подсистему, которая использует структуру дерева директорий для предоставления модулю базисной подсистемы необходимой ей информации, исходя из символического имени файла. Она также ответственна за авторизацию доступа к файлам (см. лекции 11 и 16).

**Стандартный запрос** на открытие (*open*) или создание (*create*) файла поступает от прикладной программы к логической подсистеме. Логическая подсистема, используя структуру директорий, проверяет *права доступа* и вызывает базовую подсистему для получения доступа к блокам файла. После этого *файл* считается открытым, он содержится в таблице открытых файлов, и *прикладная программа* получает в свое распоряжение *дескриптор* (или handle в системах Microsoft) этого файла. *Дескриптор* файла является ссылкой на *файл*в таблице открытых файлов и используется в запросах прикладной программы на чтение-*запись* из этого файла. *Запись* в таблице открытых файлов указывает через систему выделения блоков диска на блоки данного файла. Если к моменту открытия *файл* уже используется другим процессом, то есть содержится в таблице открытых файлов, то после проверки прав доступа к файлу может быть организован *совместный доступ*. При этом новому процессу также возвращается *дескриптор* - *ссылка* на *файл* в таблице открытых файлов. Далее в тексте подробно проанализирована работа наиболее важных системных вызовов.



**Рис. 1.**Блок-схема файловой системы

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №8

1. Тема самостоятельной работы: *Выполнение мероприятий для защиты компьютера от вирусных атак*
2. Количество часов: 1ч.
3. Задания для самостоятельной работы: ответить на контрольные вопросы
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 3.1 Защищенность и отказоустойчивость операционных систем», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

**Компьютерные вирусы** – специально написанные программы, способные самопроизвольно присоединяться к другим программам, создавать свои копии и внедрять их в файлы, системные области компьютера и в вычислительные сети с целью нарушения работы программ, порчи файлов и каталогов, создания всевозможных помех в работе компьютера. Могут быть разрушительными или проявляться в виде помехи, например:

* замена и/или удаление части или всего файла;
* форматирование диска;
* разрушение таблицы размещения файлов FAT);
* искажение сообщений программы пользователя и т. п.

Вирусы-помехи могут выводить на экран информацию, затрудняющую чтение сообщений программ. В настоящее время насчитывается несколько тысяч различных вирусов, и их количество продолжает возрастать. Например, только в глобальной сети Internet ежемесячно появляются не менее 200 вирусов.

## Способы распространения компьютерных вирусов

Возможные каналы проникновения вирусов в компьютер – накопители на сменных носителях информации, главным образом на дискетах, а также средства межкомпьютерной связи.

К последним относятся компьютерные сети, электронная почта, система BBS (Bulletin Board System – доска объявлений) и любая другая непосредственная связь между компьютерами.

Наиболее опасным является распространение вирусов по компьютерной сети, так как в этом случае за короткий промежуток времени может быть заражено большое количество компьютеров. Имеются даже специальные сетевые вирусы, предназначенные для функционирования в сетях.

При запуске инфицированной программы вирус старается отыскать незараженные программы и внедриться в них, а затем производит разрушительные действия.

## Классификация компьютерных вирусов

**Компьютерный вирус** – это программный код, встроенный в другую программу, в документ, или в определенные области носителя данных и предназначенный для выполнения несанкционированных действий на несущем компьютере.

Так, например, вирусный код может воспроизводить себя в теле других программ (этот процесс называется размножением). По прошествии определенного времени, создав достаточное количество копий, программный вирус может перейти к разрушительным действиям – нарушению работы программ и операционной системы, удалению информации, хранящейся на жестком диске и т.д. Этот процесс называется вирусной атакой.

Вирусы классифицируют по различным признакам.

### По среде обитания

* **Сетевые** вирусы распространяются по различным сетям, т.е. при передаче информации с одного компьютера на другой, соединенные между собой сетью, например Интернет.
* **Файловые** вирусы заражают исполнительные файлы и загружаются после запуска той программы, в которой он находится. Файловые вирусы могут внедряться и в другие файлы, но записанные в таких файлах, они не получают управление и теряют способность к размножению.
* **Загрузочные** вирусы внедряются в загрузочный сектор дискет или логических дисков, содержащий программу загрузки.
* **Файлово** – **загрузочные** вирусы заражают одновременно файлы и загрузочные сектора диска.

### По способу заражения среды обитания

* **Резидентный вирус** при заражении компьютера оставляет в оперативной памяти свою резидентную часть, которая потом перехватывает обращение операционной системы к объектам заражения (файлам, загрузочным секторам дисков и т.п.) и внедряется в них. Резидентные вирусы находятся в памяти и являются активными вплоть до выключения или перезагрузки компьютера.
* **Нерезидентный вирус** не заражает память компьютера и является активным ограниченное время. Активизируется в определенные моменты, например, при обработке документов текстовым редактором.

### По деструктивным (разрушительным) возможностям

* **Безвредные** вирусы проявляются только в том, что уменьшают объем памяти на диске в результате своего распространения.
* **Неопасные**, так же уменьшают объем памяти, не мешают работе компьютера, такие вирусы порождают графические, звуковые и другие эффекты.
* **Опасные вирусы**, которые могут привести к различным нарушениям в работе компьютера, например к зависанию или неправильной печати документа.
* **Очень опасные**, действие которых может привести к потере программ, данных, стиранию информации в системных областях памяти и даже приводить к выходу из строя движущихся частей жесткого диска при вводе в резонанс.

### По особенностям алгоритма

* **Паразитические** – это одни из самых простых вирусов. Они изменяют содержимое файлов и секторов диска и могут быть достаточно легко обнаружены и уничтожены.
* **Вирусы-репликаторы** (черви) распространяются по компьютерным сетям, вычисляют адреса сетевых компьютеров и записывают по этим адресам свои копии.
* **Вирусы невидимки** (стелс-вирусы) – вирусы, которые очень трудно обнаружить и обезвредить, так как они перехватывают обращения операционной системы к пораженным файлам и секторам дисков и подставляют вместо своего незараженные участки диска.
* **Мутанты** (призраки) содержат алгоритмы шифровки-расшифровки, благодаря которым копии одного и того же вируса не имеют ни одной повторяющейся цепочки байтов. Такие вирусы самые сложные в обнаружении.
* **Троянские программы** (квазивирусы) не способны к самораспространению, но очень опасны, так как, маскируясь под полезную программу, разрушают загрузочный сектор и файловую систему дисков.
* **Спутники** – вирус, который не изменяет файл, а для выполнимых программ (exe) создают одноименные программы типа com, которые при выполнении исходной программы запускаются первыми, а затем передают управление исходной выполняемой программе.
* **Студенческие вирусы** представляют собой самые простые и легко обнаруживаемые вирусы.

Однако четкого разделения между ними не существует, и все они могут составлять комбинацию вариантов взаимодействия – своеобразный вирусный «коктейль».

### Макровирусы

Эта особая разновидность вирусов поражает документы, выполненные в некоторых прикладных программах, имеющих средства для исполнения так называемых макрокоманд.

В частности, к таким документам относятся документы текстового процессора Microsoft Word. Заражение происходит при открытии файла документа в окне программы, если в ней не отключена возможность исполнения макрокоманд. Как и для других типов вирусов, результат атаки может быть как относительно безобидным, так и разрушительным.

## Защита от компьютерных вирусов

Компьютерный вирус аналогичен природному вирусу. Поэтому меры защиты от него включают в себя аналогичный комплекс средств:

* Профилактика;
* Диагностика;
* Лечение.

### Профилактика

К профилактическим средствам относятся:

* перекрытие путей проникновения вирусов в компьютер;
* исключение возможности заражения и порчи вирусами, проникшими в компьютер, других файлов.

### Диагностика

Диагностические средства позволяют обнаруживать вирусы в компьютере и распознавать их тип.

### Лечение

Лечение состоит в удалении вирусов из зараженных программных средств и восстановлении пораженных файлов.

Защитный комплекс основывается на применении антивирусных программ и проведении организационных мероприятий.

## Организационные мероприятия, производимые для защиты от компьютерных вирусов

Вирусы попадают в компьютер только вместе с программным обеспечением. Поэтому самым важным в защите от вирусов является использование незараженных программ, так как главным источником вирусов являются незаконные, так называемые «пиратские» копии программного обеспечения.

