МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Колледж экономики, управления и права

Методические указания по организации практической работы студентов по учебной дисциплине ОП 07. Операционные системы и среды

09.02.04 Информационные системы (по отраслям)
09.02.05 Прикладная информатика (по отраслям).

Ростов-на-Дону

2017

Методические указания по организации практической и самостоятельной работы студентов по дисциплине «Операционные системы и среды» разработаны с учетом ФГОС среднего профессионального образования по специальностям:

09.02.04 Информационные системы (по отраслям);

09.02.05 Прикладная информатика (по отраслям).

Методические указания содержат перечень практических работ, рекомендации по выполнению практической или самостоятельной работы, вопросы для самоконтроля.

Методические указания предназначены для обучающихся и преподавателей колледжа.

Автор - составитель: С.В. Шигаева преподаватель колледжа ЭУП

Одобрены на заседании предметной (цикловой) комиссии специальностей Информационные системы (по отраслям) и Прикладная информатика (по отраслям).

Протокол № 1 от «31» августа 2017 г

Председатель предметной (цикловой) комиссии _____ В.М. Кносаль
личная подпись

и одобрены решением учебно-методического совета колледжа.

Протокол № 1 от «31» августа 2017 г

Председатель учебно-методического совета колледжа
комиссии _____ С.В.Шинакова
личная подпись

Рекомендованы к практическому применению в образовательном процессе

Рецензенты:

Лабораторная работа №1
«Переход из одной системы счисления в другую»

Цель работы:

1. Научиться переводить числа из одной системы счисления в другую.

1 Правила перевода чисел из одной системы счисления в другую

Перевод чисел из одной системы счисления в другую составляет важную часть машинной арифметики. Рассмотрим основные правила перевода.

1.1 Перевод двоичного числа в десятичное

Для перевода двоичного числа в десятичное необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 2, и вычислить по правилам десятичной арифметики:

$$X_2 = A_n \cdot 2^{n-1} + A_{n-1} \cdot 2^{n-2} + A_{n-2} \cdot 2^{n-3} + \ldots + A_2 \cdot 2^1 + A_1 \cdot 2^0$$

При переводе удобно пользоваться таблицей степеней двойки:

Таблица 1. Степени числа 2

n (степень)	0	1	2	3	4	5	6	7	8	9	10
2^n	1	2	4	8	16	32	64	128	256	512	1024

Пример. Число 11101000_2 перевести в десятичную систему счисления.

$$11101000_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 232_{10}$$ 4

1.2 Перевод восьмеричного числа в десятичное

Для перевода восьмеричного числа в десятичное необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 8, и вычислить по правилам десятичной арифметики:

$$X_8 = A_n \cdot 8^{n-1} + A_{n-1} \cdot 8^{n-2} + A_{n-2} \cdot 8^{n-3} + \ldots + A_2 \cdot 8^1 + A_1 \cdot 8^0$$

При переводе удобно пользоваться таблицей степеней восьмерки:

Таблица 2. Степени числа 8

n (степень)	0	1	2	3	4	5	6
8^n	1	8	64	512	4096	32768	262144

Пример. Число 75013_8 перевести в десятичную систему счисления.

$$75013_8 = 7 \cdot 8^4 + 5 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8^1 + 3 \cdot 8^0 = 31243_{10}$$

1.3 Перевод шестнадцатеричного числа в десятичное

Для перевода шестнадцатеричного числа в десятичное необходимо его записать в виде многочлена, состоящего из произведений цифр числа и соответствующей степени числа 16, и вычислить по правилам десятичной арифметики:

$$X_{16} = A_n \cdot 16^{n-1} + A_{n-1} \cdot 16^{n-2} + A_{n-2} \cdot 16^{n-3} + \ldots + A_2 \cdot 16^1 + A_1 \cdot 16^0$$

При переводе удобно пользоваться таблицей степеней числа 16:

Таблица 3. Степени числа 16

n (степень)	0	1	2	3	4	5	6
16^n	1	16	256	4096	65536	1048576	16777216

Пример . Число $FDA1_{16}$ перевести в десятичную систему счисления.

$$FDA1_{16} = 15 \cdot 16^3 + 13 \cdot 16^2 + 10 \cdot 16^1 + 1 \cdot 16^0 = 64929_{10}$$

1.4 Перевод десятичного числа в двоичную систему

Для перевода десятичного числа в двоичную систему его необходимо последовательно делить на 2 до тех пор, пока не останется остаток, меньший или равный 1. Число в двоичной системе записывается как последовательность последнего результата деления и остатков от деления в обратном порядке.

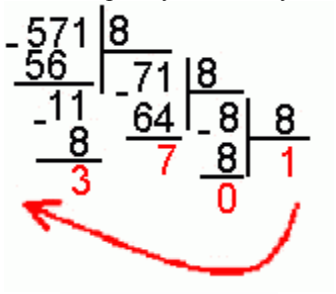
Пример. Число 22_{10} перевести в двоичную систему счисления.

$$22_{10} = 10110_2$$

1.5 Перевод десятичного числа в восьмеричную систему

Для перевода десятичного числа в восьмеричную систему его необходимо последовательно делить на 8 до тех пор, пока не останется остаток, меньший или равный 7. Число в восьмеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке.

Пример. Число 571_{10} перевести в восьмеричную систему счисления.



$$571_{10} = 1073_8$$

1.6 Перевод десятичного числа в шестнадцатеричную систему

Для перевода десятичного числа в шестнадцатеричную систему его необходимо последовательно делить на 16 до тех пор, пока не останется остаток, меньший или равный 15. Число в шестнадцатеричной системе записывается как последовательность цифр последнего результата деления и остатков от деления в обратном порядке.

Пример. Число 7467_{10} перевести в шестнадцатеричную систему счисления.

$$7467_{10} = 1D2B_{16}$$

1.7 Перевод из двоичной системы в восьмеричную

Чтобы перевести число из двоичной системы в восьмеричную, его нужно разбить на триады (тройки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую триаду нулями, и каждую триаду заменить соответствующей восьмеричной цифрой (табл. 3).

Пример. Число 1001011_2 перевести в восьмеричную систему счисления.

$$001\ 001\ 011_2 = 113_8$$

1.8 Перевод из двоичной системы в шестнадцатеричную

Чтобы перевести число из двоичной системы в шестнадцатеричную, его нужно разбить на тетрады (четверки цифр), начиная с младшего разряда, в случае необходимости дополнив старшую тетраду нулями, и каждую тетраду заменить соответствующей восьмеричной цифрой (табл. 3).

Пример. Число 1011100011_2 перевести в шестнадцатеричную систему счисления.

$$0010\ 1110\ 0011_2 = 2E3_{16}$$

1.9 Перевод восьмеричного числа в двоичную

Для перевода восьмеричного числа в двоичную необходимо каждую цифру заменить эквивалентной ей двоичной триадой.

Пример. Число 531_8 перевести в двоичную систему счисления.

$$531_8 = 101011001_2$$

1.10 Перевод шестнадцатеричного числа в двоичную

Для перевода шестнадцатеричного числа в двоичную необходимо каждую цифру заменить эквивалентной ей двоичной тетрадой.

Пример. Число $EE8_{16}$ перевести в двоичную систему счисления.

$$EE8_{16} = 111011101000_2$$

1.11 Перевод восьмеричной системы счисления в шестнадцатеричную и обратно

При переходе из восьмеричной системы счисления в шестнадцатеричную и обратно, необходим промежуточный перевод чисел в двоичную систему.

Пример 1. Число FEA_{16} перевести в восьмеричную систему счисления.

$$FEA_{16} = 111111101010_2$$
$$111\ 111\ 101\ 010_2 = 7752_8$$

Пример 2. Число 6635_8 перевести в шестнадцатеричную систему счисления.

$$6635_8 = 110110011101_2$$
$$1101\ 1001\ 1101_2 = D9D_{16}$$

Соответствие между цифрами в различных системах счисления

10сс	2сс	8сс	16сс
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Задание

1. Система счисления – ДВОИЧНАЯ

1.1 Переведите число 43 из десятичной в ДВОИЧНУЮ
1.2 Переведите число 345 из десятичной в ДВОИЧНУЮ
1.3 Переведите число 10000 из ДВОИЧНОЙ в десятичную
1.4 Переведите число 100011001 из ДВОИЧНОЙ в десятичную
1.5 Переведите число 58 из десятичной в ДВОИЧНУЮ
1.6 Переведите число 610 из десятичной в ДВОИЧНУЮ
1.7 Переведите число 101101 из ДВОИЧНОЙ в десятичную
1.8 Переведите число 1001001110 из ДВОИЧНОЙ в десятичную
1.9 Переведите число 54 из десятичной в ДВОИЧНУЮ
1.10 Переведите число 953 из десятичной в ДВОИЧНУЮ
1.11 Переведите число 1001000 из ДВОИЧНОЙ в десятичную
1.12 Переведите число 1101111000 из ДВОИЧНОЙ в десятичную
1.13 Переведите число 3AED из ШЕСТНАДЦАТЕРИЧНОЙ в двоичную
1.14 Переведите число 6B из ШЕСТНАДЦАТЕРИЧНОЙ в двоичную
1.15 Переведите число 56 из десятичной в ДВОИЧНУЮ
1.16 Переведите число 649 из десятичной в ДВОИЧНУЮ
1.17 Переведите число 1010001 из ДВОИЧНОЙ в десятичную
1.18 Переведите число 91 из десятичной в ДВОИЧНУЮ
1.19 Переведите число 196 из десятичной в ДВОИЧНУЮ
1.20 Переведите число 10011 из ДВОИЧНОЙ в десятичную
1.21 Переведите число 101000100 из ДВОИЧНОЙ в десятичную
1.22 Переведите число 1011111000 из ДВОИЧНОЙ

2. Система счисления – ВОСЬМЕРИЧНАЯ

2.1 Переведите число 65 из десятичной в ВОСЬМЕРИЧНУЮ
2.2 Переведите число 835 из десятичной в ВОСЬМЕРИЧНУЮ
2.3 Переведите число 127 из ВОСЬМЕРИЧНОЙ в десятичную
2.4 Переведите число 1361 из ВОСЬМЕРИЧНОЙ в десятичную
2.5 Переведите число 96 из десятичной в ВОСЬМЕРИЧНУЮ
2.6 Переведите число 71 из десятичной в ВОСЬМЕРИЧНУЮ
2.7 Переведите число 782 из десятичной в ВОСЬМЕРИЧНУЮ
2.8 Переведите число 17 из ВОСЬМЕРИЧНОЙ в десятичную

2.9 Переведите число 365 из ВОСЬМЕРИЧНОЙ в десятичную
2.10 Переведите число 654 из десятичной в ВОСЬМЕРИЧНУЮ
2.11 Переведите число 54 из десятичной в ВОСЬМЕРИЧНУЮ
2.12 Переведите число 84 из десятичной в ВОСЬМЕРИЧНУЮ
2.13 Переведите число 612 из десятичной в ВОСЬМЕРИЧНУЮ
2.14 Переведите число 135 из ВОСЬМЕРИЧНОЙ в десятичную
2.15 Переведите число 91 из десятичной в ВОСЬМЕРИЧНУЮ
2.16 Переведите число 227 из десятичной в ВОСЬМЕРИЧНУЮ
2.17 Переведите число 130 из ВОСЬМЕРИЧНОЙ в десятичную
2.18 Переведите число 1146 из ВОСЬМЕРИЧНОЙ в десятичную
2.19 Переведите число 928 из десятичной в ВОСЬМЕРИЧНУЮ
2.20 Переведите число 62 из ВОСЬМЕРИЧНОЙ в десятичную
2.21 Переведите число 1061 из ВОСЬМЕРИЧНОЙ в десятичную
2.22 Переведите число 112 из ВОСЬМЕРИЧНОЙ в десятичную

3. Система счисления – ШЕСТНАДЦАТЕРИЧНАЯ

3.1 Переведите число 32 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.2 Переведите число 456 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.3 Переведите число 3E из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.4 Переведите число 95 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.5 Переведите число 494 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.6 Переведите число 4C из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.7 Переведите число 56 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.8 Переведите число 684 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.9 Переведите число 33 из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.10 Переведите число 28E из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.11 Переведите число 28B из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.12 Переведите число 23 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.13 Переведите число 617 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.14 Переведите число 115 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.15 Переведите число 379 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.16 Переведите число 14 из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.17 Переведите число 12E из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.18 Переведите число 2D из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.19 Переведите число 68 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.20 Переведите число 938 из десятичной в ШЕСТНАДЦАТЕРИЧНУЮ
3.21 Переведите число 45 из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.22 Переведите число 33A из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную
3.23 Переведите число 2C8 из ШЕСТНАДЦАТЕРИЧНОЙ в десятичную

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1:

1. Как перевести из десятичной системы счисления в любую другую?
2. Как перевести из двоичной системы счисления в любую другую?
3. Правило перевода числа из восьмеричной или шестнадцатеричной системы счисления?
4. Что такое триада и тетрада?

Лабораторная работа №2
«Кодирование и подсчет количества информации»

Цель работы:

1. Научиться измерять информацию, решать задач на определение количества и объема информации.

1 Кодирование и измерение информации

Кодирование информации – это процесс формирования определенного представления информации.

В более узком смысле под термином «кодирование» часто понимают переход от одной формы представления информации к другой, более удобной для хранения, передачи или обработки.