Особенно опасны компьютерные игры и различного рода развлекательные программы, которые чаще других являются разносчиками компьютерной инфекции. Поэтому первым и наиважнейшим правилом антивирусной защиты является следующее:

**Необходимо использовать только лицензионно-чистые** **программы от надежных поставщиков.**

## **Рекомендации**

* приобретайте все программы в фирменной упаковке у надежного поставщика;
* не пользуйтесь без крайней необходимостью чужими дискетами;
* не запускайте на выполнение программы, назначение которых неизвестно или непонятно;
* не передавайте свои дискеты чужим лицам для использования, чтобы не заразить ваши дискеты;
* ограничьте доступ к вашему ПК посторонних лиц и запретите им пользоваться своими дискетами без вашего разрешения;
* перед началом работы на ПК после другого лица осуществите холодный перезапуск ПК, чтобы удалить из ОЗУ возможно присутствующие там резидентные вирусы;
* при работе на одном ПК нескольких пользователей, разделите жесткий диск на несколько логических и разграничьте право доступа к различным дискам;
* включайте программы антивирусной защиты в файл AUTOEXEC. BAT;
* не ограничивайтесь использованием только одного антивирусного программного продукта. Новые вирусы появляются постоянно, и для их выявления требуются новые антивирусные программы;
* гибкие магнитные диски используйте, по возможности, с защитой от записи.

## Антивирусные программы

Программные средства антивирусной защиты обеспечивают диагностику (обнаружение) и лечение (нейтрализацию) вирусов.

### Краткий обзор антивирусных программ

При выборе антивирусной программы необходимо учитывать не только процент обнаружения вирусов, но и способность обнаруживать новые вирусы, количество вирусов в антивирусной базе, частоту ее обновления, наличие дополнительных функций.

В настоящее время серьезный антивирус должен уметь распознавать не менее 25000 вирусов. Это не значит, что все они находятся «на воле». На самом деле большинство из них или уже прекратили свое существование или находятся в лабораториях и не распространяются. Реально можно встретить 200-300 вирусов, а опасность представляют только несколько десятков из них.

Существует множество антивирусных программ. Рассмотрим наиболее известные из них.

**Программы-детекторы** обеспечивают поиск и обнаружение вирусов в оперативной памяти и на внешних носителях, и при обнаружении выдают соответствующее сообщение. Различают детекторы универсальные и специализированные.

* **Универсальные детекторы** в своей работе используют проверку неизменности файлов путем подсчета и сравнения с эталоном контрольной суммы. Недостаток универсальных детекторов связан с невозможностью определения причин искажения файлов.
* **Специализированные детекторы** выполняют поиск известных вирусов по их сигнатуре (повторяющемуся участку кода). Недостаток таких детекторов состоит в том, что они неспособны обнаруживать все известные вирусы. Детектор, позволяющий обнаруживать несколько вирусов, называют **полидетектором**. Недостатком таких антивирусных программ является то, что они могут находить только те вирусы, которые известны разработчикам таких программ.

**Программы-доктора** (фаги), не только находят зараженные вирусами файлы, но и «лечат» их, т.е. удаляют из файла тело программы вируса, возвращая файлы в исходное состояние. В начале своей работы фаги ищут вирусы в оперативной памяти, уничтожая их, и только затем переходят к «лечению» файлов. Среди фагов выделяют **полифаги**, т.е. программы-доктора, предназначенные для поиска и уничтожения большого количества вирусов.

Учитывая, что постоянно появляются новые вирусы, программы-детекторы и программы-доктора быстро устаревают, и требуется регулярное обновление их версий.

**Программы-ревизоры** относятся к самым надежным средствам защиты от вирусов. Ревизоры запоминают исходное состояние программ, каталогов и системных областей диска тогда, когда компьютер не заражен вирусом, а затем периодически или по желанию пользователя сравнивают текущее состояние с исходным. Обнаруженные изменения выводятся на экран видеомонитора. Как правило, сравнение состояний производят сразу после загрузки операционной системы. При сравнении проверяются длина файла, код циклического контроля (контрольная сумма файла), дата и время модификации, другие параметры.

Программы-ревизоры имеют достаточно развитые алгоритмы, обнаруживают стелс-вирусы и могут даже отличить изменения версии проверяемой программы от изменений, внесенных вирусом.

**Программы-фильтры** (сторожа) представляют собой небольшие резидентные программы, предназначенные для обнаружения подозрительных действий при работе компьютера, характерных для вирусов. Такими действиями могут являться:

* попытки коррекции файлов с расширениями СОМ и ЕХЕ;
* изменение атрибутов файлов;
* прямая запись на диск по абсолютному адресу;
* запись в загрузочные сектора диска;
* загрузка резидентной программы.

При попытке какой-либо программы произвести указанные действия «сторож» посылает пользователю сообщение и предлагает запретить или разрешить соответствующее действие. Программы-фильтры весьма полезны, так как способны обнаружить вирус на самой ранней стадии его существования до размножения.

Однако они не «лечат» файлы и диски. Для уничтожения вирусов требуется применить другие программы, например фаги. К недостаткам программ-сторожей можно отнести их «назойливость» (например, они постоянно выдают предупреждение о любой попытке копирования исполняемого файла), а также возможные конфликты с другим программным обеспечением.

Поскольку функции детектора, ревизора и сторожа дополняют друг друга, то в современные антивирусные комплекты программ обычно входят компоненты, реализующие все эти функции. При этом часто функции детектора и ревизора совмещаются в одной программе.

**Вакцины** (иммунизаторы) – это резидентные программы, предотвращающие заражение файлов. Вакцины применяют, если отсутствуют программы-доктора, «лечащие» этот вирус. Вакцинация возможна только от известных вирусов. Вакцина модифицирует программу или диск таким образом, чтобы это не отражалось на их работе, а вирус будет воспринимать их зараженными и поэтому не внедрится. В настоящее время программы-вакцины имеют ограниченное применение.

Существенным недостатком таких программ является их ограниченные возможности по предотвращению заражения от большого числа разнообразных вирусов.

Несмотря на все принятые профилактические меры, стопроцентной гарантии защиты от вирусов в настоящее время не существует. Поэтому в целях восстановления разрушенной вирусом информации и удаленных зараженных программ, которые не удалось вылечить программами антивирусной защиты, необходимо соблюдать еще одно правило антивирусной защиты:

**Всегда имейте резервные копии программ и файлов данных на** **дискете, магнитной ленте и/или другом ПК не менее чем в двух** **экземплярах.**

## Убытки

Несмотря на огромные усилия конкурирующих между собой антивирусных фирм, убытки, приносимые компьютерными вирусами, не падают и достигают астрономических величин в сотни миллионов долларов ежегодно. При этом следует иметь в виду, что антивирусные программы и «железо» не дают полной гарантии защиты от вирусов. Зачастую как пользователи, так и профессионалы-программисты не имеют достаточных навыков «самообороны», а их представления о вирусе порой являются весьма поверхностными.

Борьба с компьютерными вирусами является борьбой человека с человеческим же разумом. Эта борьба является борьбой умов, поскольку задачи, стоящие перед вирусологами, ставят такие же люди. Одни придумывают новый вирус – а другим с ним разбираться.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №9

1. Тема самостоятельной работы: *Эмуляторы операционных систем*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: подготовить презентацию
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 4.1 Принципы построения ОС. Особенности работы в конкретных ОС», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

**Эмуляция** ([англ.](https://ru.bmstu.wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *emulation*) – это совокупность логических и технических средств и ресурсов, направленных на полную имитацию технического устройства выбранной пользователем системы для максимально точного воспроизведения всех процессов, происходящих внутри эмулируемой системы.

**История**

Впервые о проблеме ускорения процесса симуляции заговорили в IBM в конце 50-х годов, когда в компании столкнулись с недостаточной производительностью при использовании программной симуляции в своих разработках, а также при интеграции в них программ, написанных для машин прошлого поколения. При разработке продуктов линейки IBM System 360 инженеры компании применили систему микрокода, которая показала увеличение производительности относительно систем, использовавших инструменты программной симуляции. В 1964 году IBM вводит понятие «эмуляции» для описания принципа микрокода для программного форсирования процесса симуляции.