Компьютер может обрабатывать только информацию, представленную в числовой форме. Вся другая информация (звуки, изображения, показания приборов и т. д.) для обработки на компьютере должна быть преобразована в числовую форму. Например, чтобы перевести в числовую форму музыкальный звук, можно через небольшие промежутки времени измерять интенсивность звука на определенных частотах, представляя результаты каждого измерения в числовой форме. С помощью компьютерных программ можно преобразовывать полученную информацию, например «наложить» друг на друга звуки от разных источников.

Аналогично на компьютере можно обрабатывать текстовую информацию. При вводе в компьютер каждая буква кодируется определенным числом, а при выводе на внешние устройства (экран или печать) для восприятия человеком по этим числам строятся изображения букв. Соответствие между набором букв и числами называется кодировкой символов.

Как правило, все числа в компьютере представляются с помощью нулей и единиц (а не десяти цифр, как это привычно для людей). Иными словами, компьютеры обычно работают в двоичной системе счисления, поскольку при этом устройства для их обработки получаются значительно более простыми.

1.1 Единицы измерения информации. Бит. Байт.

Бит – наименьшая единица представления информации. Байт – наименьшая единица обработки и передачи информации.

Решая различные задачи, человек использует информацию об окружающем нас мире. Часто приходится слышать, что сообщение несет мало информации или, наоборот, содержит исчерпывающую информацию, при этом разные люди, получившие одно и то же сообщение (например, прочитав статью в газете), по-разному оценивают количество информации, содержащейся в нем. Это означает, что знания людей об этих событиях (явлениях) до получения сообщения были различными. Количество информации в сообщении, таким образом, зависит от того, насколько ново это сообщение для получателя. Если в результате получения сообщения достигнута полная ясность в данном вопросе (т.е. неопределенность исчезнет), говорят, что получена исчерпывающая информация. Это означает, что нет необходимости в дополнительной информации на эту тему. Напротив, если после получения сообщения неопределенность осталась прежней (сообщаемые сведения или уже были известны, или не относятся к делу), значит, информации получено не было (нулевая информация).

Подбрасывание монеты и слежение за ее падением дает определенную информацию. Обе стороны монеты «равноправны», поэтому одинаково вероятно, что выпадет как одна, так и другая сторона. В таких случаях говорят, что событие несет информацию в 1 бит. Если положить в мешок два шарика разного цвета, то, вытащив вслепую один шар, мы также получим информацию о цвете шара в 1 бит.

Единица измерения информации называется бит (bit) – сокращение от английских слов binary digit, что означает двоичная цифра.

В компьютерной технике бит соответствует физическому состоянию носителя информации: намагничено – не намагничено, есть отверстие – нет отверстия. При этом одно состояние принято обозначать цифрой 0, а другое – цифрой 1. Выбор одного из двух возможных вариантов позволяет также различать логические истину и ложь. Последовательностью битов можно закодировать текст, изображение, звук или какую-либо другую информацию. Такой метод представления информации называется двоичным кодированием (binary encoding).

2 Определение объема информации

Данный тип задач содержит несложные задания в одно-два действия и нацелен на проверку умений подсчета информационного объёма сообщения и количества информации.

2.1 Вероятностный подход

Количество информации в сообщении определяется объемом передаваемых новых знаний, то есть является мерой уменьшения неопределенности знаний об объекте. Если в сообщении содержится информация о том, что произошло одно из равновероятных событий, то количество информации в таком сообщении (x) зависит от общего количества вариантов событий (N):

$$2^x = N \text{ или } x = \log_2 N$$

Пример 1

Какое количество информации несет известие о том, что рулетка после вращения показала на двенадцатый сектор, если имеется 32 совершенно равных сектора.

Решение

По условию задачи есть 32 равновероятных события – выпадения одного из 32 секторов. В сообщении о выпадении конкретного сектора содержится

$x = \log_2 32 = 5$.

Ответ: 5 бит.

Пример 2

Какое количество информации содержит сообщение о том, что игрок вытащил первую карту – даму из колоды в 36 карт?

Решение

В колоде 36 карт, но в них входят по 4 карты различных мастей для каждого вида карт, в том числе и 4 дамы. Соответственно, выпадение дамы – это одно из 9 (36/4) равновероятных событий. Тогда количество информации, которое оно несет равно

$x = \log_2 9 \approx 3{,}17$.

Поскольку с точки зрения компьютерной техники бит может быть только целым числом, округляем результат вычислений до ближайшего большего целого числа.

Ответ: 4 бита.

2.2 Алфавитный подход

Алфавитный подход к определению количества информации используется в том случае, когда сообщение, содержащее информацию, закодировано с помощью определенной знаковой системы, имеющей конечный алфавит. Алфавитом здесь называется полный набор уникальных символов, используемых для записи любого сообщения в данной знаковой системе. Количество уникальных символов в этом наборе называется мощностью алфавита.

Если допустить, что все символы равновероятны, то количество информации, содержащееся в одном символе сообщения (x), зависит от мощности алфавита (N):

$x = \log_2 N$

То есть в алфавите, содержащем два символа, каждый символ несет 1 бит информации, в алфавите, содержащем 8 символов – 3 бита, а в алфавите, содержащем 256 символов – 8 бит информации.

Количество информации в сообщении в таком случае есть произведение количества информации в одном символе (информационный вес символа) на количество символов.

Для упрощения работы с описанием больших объемов информации введены следующие величины:

1 байт = 8 бит – соответствует символу языка с мощностью алфавита 256; 1 килобайт = 2^{10} байт = 1024 байта;

1 мегабайт = 2^{10} килобайт = 1024 килобайта;

1 гигабайт = 2^{10} мегабайт = 1024 мегабайта;

3 Количество информации в текстовом сообщении

Для решения задач на определение объема информации, необходимо учитывать, каким алфавитом закодирована информация, т.к. единица измерения информации зависит от конкретного алфавита. При подсчете количества информации особое внимание надо обратить на единицы измерения и применяемые таблицы кодировок (Unicode, ASCII, КОИ-8 и т.д.).

3.1 Что нужно знать?

- Все символы кодируются одинаковым числом бит[1] (алфавитный подход);
- Чаще всего используют кодировки, в которых на символ отводится 8 бит (8-битные) или 16 бит (16-битные);
- При измерении количества информации принимается, что в одном байте 8 бит, а в одном килобайте (1 кбайт) – 1024 байта, в мегабайте (1Мбайт) – 1024 кбайта;
- после знака препинания внутри (не в конце!) текста ставится пробел чтобы найти информационный объем текста *I*, нужно умножить количество символов *N* на число бит на символ *K*: $I = N * K$ две строчки текста не могут занимать 100 кбайт в памяти.

Для кодирования текстовой информации распространены два вида кодировок:

- ASCII кодировка имеет алфавит мощностью 256 символов (1 байт на символ), что позволяет использовать в сообщении буквы двух языков (английского и регионального), цифры, знаки препинания и ряд дополнительных символов;
- Unicode кодировка имеет алфавит мощностью 65536 символов (2 байта на символ), что позволяет использовать в сообщении буквы большого количества языков.

Обычно при решении задач такого вида делают две самые типичные ошибки: невнимательно читают задание и начинают считать не в нужной кодировке. Сосчитав количество символов в байтах и не найдя нужного ответа, решают, что задание содержит ошибку, и даже не пытаются перевести байты в биты или наоборот.

Необходимо помнить, что: в 1 байте 8 бит, в 1 кб (килобайте) 1024 бит, в 1 Мб (Мегабайте) 1024 кб

(килобайт).

Разберем несколько примеров решения задач такого типа.

Пример 1:

Автоматическое устройство осуществило перекодировку информационного сообщения на русском языке, первоначально записанного в 16-битном коде Unicode, в 8-битную кодировку КОИ-8. При этом информационное сообщение уменьшилось на 720 бит. Какова длина сообщения в символах?

1) 902) 45 3) 180 4) 720

Решение:

При переходе от 16-битной к 8-битной кодировке, размер сообщения в битах уменьшится вдвое. Раз оно уменьшилось на 720 бит, то длина сообщения в символах кодировки КОИ-8 составила 720 символов. *Ответ: 4.*

Пример 2:

В кодировке Unicode на каждый символ отводится два байта. Определите информационный объем слова из двадцати четырех символов в этой кодировке.

1) 384 бита; 2) 192 бита; 3) 256 бит; 4) 48 бит.

Решение:

24 * 2байта=48 байтов=384 бита. *Ответ: №1*

Пример 3:

Считая, что каждый символ кодируется 16 битами, оцените информационный объем следующей Пушкинской фразы в кодировке Unicode:

Привычка свыше нам дана: Замена счастию она.

1) 44 бита; 2) 704 бита; 3) 44 байта; 4) 704 байта.

Решение:

Данное предложение содержит 44 символа (включая точку и двоеточие), то есть в кодировке Unicode оно содержит 88 байт или 704 бита. *Ответ: № 2*

Пример 4:

В велокроссе участвуют 779 спортсменов. Специальное устройство регистрирует прохождение каждым из участников промежуточного финиша, записывая его номер с использованием минимально возможного количества бит, одинакового для каждого спортсмена. Каков информационный объем сообщения, записанного устройством, после того как промежуточный финиш прошли 280 велосипедистов?

1) 280 бит 2) 779 бит 3) 280 байт 4) 350 байт.

Решение:

Справедлива оценка $29 < 779 < 210$, поэтому для равномерного двоичного кодирования номера каждого спортсмена требуется 10 бит информации. Тогда информационный объем сообщения, записанного устройством, после того как промежуточный финиш прошли 280 велосипедистов, составляет 2800 бит=350 байт. *Ответ: №4.*

Пример 5:

Одна ячейка памяти «троичной ЭВМ» (компьютера, основанного на использовании троичной системы счисления) может принимать одно из трех возможных состояний. Для хранения некоторой величины отвели 6 ячеек памяти. Сколько различных значений может принимать эта величина?

1) 182) 64 3) 216 4) 729.

Решение:

Мощность алфавита троичной ЭВМ N равно 3. Для 6 ячеек памяти количество различных значений оценивается величиной N6 = 36 = 729. *Ответ: №4.*

4 Количество информации в растровом изображении

Растровый способ описания графики представляет собой описание прямоугольной матрицы точек (пикселей), каждый из которых может быть одного цвета. Количество цветов, которыми может быть окрашен пиксель, зависит от количества бит, отведенных на его описание (глубины цвета). В этом случае количество цветов можно трактовать как мощность алфавита знаковой системы. Так, глубина цвета 1 бит позволяет закрасить каждый пиксель изображения в один из двух цветов, глубина цвета 8 бит – 256 цветов, глубина цвета 24 бита – приблизительно 16,7 млн. цветов, и т.д. Общий информационный объем изображения в таком случае равен произведению количества пикселей в матрице (ширина на высоту) и количества бит, отводимых на каждый пиксель.

Пример 1:

Растровое изображение размером 64x64 пикселей занимает 2 килобайта памяти. Укажите максимальное число цветов в палитре изображения.

Решение

Если матрица точек – 64x64 пикселя, значит общее количество пикселей равно $2^6 \cdot 2^6 = 2^{12}$ пикселей. 2 килобайта – это $2 \cdot 2^{10} \cdot 8$ бит = $2^1 \cdot 2^{10} \cdot 2^3$ бит = 2^{14} бит. Следовательно, на один пиксель приходится $2^{14} / 2^{12} = 2^2 = 4$ бита, что соответствует мощности алфавита в $2^4 = 16$ символов (цветов). *Ответ: 16 цветов.*

5 Измерение объема информации, передаваемой через канал с фиксированной пропускной способностью

При решении задач этого типа необходимо лишь не запутаться в размерности.

Пример 1:

Скорость передачи данных через ADSL-соединение равна 128000 бит/с. Передача файла через это соединение заняла 2 минуты. Определите размер файла в килобайтах.

Решение:

128000*120=15360000(бит)

Ответ: 1875.(15360000/8*1024 = 1875 (Килобайт)

Задание

Задача 1. Вычислить, какой объем памяти компьютера потребуется для хранения одной страницы текста на английском языке, содержащей 2400 символов.

Задача 2. В течение 5 секунд было передано сообщение, объем которого составил 375 байт. Каков размер алфавита, с помощью которого записано сообщение, если скорость передачи составила 200 символов в секунду?

Задача 3. Считая, что каждый символ кодируется одним байтом, определите, чему равен информационный объем следующего высказывания Жан-Жака Руссо: **Тысячи путей ведут к заблуждению, к истине – только один.**

Задача 4. Для записи текста используются только строчные буквы русского алфавита и пробел для разделения слов. Какой информационный объем имеет текст, состоящий из 2000 символов?

Задача 5. Получено сообщение, информационный объем которого равен 32 битам. чему равен этот объем в байтах?

Задача 6. Объем информационного сообщения 12582912 битов выразить в килобайтах и мегабайтах.

Задача 7. Компьютер имеет оперативную память 512 Мб. Количество соответствующих этой величине бит больше:

Задача 8. Определить количество битов в двух мегабайтах, используя для чисел только степени 2.

Задача 9. Сколько мегабайт информации содержит сообщение объемом 223бит?

Задача 10. Один символ алфавита "весит" 4 бита. Сколько символов в этом алфавите?

Задача 11. Каждый символ алфавита записан с помощью 8 цифр двоичного кода. Сколько символов в этом алфавите? Найти N - ?

Задача 12. Алфавит русского языка иногда оценивают в 32 буквы. Каков информационный вес одной буквы такого сокращенного русского алфавита? Найти: i- ?