**Различие процесса эмуляции и симуляции**

**Эмуляция** это всего лишь попытка зеркально сымитировать внутреннее устройство эмулируемой системы таким образом, чтобы программа, отвечающая за эмуляцию какой либо из систем в точности повторяла все ее процессы и работу компонентов эмулируемой системы.

**Симуляция**, в свою очередь, это попытка сымитировать конкретную функцию или часть устройства, не повторяя в точности ее процессы и работу ее отдельных компонентов для данной системы. Например, программа, имитирующая игровой автомат, которая выполняет родную для него программу ROM, будет являться эмулятором, а эта же программа, написанная для другой системы, но при этом не являющейся файлом ROM, который был предназначен только работы с той системой под которую он был изначально сделан, называется симулятор.

**Виды эмуляций**

Самый частый вид эмуляции - это вид, в котором воспроизводится только архитектура компьютера. Таким образом, если требуется операционная система, хранящаяся в ПЗУ, или другое программное обеспечение, его следует получить дополнительно (впрочем, оно тоже может быть эмулировано). В дальнейшем и операционная система и программное обеспечение будут интерпретированы эмулятором таким же образом, как и на изначальном оборудовании. Помимо интерпретаторов эмулированных двоичных машинных кодов требуется эмуляция других аппаратных составляющих. В идеальном случае, эмулятор должен копировать поведение оригинального схематического решения эмулируемой системы.  Как правило, эмуляторы отталкиваются от модели, построенной на имеющейся документации и логической схеме устройства. Но существуют такие эмуляции, где для систем важным оказывается высокая точность эмуляции вплоть до тактовой частоты отдельных элементов, недокументированных функций, непредсказуемых аналоговых компонентов и допущенных ошибок.

Стоит отметить существование устройств, которые имеют очень ограниченный прямой доступ к оборудованию. В их случаях достаточно простого слоя совместимости. Системные запросы эмулируемой программы транслируются в системные запросы хоста, то есть в системах FreeBSD, NetBSD и OpenBSD для запуска Linux -приложений с закрытым кодом используется слой совместимости с Linux. Например, графический процессор Nintendo 64 был полностью программируемым, и большинство разработчиков игр использовало заложенные заводские программы, которые были самодостаточными и обменивались информацией с игрой через буфер FIFO. Поэтому многие эмуляторы вообще не эмулируют графический процессор, интерпретируя вместо этого команды центрального процессора также как и оригинальная программа.

**Структура систем эмуляций**

**Эмулятор** – программа эмуляции, как правило, состоит из модулей, которые выполняют процессы, соответствующие процессам эмулируемой системы. Наиболее распространенная схема представляет собой:

* модуль эмуляции CPU;
* модуль эмуляции подсистемы памяти;
* модули эмуляции различных устройств ввода-вывода

**Центральный процессор** – самая трудоемкая часть создания эмулятора. Для упрощения процесса создания эмуляторов аще всего используют «готовые» модули CPU. Самая простая форма CPU – интерпретатор, программа отслеживающая поток выполнения программы и при встрече машинной инструкции выполняет операцию «языкового перевода» оригинальным инструкциям на процессоре хоста.

Интерпретаторы популярны при моделировании ЭВМ крайне низкой по сегодняшним меркам производительностью, однако этого бывает достаточно, учитывая простоту реализации таких способов эмуляции Для эмуляции устройств с сравнимой производительностью машины-хоста используется процесс «компиляции на лету», что позволило решить проблемы с самомодифицирующиесями кодами и отсутствием надежного способа разделения данных. Компилятор ожидает, пока поток управления процессором не перейдет в область, содержащую нетранслированный код. Только тогда («на лету») происходит трансляция блока кода в код, который может быть выполнен. Обработанный код помещается в кэш кода, при этом оригинальный код не подвергается изменению. В таком случае даже блоки данных подвергнутся бессмысленной трансляции компилятором, единственным эффектом чего будет увеличение времени работы транслятора.

**Эмуляция подсистемы памяти** чаще всего представляет из себя как минимум две процедуры – для чтения из памяти и для записи в нее, отвечающие за координаты нахождения правильного объекта. Такой подход связан с особенностью совместимости логического адреса и физической памятью при осуществлении процесса эмуляции.

Что касается, устройства **системы ввода-вывода** при эмуляции, то тут стоить отметить, что системные шины эмулируются очень редко, так в эмуляторе отсутствует система универсального интерфейса. В виду существования возможности идеально подогнать устройство ввода–вывода к параметрам эмулированного устройства значительно увеличивается производительность. Конкуренцию упрощенным моделям могут составить решения на базе унифицированных API. Их преимущество заключается в наборе плагинов, с помощью которых, с эмулятором могут работать сторонние устройства. Как правило, даже в самых простых эмуляторах предусмотрена виртуальная инфраструктура такого рода:

* управление прерываниями посредством процедуры, которая устанавливает флаги, считываемые эмулятором CPU, когда объявлено прерывание, что позволяет виртуальному CPU «опрашивать прерывания»
* запись и чтение физической памяти посредством двух процедур, подобных обслуживающим логическую память (однако, в отличие от последнего, первые часто могут быть заменены простыми ссылками на массив памяти).

**Отличительные особенности процессов виртуализации от эмуляции**

Виртуализатор исполняет код на том же процессоре, используя специальные режимы работы процессора. Ресурсы (память, диск и др.) предоставляет также напрямую или перехватывая обращения для «подсовывания» нужных данных. Иначе говоря, виртуализатор или совсем не эмулирует (имитирует) реальную машину, её архитектуру и процессор или делает это в минимальном варианте для отдельных ресурсов, для выполнения поставленной перед ним задачи.

Эмулятор - полностью или почти полностью реализует для исполнения кода отдельную машину со своей архитектурой и своими ресурсами. Вплоть до того, что может быть процессор совершенно другой архитектуры, к примеру ARM, Power или калькулятора МК-52. Так же необходимо отметить, что при эмуляции осуществляется изоляция процессов эмулируемой системы, для максимально точного исполнения поведения данной системы в пределах хост-машины и для защиты корректного поведения систем этой машины

Соответственно, преимущества виртуализатора в скорости исполнения кода, программа в "виртуалке" без специальной поддержки будет работать медленнее всего на 30-50%. При наличии поддержки виртуализации специальными командами процессора задержки могут быть совсем не большими по сравнению с нативным исполнением, разница в скорости может быть буквально несколько процентов.

**Оценка процесса эмуляции**

К плюсам можно отнести

* Возможность сохранения работы в одной технологической инфраструктуре при использовании разных библиотек приложений и операционных систем относительно их платформ создания и хронологии.
* Эмуляторы сокращают трудозатраты и оптимизируют возможность разработки кросс-платформенных приложений.

Из недостатков можно выделить следующее:

* Использование эмуляции в пиратских целях.
* Эмуляция способствует развитию сферы нарушений авторских прав и интеллектуальной собственности.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №10

1. Тема самостоятельной работы: *Стандартные программы ОС Windows 10*
2. Количество часов: 4ч.
3. Задания для самостоятельной работы: подготовить презентацию
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 4.2 Способы организации поддержки устройств. Стандартные программы операционной системы», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

Смартфоны и планшеты давно перестали быть вотчиной таких гигантов, как iOS и Android. После того, как мобильные гаджеты начали снабжать «атомными» Intel на борту, было вопросом времени, когда произойдет адаптация Windows 10 на карманные устройства. Сейчас на рынке просто неимоверное количество гаджетов, преимущественно планшетов, на «десятке».

Однако иногда возникает проблема с поиском качественного софта для них, учитывая то, что технические характеристики на порядок ниже настольных собратьев. Мы подготовили список из 12 нетребовательных приложений для Windows 10, которые будут одинаково полезны обладателям не только слабых ПК, но и владельцам планшетов.

**Nextpad**

Nextpad – это простой по функционалу редактор текста, который работает с большинством распространенных форматов. Эта программа предназначена только для планшетов или телефонов с Windows 10, так как разработана для использования на сенсорных экранах.

Помимо обычной редактуры файла, доступен голосовой ввод информации, воспроизведение набранного текста, опция «обмен контентом» (передача текста или всего документа другому приложению). В настройках вы по своему желанию можете изменить оформление. В конфигурациях интерфейса доступен только один язык – английский.

Программа находится в свободном доступе, но есть надоедливая реклама, которая отключается одним платежом примерно в $2,5.

**Code Writer**

В первую очередь эта программа – текстовый редактор, рассчитанный на программистов и специалистов из IT-области. Приложение работает и подсвечивает синтаксис таких языков, как Python, HTML, CSS, PHP, XML, Ruby и так далее. Для некоторых языков работают функции автокоррекции, поиска ошибок синтаксиса и тому подобное.

В приложении очень удобно реализовано форматирование кода: дублирование определенной строки, присвоение комментариев, возможность менять регистр отдельно взятых кусков кода и прочее.

Программа легко работает одновременно сразу с несколькими кодами из разных документов. А если вы ненадолго отрываетесь от работы, то все измененные файлы отмечаются.

Весьма гибкие настройки позволят вам откорректировать приложение под себя. Надо выделить пробелы? Пожалуйста: функция выделения пробелов и текущей строки. Нужно поменять кодировку или отключить неиспользуемые функции? Тоже без проблем.

Программа находится в свободном доступе (внутренние покупки также отсутствуют), но для того, чтобы работать с этим редактором, на вашем устройстве должна быть Windows 10 версии Fall Creators Update.

**GIMP**

Когда разговор заходит про обработку фотографий, на ум сразу приходит Photoshop. Да, его функционал огромен, но он слишком большой, громоздкий, требовательный к «железу». А ведь за него ещё и платить надо. На помощь придет GIMP.

Программа запустится даже на слабом планшете. Хватит 512 Mb RAM, 300 Mb ROM и 1,2 ГГц процессора. А про смешное требование в 1024х768 точек для дисплея и вспоминать нечего. Однако нужен постоянный доступ к интернету.

GIMP поддерживает все популярные форматы изображений (в том числе PDF и PSD), имеет в своем арсенале множество инструментов для обработки, работы со слоями, анимированными файлами и многое другое. Несколько непривычен многооконный интерфейс, но дискомфорт длится всего полчаса, потом от него нет и следа. Также присутствует русская локализация.

**LibreOffice**

Большинство обычных пользователей привыкло при работе с различной документацией (презентации, текстовые файлы, таблицы и прочее) пользоваться Microsoft Office. Бесспорно, это очень хороший продукт, но с ним не всё так просто. Офисный пакет от Microsoft платный, требует много места для своих компонентов. А пользуясь нелицензионной версией, надо следить за «кряками», и если они перестают работать, то нужно искать новые.

Есть отличная альтернатива Office: LibreOffice. Это полностью бесплатный офисный софт, поддерживающий распространенные форматы документов. В нем комбинируется несколько компонентов (текстовый модуль, редакторы для формул и рисунков, СУБД, модуль для создания и редактирования презентаций), которые помогут решить почти все нужные пользователю задачи.

Программа распространяется бесплатно, исходный код открыт для всех, а на официальном сайте LibreOffice даже есть дополнительный софт для написания расширений к основному пакету.

Представленный офисный пакет переведен на 30 языков, в том числе и на русский. Поэтому никаких проблем с освоением интерфейса не будет.

Для работы программы нужен Java SE Runtime Environment 7.0.

**BatteryMode**

Интересная программа для слежения за зарядом батареи. С ней вы можете управлять энергопотреблением гаджета как самостоятельно, так и автоматически.

Приложение оснащено большим функционалом, который позволяет создавать разнообразные схемы управления питанием, настраивать уведомления, присваивать приоритет выполнения тех или иных команд и так далее. Кроме гибких настроек вы всегда может проверить состояние батареи встроенной диагностикой, сохраняя при этом результат в текстовый документ.

BatteryMode полностью бесплатна, а назойливая реклама не будет мозолить глаза.

**FluentFileExplorer**

Простенький, но внешне красивый файловый менеджер, позволяющий встроенными средствами открывать популярные архивы, документы форматов PDF и MicrosoftOffice. FluentFileExplorer также позволяет синхронизироваться с такими облачными хранилищами, как OneDrive, Box и другие.

Оформление файлового менеджера можно настроить по своему вкусу. Здесь, кроме множества встроенных тем, вы можете самостоятельно создавать новые, менять прозрачность и прочее.

В приложении присутствует русскоязычное меню, но, судя по всему, перевод выполнен через переводчик и не самый качественный.

Программа предоставляет бесплатную 7-дневную демо-версию, а затем попросит купить полную версию за $6,89.

**TorrentGear**

Максимально простая торрент-программа. Может не только принимать, но и отправлять файлы посредством магнет-ссылок и торрент-файлов. У приложения практически нет настроек, кроме назначения приоритета скачиваемым файлам и возможности ограничения скорости при работе с каждым отдельным торрентом.

О каждом торренте можно получить данные по его состоянию, параметры загрузки, сиды, пиры и прочее.

Программа имеет только английскую локализацию, но полностью бесплатна и, что особенно радует, без рекламных баннеров.

AIDA64

Популярная диагностическая программа для компьютеров теперь работает и на всех гаджетах, у которых установлена Windows 10 или Windows Phone 8.1. На официальном сайте есть также версии для Android и iOS.

Собственно, что-то рассказывать о ней нет смысла. Просто скачивайте и смотрите.

Exaile

Этот плеер для аудиофайлов перекочевал к нам с Linux. Программа обладает незамысловатым интерфейсом, достаточным набором инструментов для работы с музыкальными файлами. А ещё её функционал можно расширять плагинами, общее количество которых больше 50 штук. Вы сможете наслаждаться не только сохраненной у вас на жестком диске музыкой, но и слушать онлайновые радиостанции.

Благодаря плагинам Exaile может стать будильником, сохранить историю композиций и привязаться к крупной радиостанции Last.fm.

Если у вас несколько устройств вывода звука, то можно задать приоритет одному из них. А если наскучило оформление, то можно выбрать сервис, из которого в автоматическом режиме будут скачиваться новые темы.

Русского языка в меню нет, управление очень простое, так что проблем с этим не будет. Плеер распространяется бесплатно.

**BluetoothBatteryMonitor**

Еще одна программа для мониторинга уровня заряда батареи. Только это приложение следит не только за планшетом или ноутбуком, но и за зарядкой устройств с беспроводным подключением. BluetoothBatteryMonitor следит за любым работающим устройством: мышью, клавиатурой, геймпадами, гарнитурой и так далее.

Приложение показывает, сколько в действительности осталось работать гаджету. Единственным условием для корректной работы программы является поддержка передачи такой информации.

Запущенная программа находится в области уведомлений, а когда заряд становится низким – всплывающая подсказка сообщит об этом.

Разработчики за свое приложение просят $2,99. Они же обещают вернуть полную стоимость программы в течение 14 дней после покупки, если вы остались недовольны их продуктом.

**Realarm**

В Windows 10 встроен «Будильник и часы», которого кому-то хватает, а кому-то – нет. Тут на помощь придет программа Realarm, в которой весьма много инструментов для настройки.

В приложении можно переименовывать названия установленных сигналов будильника, задавать точное и приблизительное время запуска сигнала, устанавливать будильники на нужные даты и прочее. А соням придется проснуться: для отключения сигнала им надо будет решить миниатюрную головоломку.

Список всех заданных сигналов отображается в главном окне и в календаре. Устанавливать будильники можно двумя инструментами: продвинутым или быстрым. При использовании первого вы задаете дату, время, периодичность, наименования и тому подобное, а при использовании функции быстрого создания – указывается лишь время срабатывания. Через сутки после воспроизведения будильник самостоятельно удалится. Так что через пару месяцев вас не будет ждать бардак в главном окне программы.

Также у приложения весьма приятный интерфейс и симпатичное оформление. Программа как бы бесплатная, но внутренние расширения придется покупать. По сути, на портале Microsoft доступна бессрочная триал-версия приложения.

**RazerCortex**

Программа будет интересна геймерам. В ней сочетаются оптимизатор системы для игр, библиотека установленных игр, акции и специальные предложения от всех крупных игровых площадок (GoGo, Steam и другие), возможность транслирования в Twitter, YouTube, Panda.tv и так далее.

Встроенный бустер, конечно, значительного прироста производительности не сделает, но на слабых гаджетах играть будет значительно комфортнее. Транслированием игр можно управлять посредством назначения горячих клавиш.