Задача 13. Алфавит состоит из 100 символов. Какое количество информации несет один символ этого алфавита? Найти: i- ?

Задача 14. У племени «чичевоков» в алфавите 24 буквы и 8 цифр. Знаков препинания и арифметических знаков нет. Какое минимальное количество двоичных разрядов им необходимо для кодирования всех символов? Учтите, что слова надо отделять друг от друга! Найти: i- ?

Задача 15. Книга, набранная с помощью компьютера, содержит 150 страниц. На каждой странице — 40 строк, в каждой строке — 60 символов. Каков объем информации в книге? Ответ дайте в килобайтах и мегабайтах. Найти: I- ?

Задача 16. Информационный объем текста книги, набранной на компьютере с использованием кодировки Unicode, — 128 килобайт. Определить количество символов в тексте книги. Найти: K- ?

Задача 17. Информационное сообщение объемом 1,5 Кб содержит 3072 символа. Определить информационный вес одного символа использованного алфавита. Найти: i- ?

Задача 18. Сообщение, записанное буквами из 64-символьного алфавита, содержит 20 символов. Какой объем информации оно несет? Найти: I- ?

Задача 19. Сколько символов содержит сообщение, записанное с помощью 16-символьного алфавита, если его объем составил 1/16 часть мегабайта? Найти: K- ?

Задача 20. Объем сообщения, содержащего 2048 символов, составил 1/512 часть мегабайта. Каков размер алфавита, с помощью которого записано сообщение?

Задача 21. Сообщение, записанное буквами 64-символьного алфавита, содержит 20 символов. Какой объем информации оно несет?

Задача 22. Жители планеты Принтер используют алфавит из 256 знаков, а жители планеты Плоттер — из 128 знаков. Для жителей какой планеты сообщение из 10 знаков несет больше информации и на сколько?

Задача 23. Для кодирования нотной записи используется 7 значков-нот. Каждая нота кодируется одним и тем же минимально возможным количеством бит. Чему равен информационный объем сообщения, состоящего из 180 нот?

Задача 24. Цветное растровое графическое изображение, палитра которого включает в себя 65 536 цветов, имеет размер 100X100 точек (пикселей). Какой объем видеопамяти компьютера (в Кбайтах) занимает это изображение в формате BMP?

Задача 25. В велокроссе участвуют 119 спортсменов. Специальное устройство регистрирует прохождение каждым из участников промежуточного финиша, записывая его номер с использованием минимально возможного количества бит, одинакового для каждого спортсмена. Каков информационный объем сообщения, записанного устройством, после того как промежуточный финиш прошли 70 велосипедистов?

Задача 26. Словарный запас некоторого языка составляет 256 слов, каждое из которых состоит точно из 4 букв. Сколько букв в алфавите языка?

Задача 27. Информационное сообщение объемом 1,5 килобайта содержит 3072 символа. Сколько символов содержит алфавит, с помощью которого было записано это сообщение?

Задача 28. Мощность алфавита равна 64. Сколько Кбайт памяти потребуется, чтобы сохранить 128 страниц текста, содержащего в среднем 256 символов на каждой странице?

Задача 29. Считая, что один символ кодируется одним байтом, подсчитать в байтах количество информации, содержащееся в фразе: «Терпение и труд всё перетрут».

Задача 30. Алфавит племени содержит всего 8 букв. Какое количество информации несет одна буква этого алфавита?

1) 8 бит 2) 1 байт 3) 3 бита 4) 2 бита

Задача 31. Если вариант теста в среднем имеет объем 20 килобайт (на каждой странице теста 40 строк по 64 символа в строке, 1 символ занимает 8 бит), то количество страниц в тесте равно:

1) 10 2) 16 3) 4 4) 8

Задача 32. В пяти килобайтах:

1) 5000 байт 2) 5120 байт 3) 500 байт 4) 5000 бит

Задача 33. Сколько байт в 32 Гбайт?

1) 235 2) 16220 3) 224 4) 222

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2:

1. Что такое кодирование?
2. Чем отличается вероятностный подход от алфавитного?
3. Формула измерения количества информации.
4. Наименьшая единица количества информации?
5. Как перевести Мбайт в бит?

Лабораторная работа №3
«Установка Windows на Virtual Box»

Цель работы:

1. Познакомиться с понятием виртуальной машины;
2. Освоить работу на ВМ;
3. Научиться создать эмуляцию современных ОС;

1 Что такое виртуальная машина

Виртуальная машина – это программа, которая эмулирует реальный (физический) компьютер со всеми его компонентами (жёсткий диск, привод, BIOS, сетевые адаптеры и т.д.). На такой виртуальный компьютер можно установить, например, операционную систему, драйверы, программы и т.д. Таким образом, вы можете запустить на своем реальном компьютере еще несколько виртуальных компьютеров, с такой же или другой операционной системой. Вы можете без проблем осуществить обмен данными между вашим реальным и виртуальным компьютером.

Зачем нужна виртуальная машина?

Не каждому пользователя ПК нужна виртуальная машина, но продвинутые пользователи довольно часто используют ее. Виртуальную машину используют для различных целей и задач:

- Установка второй/другой операционной системы;
- Тестирование программного обеспечения;

- Безопасный запуск подозрительных программ;
- Эмуляция компьютерной сети;
- Запуск приложений, которые нельзя запустить из вашей операционной системы.

Например, на вашем реальном компьютер может быть установлена операционная система Windows 7, а на виртуальную машину можно поставить и протестировать другие операционные системы, такие как Windows XP/8/10 или Linux, а также устанавливать и тестировать различные программы и утилиты.

Виртуальная машина – это очень удобно, т.е. можно тестировать различные незнакомые программы в виртуальной среде, чтобы не подвергать опасности и не захламлять основной физический компьютер.

Существует большое количество различных программ для создания и управления виртуальными компьютерами.

- **Виртуальная машина VirtualBox.**

VirtualBox – это бесплатная виртуальная машина, на которую можно установить все самые популярные операционные системы. VirtualBox поддерживает работу с Windows, Linux, FreeBSD, Mac OS.

VirtualBox поддерживает как 32-х, так и 64-разрядные версии операционных систем. VirtualBox поддерживает работу с виртуальными компьютерами, созданными в платной программе **VMware Workstation**.

Настройка и работа с VirtualBox очень удобная и простая. Программа довольна производительна и стабильна. Она обладает широким функционалом, удельным интерфейсом и совершенно бесплатна.

VirtualBox — это лучшая виртуальная машина для домашнего использования.

- **Виртуальная машина VMware.**

VMware – это наиболее известная и распространенная виртуальная машина. VMware, как правило, используют для работы крупные площадки или корпорации.

Виртуальная машина VMware поставляется в двух видах: **Workstation** и **Player**. **VMware Workstation** отличная и мощная машина, но платная. VMware Player – бесплатная урезанная версия VMware Workstation.