У RazerCortex есть интересная фишка: при использовании программы вам начисляются виртуальные деньги Cortex, которые можно потратить на приобретение игр.

В меню программы присутствует русский язык, в общих настройках можно выбрать тему оформления и произвести настройку уведомлений.

Программа бесплатная, но потребуется завести учетную запись Razor.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №11

1. Тема самостоятельной работы: *Драйверы оборудования*
2. Количество часов: 5ч.
3. Задания для самостоятельной работы: подготовить презентацию
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 5.1 Драйверы оборудования», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

Драйвер - это специальный набор системных программ, который позволяют операционной системе (ОС) компьютера «сотрудничать» с дополнительным устройством, к которому и непосредственно прилагаются драйвера.

Драйвер служит неким декодировщиком между операционной системой ПК пользователя и установленным устройством. Основная задача драйверов - переформатировать запросы, которые поступают от системных и прикладных программ, отсортировать и перевести их на доступный для физического устройства язык. При помощи драйверов производиться управление процессами инициализации, настройками внутренних и внешних параметров, поддерживается обратная связь, обеспечивающая полноценный и своевременный обмен данными, а так же параметры определения функций устройства и их изменение. Драйвера обеспечивают доступ операционной системе к конкретному устройству через общий интерфейс, без дополнительных нюансов и корректировок, которые они берут на себя. В принципе, драйвер принимает и обрабатывает сложные широкоформатные запросы операционной системы и транслирует их устройству на простом доступном ему языке.

Однако далеко не каждому устройству требуется индивидуальный личный драйвер. В операционной системе имеется стандартный набор команд, учитывающий основные параметры последовательности, скорости и своевременности операций многих классов устройств. Если устройство не обладает высокими нестандартными запросами, тогда ему не требуется дополнительный драйвер.

Например, клавиатура, мышь, колонки, наушники, USB порты и некоторые виды модемов, не требуют установки дополнительных драйверов, поскольку изначально распознаются оперативной системой. Можно сказать, что на эти устройства, операционная система имеет встроенные драйвера. Однако и для этих устройств, при замене на иной класс или формат, обязательно потребуются установка дополнительного драйвера.

В некоторых случаях, в приложениях операционной системы можно найти тот или иной драйвер для необходимого устройства. Но естественно, это относится только к устройствам «ровесникам» имеющейся операционной системы, и к тем, которые были выпущены до нее. А все новые устройства, презентованные после выхода установленной операционной системы, обязательно затребуют установку индивидуального драйвера. Также стоит учесть, что речь идет только о самых распространенных простых устройствах, которые считаются стандартными для комплектации ПК.

Согласно стандартизации программного обеспечения, драйверы имеют классификацию по версиям. Маркировка не имеет строгого формата и определяется производителем. Следовательно, пользователям стоит поинтересоваться интерпретацией количества и последовательности цифр на официальном сайте производителя. Некоторые простые стандартные устройства имеют определенную соотносительную согласованность в маркировке. Драйверы Windows для самых востребованных продуктов (звуковые и видео карты, поддерживающие DirectX), всегда нумеруются строго определенным образом.

Порядок таков: первая цифра показывает номер версии требуемой операционной системы, далее определяется подходящая DirectX, а последние заключительные цифры определяют непосредственно версию драйвера.

Определить версию драйвера не составляет особой проблемы. Стоит лишь его распаковать до состояния «отдельных» компонентов, а потом, щелкнуть по файлу с помощью правой кнопки мыши, и выбрать «свойства». Зачастую, версия драйвера наглядно выделяется цветом. Правда, некоторые производители снабжают драйвера индивидуальным инсталлятором. В таком случае, крайне сложно определить версию используемого драйвера, что порой становиться сложной задачей.

1. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

### Методические рекомендации (указания) к выполнению самостоятельной работы №12

1. Тема самостоятельной работы: *Средства восстановления и защиты ОС*
2. Количество часов: 5ч.
3. Задания для самостоятельной работы: подготовить презентацию
4. Цель самостоятельной работы: формирование умений и знаний по теме

«Тема 5.1 Драйверы оборудования», создание условий для формирования компетенций:

ОК2 - организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК4 - осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК5 - владеть информационной культурой, анализировать и оценивать информацию с использованием информационно- коммуникационных технологий.

ОК8 - самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК9 - ориентироваться в условиях частой смены технологий в профессиональной деятельности.

1. Рекомендации преподавателя:

## Краткие теоретические сведения

**1. *Цифровая подпись драйверов*.**

Нет операционных систем, абсолютно надежных и отказоустойчивых. В процессе функционирования компьютера может возникнуть ряд проблем, приводящих к сбоям, повреждениям и отказам операционной системы. В составе операционных систем Windows имеется большой набор средств обеспечений отказоустойчивости и предотвращения сбоев.

Все системные файлы и драйверы Windows XP/2000/2003 защищены цифровой подписью. Цифровая подпись Microsoft гарантирует, что файл, подписанный ею, тестировался на совместимость с Windows 2000 и не был модифицирован или переписан во время установки дополнительного программного обеспечения.

В зависимости от установленных опций настройки Windows 2000 может игнорировать драйверы, не имеющие цифровой подписи, или выводить предупреждение при обнаружении таких драйверов (опция по умолчанию), или не допускать их установки. Для установки требуемой опции защиты системных файлов Windows 2000 необходимо выполнить следующие действия.

На панели управления щелкнуть по значку *Система* (System) и перейти на вкладку *Оборудование* (Hardware).

Нажать кнопку *Подписывания драйверов* (Driver Signing). На экране появится диалоговое окно *Параметры подписывания драйвера* (Driver Signing Options), в котором имеется группа *Проверка подписи файла* (File Signature Verification).

Выбрать требуемую опцию из следующих возможных:

* + *Пропускать* (Ignore) - система дает возможность устанавливать любые драйверы и системные файлы, игнорируя наличие или отсутствие у них цифровой подписи;
  + *Предупреждать* (Warn) - система будет выводить предупреждающие сообщения при попытке установить драйвер или системный файл, не имеющий цифровой подписи;
  + *Блокировать* (Block) - драйверы, не имеющие цифровой подписи, устанавливаться не будут.

**2. *Защита системных файлов*.**

Кроме цифровых подписей, Windows 2000 имеет ряд функциональных возможности по защите драйверов и системных файлов. Все версии Windows (до Windows 2000) имели недостаток: при установке дополнительного программного обеспечения практически любые совместно используемые системные файлы, в том числе \*.dll и \*.ехе, могли быть изменены, что очень часто приводило к проблемам вплоть до краха всей системы.

Попытка исправить такую ситуацию сделана в Windows 2000/NT. Специальная функция защиты системных файлов Windows (Windows File Protection, WFP) включает в свой состав два компонента:

* Сервис защиты системных файлов (Windows File Protection service);
* Утилиту командной строки System Checker (Sfc.exe).

Сервис WFP работает по принципу определения цифровых подписей защищенных системных файлов, не позволяя произвольно модифицировать и замещать эти файлы. Сервис WFP работает в фоновом режиме и защищает все системные файлы, установленные программой Windows Setup при инсталляции операционной системы. Выявив попытку изменения или перемещения защищенного файла, сервис WFP проверяет измененный файл на наличие у него цифровой подписи.

В случаях, когда версия, предназначенная для замены, не является корректной, этот файл замещается резервной копией из папки %SystemRoot%\system32\dllcache и регистрируется попытка замещения файла в журнале системных событий.

По умолчанию функция WFP всегда активизирована и позволяет выполнять замену системных файлов только в случае установки следующих видов программного обеспечения:

сервисные пакеты Windows 2000 (с использованием программы Update.exe);

дистрибутивные пакеты типа Hotfix (с использованием Hotfix.exe);

обновления версии операционной системы (с помощью Winnt32.exe);

программное обеспечение Windows Update.

**3. *Проверка системных файлов (System File Checker)*.**

Средство проверки системных файлов (System File Checker, Sfc.exe) представляет собой утилиту командной строки. Она сканирует все установленные системные файлы и выполняет проверку их версий при перезагрузке компьютера. Если эта утилита обнаружит, что один из защищаемых системных файлов был замещен, она найдет корректную версию этого файла в каталоге %SystemRoot%\system32\dllcache и запишет ее поверх измененного файла.

Папка %SystemRoot%\system32\dllcache может быть достаточно объемной (до 300 Мбайт). По этой причине в целях экономии дискового пространства ее иногда удаляют. Если сервис WFP не сможет обнаружить нужной версии файла в папке %SystemRoot%\system32\dllcache, то система потребует установить дистрибутивный CD-ROM или указать путь к папке, из которой такая версия может быть скопирована. Запустив команду sfc /scannow, можно проверить все защищенные системные файлы. Если каких-либо файлов нет, утилита потребует их установить.

**4. *Верификация цифровой подписи файлов*.**

Верификация (проверка) цифровой подписи файлов (программа sigverif) позволяет идентифицировать все установленные на проверяемом компьютере файлы, не имеющие цифровой подписи, и получить об этих файлах следующую информацию: имя файла и путь к нему; дату модификации файла; тип файла и точный номер его версии.

Для устранения проблем, связанных с заменой системных файлов некорректными версиями, нужно использовать информацию, собранную программой Sigverif в файле журнала. Для этого необходимо сделать следующее.

1. Запустить программу Sigverif (из командной строки) и в появившемся на экране окне *Проверка подписи файла* (File Signature Verification) щелкнуть по кнопке *Дополнительно* (Advanced).
2. В новом окне *Дополнительные параметры* проверки подписи файла (Advanced File Signature Verification Settings) перейти на вкладку *Ведение журнала* (Logging) и установить флажок *Сохранять результаты проверки подписи в журнале* (Save the fib signature verification results to a log file).
3. Перейти в группу *Параметры журнала* (Logging options) и установить желаемую опцию ведения журнала из следующих возможных:
   * добавлять к существующему журналу (Append to existing log file);
   * заменять существующий журнал (Overwrite existing log file).
4. В поле *Имя файла журнала* (Log file name) можно ввести имя файла журнала.
5. Нажать кнопку ОК. Произойдет возврат в окно *Проверка подписи файла*.
6. Для начала сканирования нажать кнопку *Начать* (Start). Процесс сканирования индицируется индикатором *Просмотр файлов* (Scanning files). Его можно прервать нажатием кнопки *Остановить* (Stop).
7. После завершения сканирования на экране появится окно *Результаты проверки подписи* (Signature Verification Results). В нем будет отображен список файлов, не имеющих цифровой подписи. Если таких файлов нет, можно просмотреть журнал результатов проверки.

**5. *Откат драйверов*.**

Довольно часто такие проблемы, как аппаратные конфликты, нестабильное поведение системы, неправильная работа устройств и даже ошибки STOP бывают вызваны некорректным драйвером. В таких случаях желательно было бы быстро заменить проблемный драйвер предыдущей версией без переустановки системы. Функция *отката драйвера* оказывается незаменимой при устранении неполадок, при отладке бета-версий драйверов и в других ситуациях.

Чтобы воспользоваться функцией отката драйверов, нужно выполнить следующие действия.

1. Выбрать значок *Система* на панели управления, перейти на вкладку *Оборудование* и нажать кнопку *Диспетчер устройств*.
2. Выполнить щелчок правой кнопкой мыши по устройству, обновленный драйвер которого вызывает проблему, и выбрать из контекстного меню строку *Свойства*.
3. В раскрывшемся диалоговом окне свойств выбранного устройства перейти на вкладку *Драйвер* и нажать кнопку *Откатить*.
4. Диспетчер устройств предложит подтвердить намерение выполнить откат драйвера. Для этого следует нажать кнопку *Да*. Если старая версия драйвера недоступна, функция отката драйвера выведет окно с уведомлением и предложит воспользоваться другими средствами устранения неполадок.

**6. *Безопасный режим загрузки*.**

Если при появлении меню загрузки Windows XP нажать клавишу F8, то на экране появится меню опций отладки и дополнительных режимов загрузки.

В *безопасном режиме* Windows 2000/XP использует параметры по умолчанию: VGA-монитор; драйвер мыши Microsoft; минимальный набор драйверов устройств необходимый для запуска системы. Если при загрузке в безопасном режиме проблемы исчезли, нужно изменить значения параметров по умолчанию и удалить лишние драйверы до полного решения проблемы.

Если несовместимый драйвер вызвал проблему при первой же перезагрузке, то в этом случае поможет опция *Загрузка последней удачной конфигурации* (Last Know Good Configuration). Когда пользователь выбирает из меню безопасного режима эту опцию, система при запуске использует информацию ключа реестра HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet и восстанавливает всю конфигурационную информацию, сохраненную после того, как компьютер был в последний раз успешно загружен.

Если известен драйвер, вызвавший проблему, то можно попробовать использовать такие опции меню безопасного режима, как *Безопасный режим* (Safe Mode), *Безопасный режим с загрузкой сетевых драйверов* (Safe Mode with Networking) или *Безопасный режим с поддержкой командной строки* (Safe Mode with Command Prompt). После загрузки системы можно будет удалить проблемный драйвер с помощью штатных средств Windows 2000 -*Мастера оборудования* (Hardware Wizard) или *Диспетчера устройств* (Device Manager).

Если системный и загрузочный разделы отформатированы для использования файловой системы FAT, можно попытаться загрузить компьютер с помощью загрузочной дискеты MS DOS (или Windows 9x) и вручную удалить или переименовать файл проблемного драйвера.

**7. *Точки восстановления системы*.**

В полностью сконфигурированной системе крах нового приложения может привести к значительным временным затратам по восстановлению системы, что делает важной функцию отката Windows XP.

Перед установкой приложения можно создать точку восстановления, представляющую собой «слепок» операционной системы и установленных приложений, который помещается на жесткий диск. После создания точки восстановления, можно устанавливать и тестировать новое приложение.

Если после установки нового приложения возникли проблемы, можно отменить установку приложения. Если это не поможет или нет уверенности в результатах удаления приложения, можно воспользоваться точкой восстановления и «откатить» систему к ее предыдущему состоянию. Файлы приложения при этом не удаляются. Если они остались, то их требуется удалить вручную.

Для повышения производительности системы часто отключают службу восстановления. Тогда перед созданием точки восстановления придется каждый раз проверять, можно ли будет ею воспользоваться.

*Чтобы узнать состояние службы восстановления системы*, нужно щелкнуть правой клавишей мыши по значку *Мой компьютер* и выбрать в контекстном меню строку *Свойства*. Далее перейти на вкладку *Восстановление системы*. Если установлен флажок *Отключить восстановление системы на всех дисках*, то точку восстановления создать нельзя. Чтобы изменить параметры отдельного диска, нужно его выделить и щелкнуть по кнопке *Параметры*. Появится окно *Свойства системы*, которое позволит включить или выключить наблюдение и задать объем диска, резервируемый под точки восстановления.

При решении определенных задач, таких как установка заплат, Windows создает точки восстановления автоматически. Однако лучше создавать точки восстановления вручную.

Чтобы *создать точку восстановления*, нужно воспользоваться мастером восстановления системы, который запускается последовательностью команд *Пуск* → *Программы* → *Специальные* → *Служебные* → *Восстановление системы*. В начальном окне мастера нужно сделать выбор между созданием новой точки восстановления и использованием существующей точки. Для создания новой точки восстановления установить переключатель *Создать точку восстановления* и щелкнуть по кнопке *Далее*. Теперь нужно ввести имя точки восстановления и щелкнуть по кнопке *Создать*. После создания точки восстановления (это займет некоторое время) мастер откроет последнее диалоговое окно с именем и датой создания точки восстановления. Щелкнув по кнопке *Домой*, можно вернуться в начальное окно восстановления системы, а щелкнув по кнопке *Закрыть* - закрыть окно.

Точка восстановления - это не резервная копия системы, для которой нужна полная архивация. Причина создания точки восстановления - подготовка места для отката системы в случае сбоя приложения, обновления или заплаты. Следует сначала удалить приложение, а затем воспользоваться точкой восстановления, чтобы ликвидировать вред, нанесенный реестру и другим приложениям.