VMware Workstation поддерживает работу с 32 и 64-битными системами, USB 3.0, установку различных операционных систем.

VMware Workstation безусловно лучшая виртуальная машина, которой пользуются крупные компании, но ее стоимость снижает популярность среди рядовых пользователей.

- **Виртуальная машина Microsoft Virtual PC.**

Microsoft Virtual PC – это еще одна бесплатная виртуальная машина. Она обладает широким функционалом и удобным интерфейсом, но у нее есть один большой недостаток – она работает только с операционными системами Windows. На ней нельзя запустить Linux или Mac OS.

Подведя итог, хотелось бы отметить, что все-таки для домашнего использования лучше всего подходит VirtualBox.

Сравнение некоторых VM:

Название	Процессор машины-носителя	ОС машины-носителя	Официально поддерживаемые гостевые ОС	Поддержка любой ОС	Лицензия	Скорость работы гостевой ОС в сравнении с ОС носителя
DOSBox	Intel x86, AMD64, SPARC, PowerPC, Alpha, MIPS	Linux, Windows, Mac OS Classic, Mac OS X, BeOS, FreeBSD, OpenBSD, Solaris, QNX, IRIX, Kolibri, Android	Внешне эмулирует оболочку DOS	Нет	GPL	Крайне низкая. Скорость работы никак не связана с тем, какое приложение исполняется
KVM	Процессор Intel/AMD с поддержкой аппаратной виртуализации	Linux	Linux, HURD, Windows, xBSD, Darwin, QNX, MINIX, MS DOS, Free DOS, Solaris[1]	Нет	GPL2	Близка к производительности системы
PearPC	x86, AMD64, PowerPC	Windows, Linux, OS X, NetBSD	OS X, Darwin, Linux	Есть	GPL	10 % производительности системы-носителя
QEMU с модулем kqemu	Intel x86, AMD64	Linux, FreeBSD, Windows	Список постоянно меняется	Есть	GPL	Близка к производительности системы-носителя
QEMU с модулем qvm86	x86	Linux,NetBSD, Windows	Список постоянно меняется	Есть	GPL	Близка к производительности системы-носителя
Virtual PC2007	Intel x86, x64	Windows Vista (Business, Enterprise,	DOS, Windows, OS/2, Linux (SUSE, Xubuntu),	Есть	Проприетарная (бесплатная с	Практически без потерь, если используются

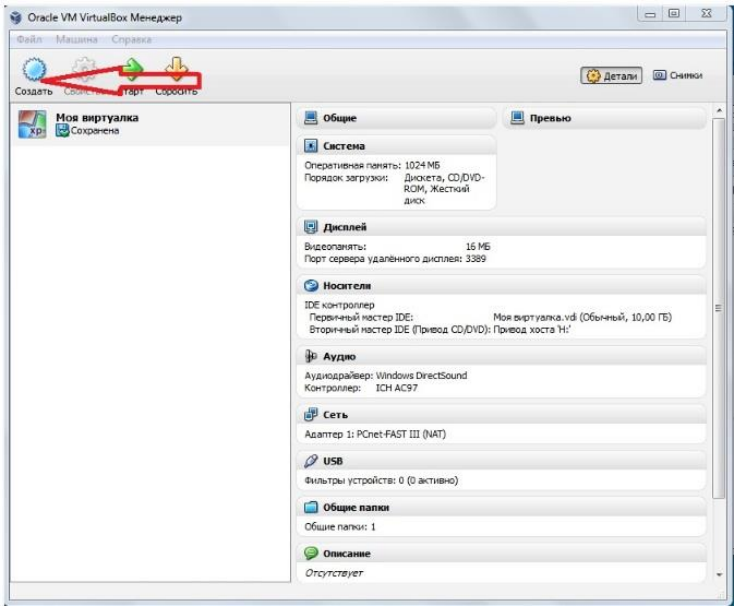
Название	Процессор машины-носителя	ОС машины-носителя	Официально поддерживаемые гостевые ОС	Поддержка любой ОС	Лицензия	Скорость работы гостевой ОС в сравнении с ОС носителя
		Ultimate), XP Pro, XP Tablet PC Edition	OpenSolaris (Belenix)		июля 2006 года)	расширения Virtual Machine additions
VirtualBox	Intel x86, AMD64	MS Windows, Linux, Solaris, OpenSolaris, Mac OS X, FreeBSD	DOS, OS/2, MS Windows, Linux, Solaris, FreeBSD, NetBSD, Netware, QNX, L4, Mac OS X	Есть	Свободная и проприетарная версии (GPL, PUEL)	Практически без потерь, если используются расширения
VMware Player	Intel x86, AMD64	Windows, Linux	DOS, Windows, Linux, Netware, Virtual Appliances	Есть	Проприетарная (Free)	Практически без потерь, если используются расширения
VMwareWorkstation 5.5	Intel x86, AMD64	Windows, Linux	DOS, Windows, Linux, FreeBSD, Netware, Solaris, Virtual Appliances	Есть	Проприетарная	Практически без потерь, если используются расширения

1.1 Как создать виртуальную машину VirtualBox

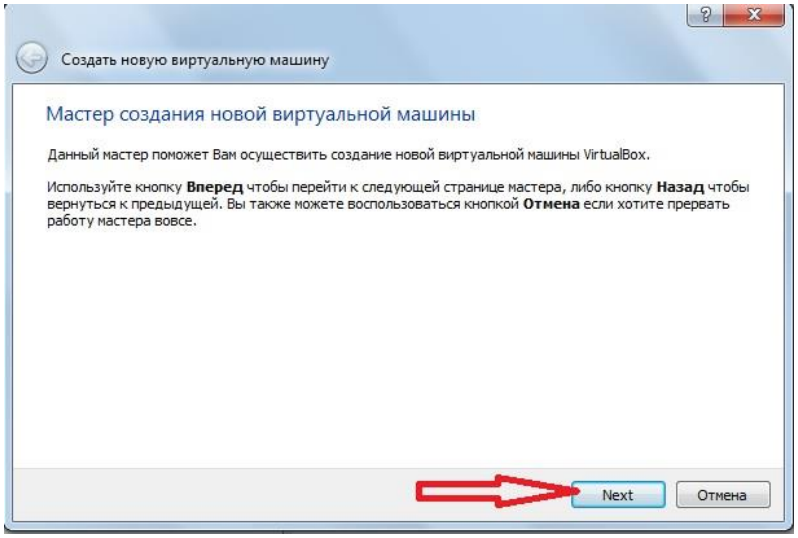
После того, как скачали программу по созданию бесплатной виртуальной машины, ее нужно установить. Устанавливается программа виртуализации для создания виртуальных машин на основную ОС.

Создается новая виртуальная машина в **Менеджере Oracle VM VirtualBox**.