Для восстановления системы нужно вызвать мастера восстановления и в его начальном диалоговом окне установить переключатель *Восстановление более раннего состояния компьютера* и щелкнуть по кнопке *Далее*. На экране появится диалоговое окно *Выбор контрольной точки восстановления*. После выбора конкретной точки восстановления нужно щелкнуть по кнопке *Далее*. Мастер восстановления отобразит окно, в котором будут описаны результаты воздействия данной точки восстановления на систему. Чтобы начать процедуру восстановления системы, нужно убедиться, что все окна приложений закрыты, а затем щелкнуть по кнопке *Далее*. Система восстановит себя и перезагрузится. После завершения перезагрузки система сообщит, что можно отменить восстановление, если его результаты не устраивают пользователя.

**8. *Резервное копирование и восстановление*.**

Для резервного копирования и восстановления данных в Windows XP/2003 используется встроенная утилита *Архивация* (Backup). Новая версия программы обеспечивает поддержку различных видов носителей резервной копии, что позволяет выполнять резервное копирование на любое устройство хранения информации, поддерживаемое операционной системой (любые жесткие диски, магнитооптические накопители и другие устройства, а не только стримеры).

Новой версией программы также поддерживается технология теневых копий томов, которая позволяет создать «моментальный снимок» жесткого диска на момент начала резервного копирования. В процессе копирования будет использоваться этот моментальный снимок, остающийся неизменным, а не фактическое содержимое диска, которое в ходе резервного копирования может изменяться, что позволяет пользователю продолжать работать в ходе выполнения резервного копирования. При этом программа архивации данных может выполнять резервное копирование открытых файлов, с которыми на данный момент времени работает пользователь.

Чтобы вызвать программу архивации, нужно последовательно выполнить команды: *Пуск* → *Программы* → *Стандартные* → *Служебные* → *Архивация данных*. На экране появится окно *Программа архивации*, если программа работает в расширенном режиме. Если программа *Архивации* данных работает в режиме мастера (режим легко переключается), откроется окно *Мастер архивации или восстановления*.

Выполнить резервное копирование можно двумя способами.

*Первый способ* заключается в использовании *Мастера архивации*, который можно вызвать, нажав кнопку *Мастер архивации* на вкладке *Добро пожаловать* программы архивации. Нажав кнопку *Далее*, можно выполнить резервное копирование конфигурационных файлов, установив переключатель *Архивировать только данные состояния системы* и следуя указаниям *Мастера*.

*Второй способ* связан с использованием вкладки *Архивация* программы архивации. В этом случае из списка дисков, файлов и папок, подлежащих системному копированию, нужно выбрать опцию System State (Состояние системы). В списке *Местонахождение архива* нужно указать ленточное устройство, если оно имеется на компьютере, и копирование должно быть выполнено на ленту. Если такого устройства нет, по умолчанию будет установлена опция *Файл*. В поле *Носитель архива* или *имя файла* нужно указать путь к файлу, в который будет выполняться резервное копирование. Дополнительные опции резервного копирования можно задать, выбрав команду *Параметры* из меню *Сервис*.

Чтобы начать процедуру резервного копирования, следует нажать кнопку *Архивировать*. На экране появится окно *Сведения о задании архивации*. Чтобы установить в этом окне дополнительные опции задания на резервное копирование, нужно нажать кнопку *Дополнительно*. Раскроется окно *Дополнительные параметры архивации*. Обратите внимание на состояние флажка *Автоматически архивировать защищенные системные файлы вместе с состоянием системы*. Программа архивации не позволяет выполнить выборочное резервное копирование отдельные компонентов набора System State.

Windows XP/2003 предоставляют возможность одновременно с резервными копированием системных конфигурационных файлов выполнить резервное копирование всех защищенных файлов операционной системы. По умолчанию эта опция активизирована, однако ее можно блокировать, сбросив данный флажок.

Ход архивации система отображает на экране, о завершении архивации выдается сообщение. Отчет о результатах архивации можно просмотреть, используя *Блокнот*.

При выполнении резервного копирования данных из набора System State система сохраняет копии файлов реестра в папке %SystemRoot%\repair\regback. В случае удаления или повреждения файлов реестра резервные копии его файлов, сохраненные в этой папке, могут быть использованы для восстановления системы без необходимости полной процедуры восстановления системных конфигурационных данных.

Если попытки восстановить поврежденную систему завершаются неудачей, работоспособная копия системных конфигурационных данных может быть использована для восстановления работоспособности.

*Чтобы восстановить системные конфигурационные данные*, нужно вызвать программу архивации в расширенном режиме и на вкладке *Восстановление и управление носителем* выбрать опцию System State (Состояние системы), после чего нажать кнопку *Восстановить*. Через некоторое время будут восстановлены системные файлы, а также другие запрошенные данные.

**9. *Аварийное восстановление системы*.**

Кроме традиционных функциональных возможностей по резервному копированию и восстановлению данных, версия программы архивации, входящая в состав Windows ХР, включает в себя новую функцию подготовки к аварийному (автоматическому) восстановлению системы (Automated System Recovery, ASR). Аварийное восстановление системы представляет собой двухступенчатый процесс, который позволяет пользователю восстановить поврежденную копию Windows XP, используя для этого резервную копию конфигурационных данных операционной системы и информацию о дисковой конфигурации, сохраненную на дискете.

Для подготовки к аварийному восстановлению системы необходимо выполнить следующие действия.

1. Освободить достаточный объем дискового пространства для выполнения резервного копирования (если есть стример, подготовить его).
2. Запустить программу архивации в расширенном режиме и нажать кнопку *Мастер аварийного восстановления системы*. В первом окне программы-мастера подготовки к автоматическому восстановлению системы нажать кнопку *Далее*.
3. На следующей странице мастера указать тип носителя, на который будет производиться резервное копирование, и путь к резервной копии. Нажать кнопку *Далее*.
4. В последнем окне мастера аварийного восстановления нажать кнопку *Готово*. Программа архивации начнет сканирование системы и составит список файлов, которые необходимо включить в состав резервной копии ASR. После ряда информационных сообщений на экране появится окно *Ход архивации*, сообщающее о ходе процесса создания аварийной копии системы.
5. Когда процесс резервного копирования завершится, мастер аварийного восстановления предложит вставить в дисковод чистую дискету, на которой будет сохранена информация о конфигурации дисковой подсистемы, в том числе сигнатуры дисков, таблица разделов, данные о томах, информация о конфигурации компьютера, а также список файлов, подлежащих восстановлению. При выполнении процедуры аварийного восстановления с этой дискеты будет считываться информация о конфигурации дисковой подсистемы и другие данные.
6. Для просмотра отчета о ходе резервного копирования нужно нажать кнопку *Отчет* в окне *Ход архивации*. Отчет открывается в блокноте.

Процесс аварийного восстановления поврежденной системы с использованием процедуры ASR запускается из программы установки операционной системы Windows ХР Setup, что требует наличия дистрибутивного компакт-диска. Процедура ASR восстановит конфигурацию дисковой подсистемы с использованием данных, сохраненных на дискету ASR, переформатирует раздел %SystemDrive% (на котором установлена копия ОС, подлежащая восстановлению), затем переустановит на этот раздел Windows ХР и восстановит конфигурационную информацию системы с резервной копии.

Чтобы провести процесс аварийного восстановления с помощью резервной копии ASR, следует выполнить следующую последовательность шагов.

1. Подготовить дискету ASR с конфигурационной информацией, соответствующей конфигурации компьютера на тот момент, когда последний раз выполнялась процедура подготовки ASR и компакт-диск Windows XP (имеется в виду, что резервная копия данных ASR создана на жестком диске).
2. Запустить программу Windows Setup (можно просто перезагрузить компьютер с дистрибутивного диска CD\_ROM).
3. Нажать любую клавишу, когда на экране появится соответствующее сообщение (Press any key to boot from CD...).
4. После запуска программы Windows Setup при появлении сообщения Нажмите F2 для запуска автоматического восстановления системы (ASR) нажать клавишу F2.
5. Вставить в дисковод дискету ASR, когда программа Setup предложит это сделать.
6. На экране появится сообщение следующего содержания: Подготовка к автоматическому восстановлению системы. Для его отмены нажмите клавишу <ESC>.
7. Если пользователь намерен продолжать процедуру, следует не реагировать на это предложение, и программа Setup последовательно отобразит следующие сообщения: Запуск автоматического восстановления Windows; Копирование файлов; Программа установки запускает Windows. После этого начинается переформатирование раздела %SystemDrive% и проверка остальных разделов с целью определения, не нуждаются ли они в восстановлении.
8. После форматирования и завершения проверки всех разделов ASR построит список файлов для копирования и предложит вставить носитель с резервной копией ASR. Если при подготовке к ASR резервное копирование выполнялось в файл, то этот шаг будет пропущен. Далее процедура ASR выполнит автоматическую установку Windows XP, а затем восстановит системную конфигурацию.

В случае утраты дискеты ASR её можно восстановить. Дело в том, что файлы, которые копируются на дискету, включаются также в состав резервной копии ASR. Поэтому имеется возможность восстановления дискеты. Для этого нужно выполнить следующие действия.

1. Отформатировать дискету и вставить ее в компьютер.
2. Запустить программу архивации в расширенном режиме. Перейти на вкладку *Восстановление и управление носителем* и выбрать команду *Каталогизировать архивный файл* из меню *Сервис* (рассматривается случай, когда резервная копия была создана на жестком диске).
3. В левой части окна развернуть архив, соответствующий диску ASR, который требуется восстановить.
4. Развернуть папку %SystemRoot%\repair и пометить для восстановления файлы asr. sif, asrpnp.sif, setup.log. В списке *Восстановить файлы в* выбрать опцию *Альтернативное размещение*, а в поле *Альтернативное размещение* указать путь к корневому каталогу дискеты («А:»). Программа попросит подтвердить восстановление. Щелкнуть по кнопке *ОК*.
5. Нажать кнопку *Восстановить*, выбранные файлы будут скопированы на дискету. Далее дискету нужно извлечь и использовать ее в ходе восстановления ASR.

**10. *Консоль восстановления*.**

Консоль восстановления Windows XP (Recovery Console) представляет собой исполняющую среду с интерфейсом командной строки, которая предоставляет администраторам и пользователям с административными правами необходимый минимум средств, позволяющих выполнить восстановительные процедуры в системе, имеющей проблемы с загрузкой: запускать и останавливать сервисы, форматировать диски, выполнять чтение и запись данных на локальные жесткие диски, устранять проблемы с поврежденной главной загрузочной записью (MBR) и поврежденными загрузочными секторами, выполнять другие административные задачи.

Существует два способа запуска консоли восстановления.

*Первый способ* предполагает запуск консоли восстановления из программы Windows Setup. Это можно сделать, если имеется загрузочное устройство CD-ROM или загрузочная дискета и доступ к дистрибутивным файлам для запуска программы Windows Setup. После запуска программы Windows Setup и завершения процесса начального копирования файлов появится экран, в котором Setup приглашает к инсталляции системы и предлагает на выбор установить Windows, восстановить поврежденную копию Windows или завершить программу установки. Нужно нажать клавишу R (восстановление). Из двух предложенных опций по восстановлению поврежденной системы нужно выбрать опцию: *с помощью консоли восстановления*, нажав клавишу С. Далее следовать указаниям системы.

*Второй способ* запуска консоли восстановления предполагает предварительную установку консоли на жесткий диск и включение ее как одну из доступных опций в меню загрузки системы. В этом случае нужно выполнить следующие действия.

1. Зарегистрироваться в Windows как администратор.
2. Вставить дистрибутивный компакт-диск Windows XP в устройство CD-ROM.
3. Нажать кнопку *Нет* (No), если будет предложено обновить операционную систему.
4. В режиме командной строки перейти на дистрибутивный диск Windows XP и ввести команду имя\_CD-ROM\i386\winnt32.ехе/cmdcons. Откроется окно Windows 2000 Setup. Щелкнуть по кнопке Да.
5. Следовать инструкциям, появляющимся на экране.

Интерфейс консоли восстановления представляет собой полноэкранный интерфейс командной строки. Для вывода доступных команд можно воспользоваться командой help. Консоль восстановления запоминает предыдущие введенные команды и позволяет их выбирать с помощью клавиш «вверх» и «вниз». Для редактирования предыдущей команды можно использовать клавишу Backspace. Для выхода из консоли используется команда exit.

Если требуется удалить консоль восстановления из списка опций, доступных в меню загрузки, нужно выполнить следующие операции.

1. Удалить из корневого каталога системного раздела папку \cmdcons и файл cmldr.
2. Открыть для редактирования файл Boot.ini и удалить в нем строку, соответствующую опции запуска консоли восстановления.

**11. *Загрузочная дискета*.**

В случае возникновения проблем с загрузкой операционной ситемы Windows можно воспользоваться предварительно созданной загрузочной дискетой. Для ее изготовления необходимо отформатировать дискету в Windows 2000/XP и скопировать на нее следующие файлы:

* Ntldr;
* Ntdetect.com;
* Boot.ini;
* Bootsect.dos - если используется мультизагрузочная система и нужно обеспечить возможность загрузки с дискеты также Windows 9x или DOS;
* Ntbootdd.sys - если в файле Boot.ini применяется синтаксис scsi ().

Загрузочная дискета поможет выполнить загрузку компьютера в следующих случаях:

1. повреждены главная загрузочная запись и/или загрузочный сектор раздела на системном разделе;
2. возникли проблемы с диском, на котором находится системный раздел;
3. выполняется переконфигурирование жестких дисков и нужно обеспечить возможность запуска Windows 2000 в случае возникновения проблем.

#### ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО ВЫПОЛНЕНИЯ

**Задание 1.**

1. Проверьте параметры подписывания драйверов на вашем компьютере. Установите опцию *Предупреждать*, если устанавливается драйвер без цифровой подписи.
2. Проверьте наличие файла %SystemRoot%\system32\dllcache на жестком диске. Изучите опции утилиты Sfc.exe. Запустите ее, проанализируйте полученные результаты.
3. Проведите верификацию цифровой подписи файлов на вашем компьютере, используя утилиту sigverif. Имеются ли неподписанные файлы? Просмотрите журнал результатов проверки.
4. Попробуйте заменить драйвер какого-либо устройства и выполните функцию отката драйвера.

**Задание 2.**

1. Загрузите компьютер в безопасном режиме. Как изменились возможности компьютера? Когда используется опция *Загрузка последней удачной конфигурации*?
2. Определите состояние службы восстановления компьютера. Создайте точку восстановления системы компьютера.
3. Ознакомьтесь с возможностями системной программы архивации. Выполните резервное копирование системных конфигурационных файлов из набора System State. Найдите архивный файл и определите его объем. Выполните восстановление файлов из полученного архива.

**Задание 3.**

1. Выполните процедуру аварийного восстановления системы. Для архивации выберите системные файлы диска С:.
2. Восстановите дискету ASR (считаем ее утраченной), как это рассмотрено выше.

**Задание 4.**

1. Установите на компьютере консоль восстановления. Проверьте, изменилось ли меню загрузки системы.
2. Запустите консоль восстановления. Используя команду help, ознакомьтесь со списком команд, доступных при работе с консолью.
3. Изготовьте загрузочную дискету и проверьте ее работоспособность.

### Контрольные вопросы

1. Для чего предназначена цифровая подпись системных файлов?
2. Какие функции выполняет *Сервис защиты системных файлов*?
3. С помощью какой утилиты осуществляется проверка системных файлов? Какие функции она выполняет?
4. Какие функции выполняет утилита Sigverif?
5. Для чего служит процедура Откат драйверов?
6. Что такое безопасный режим загрузки Windows? Какие задачи с помощью его решаются?
7. Что такое точки восстановления системы? Как с помощью их решается проблема устранения проблем, вызванных установкой нового приложения?
8. Какие возможности по резервному копированию и восстановлению предоставляет утилита *Архивация*?
9. Что такое *Аварийное восстановление системы*? Какая утилита его выполняет?
10. Литература для самостоятельной работы:

[Федорова Г.Н.](http://www.academia-moscow.ru/authors/detail/46229/) Основы проектирования баз данных: учебник для студ. учреждений сред. проф. образования / Г.Федорова. – М.: Издательский центр «Академия», 2017. – 224 с.

Критерии оценки:

**«5» -** ответы на теоретические вопросы 100%

**«4» -** ответы на теоретические вопросы 70%

**«3» -** ответы на теоретические вопросы 50%

1. [↑](#footnote-ref-1)
2. [↑](#footnote-ref-2)
3. [↑](#footnote-ref-3)
4. [↑](#footnote-ref-4)