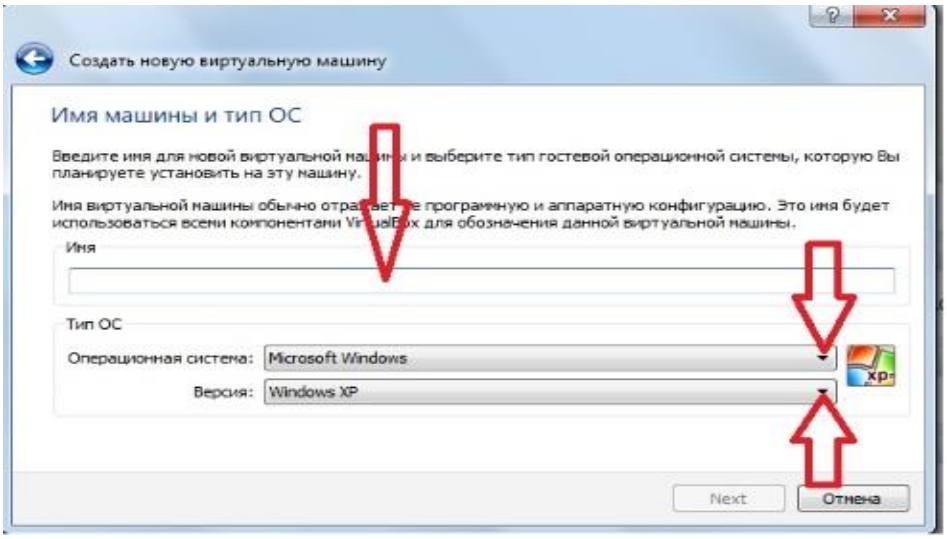
Итак, запускаем программу по созданию бесплатных виртуальных машин VirtualBox и щелкаем в **Менеджере** по созданию и управлению виртуальными машинами по кнопке ***Создать***.



Появится диалоговое окно **Мастер создания новой виртуальной машины**. Знакомимся с информацией и щелкаем кнопку **Следующая** или

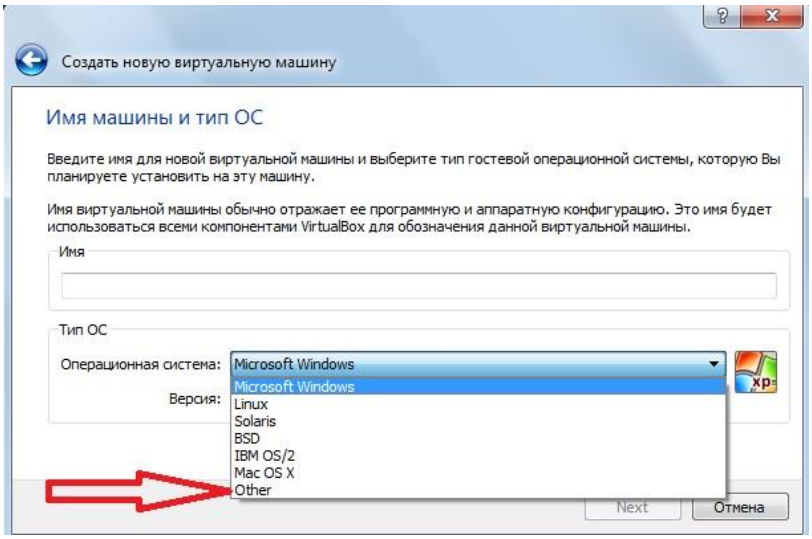


В следующем окне мы задаем имя создаваемой виртуальной машины **VirtualBox** и выбираем какую операционную систему мы будем в последствии устанавливать на созданную виртуальную машину.

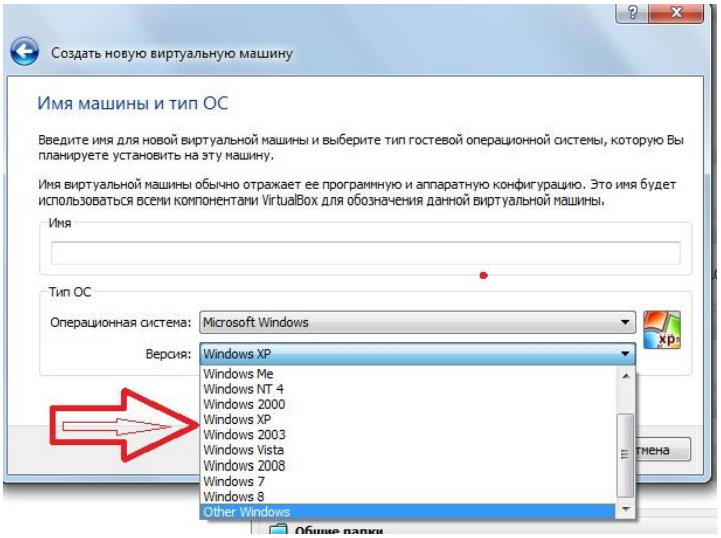


Для выбора ОС нужно щелкнуть на треугольнике в окошке ***Операционные системы***. Откроется выпадающий список, из которого можно выбрать систему, предполагаемую к установке на виртуальную машину.

Если с выбором ОС еще не определились, то можно выбрать **Other –Другую**.



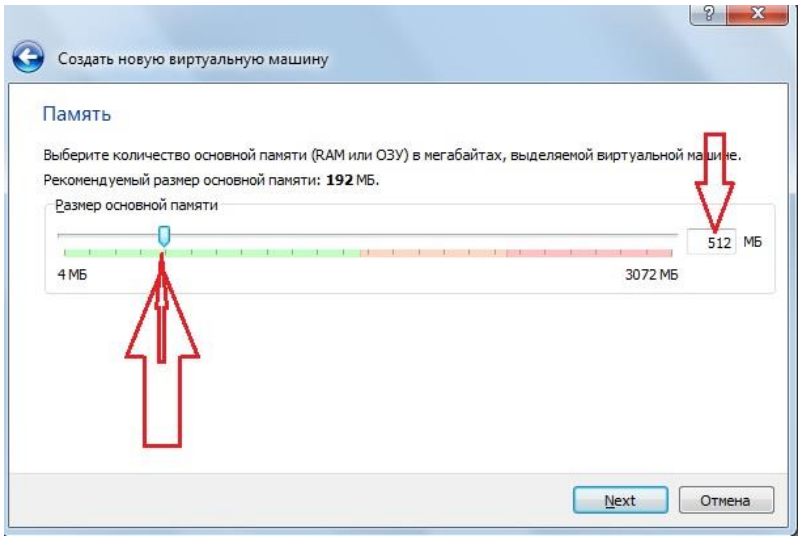
Если вы выбрали операционную систему для установки на создаваемую виртуальную машину **Microsoft Windows**, то в окошке ***Версия*** нужно будет выбрать версию ОС.



При выборе следует иметь ввиду, что каждая ОС и ее версии требуют разное количество ресурсов для комфортной работы, в частности размер минимальной оперативной памяти.

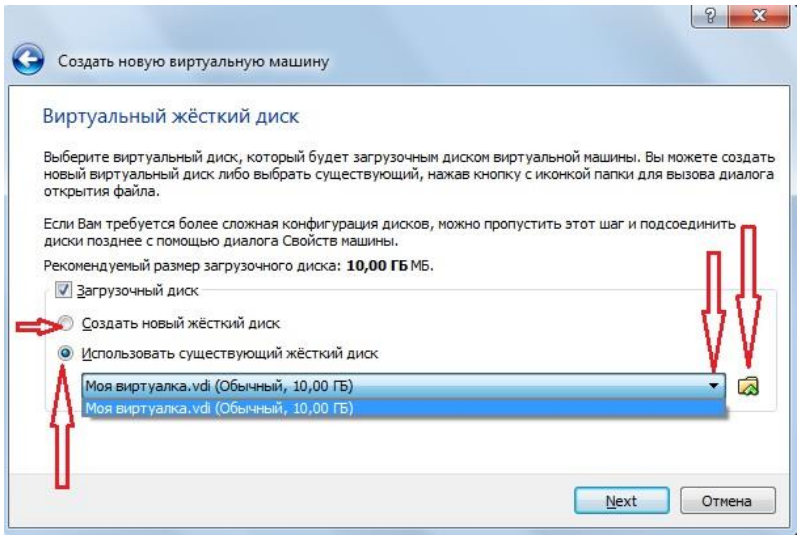
Определившись с ОС для создаваемой бесплатной виртуальной машины VirtualBox, щелкаем кнопку **Next – Следующий**.

В следующем окне мы выбираем количество оперативной памяти, выделяемой виртуальной машине. Причем, если мы предварительно выбрали операционную систему для создаваемой машины, то здесь будет рекомендован минимальный размер основной памяти. Меньше выбирать не желательно, а больше это уже в зависимости от мощности вашего компьютера. Выбор осуществляется передвижением ползунка или введением цифр в окошко.



Определившись с оперативной памятью, щелкаем **Далее.**

На следующем шаге мы выбираем размер загрузочного жесткого диска для создаваемой виртуальной машины.



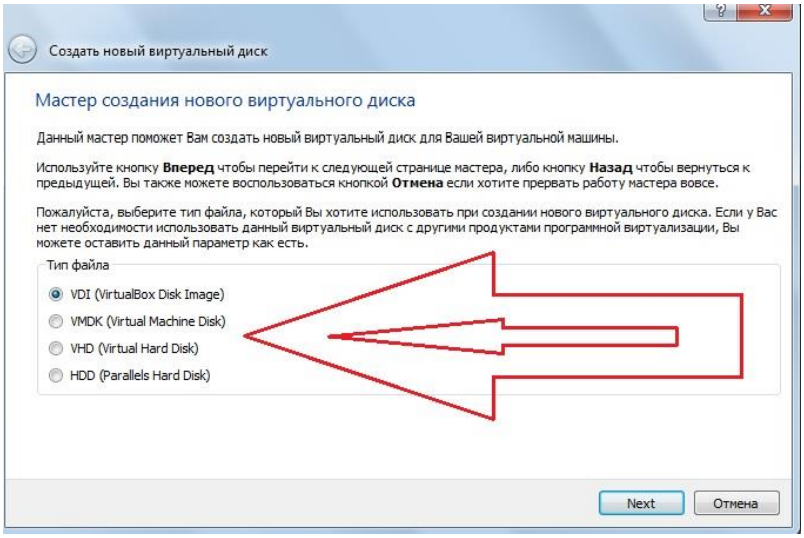
Птичка в чекбоксе ***Загрузочный диск*** стоит по умолчанию.

Здесь у нас есть выбор или создать новый жесткий диск или использовать существующий.

Существующий диск можно выбрать из выпадающего списка или щелкнув по иконке папки выбрать файл с образом диска на вашем компьютере.

Так как мы создаем первую виртуальную машину, то и диск будем создавать новый, выбрав радиокнопку – ***Создать новый жесткий диск*** и щелкнув **Next.**

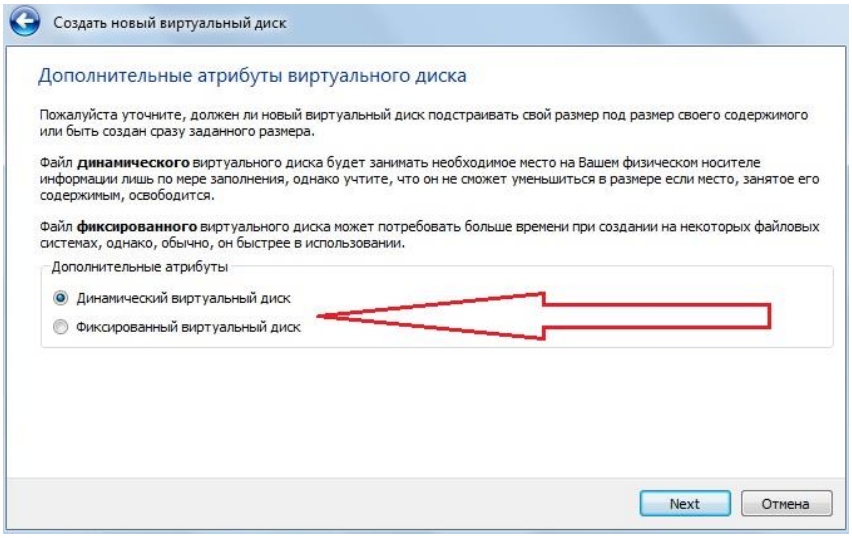
На следующем шаге откроется диалоговое окно **Мастера создания нового виртуального диска** для вашей виртуальной машины.



- VDI (VirtualBox Disk Image) — формат диска VirtualBox
- VMDK (Virtual Machine Disk) — формат диска VMware
- VHD (Virtual Hard Disk) — формат диска Microsoft
- HDD (Parallels Hard Disk) — формат диска Parallels Workstation
- QED (QEMU enhanced disk) — формат для QEMU/KVM
- QCOW (QEMU Copy-On-Write) — формат для QEMU (qcow2)

Здесь вам нужно будет выбрать тип файла, который будет использован при создании виртуального жесткого диска. Иначе говоря, нужно выбрать с каким расширением будут сохраняться файлы создаваемой виртуальной машины.

По умолчанию выбран тип файла **vdi.** Файлы с таким расширением свойственны для **VM VirtualBox.** Если вы предполагаете использовать создаваемый виртуальный диск с программами виртуализации других производителей, то нужно выбрать соответствующее расширение. В противном случае оставьте все как есть и переходите к следующему шагу, где нужно будет выбрать тип виртуального диска для создаваемой виртуальной машины.



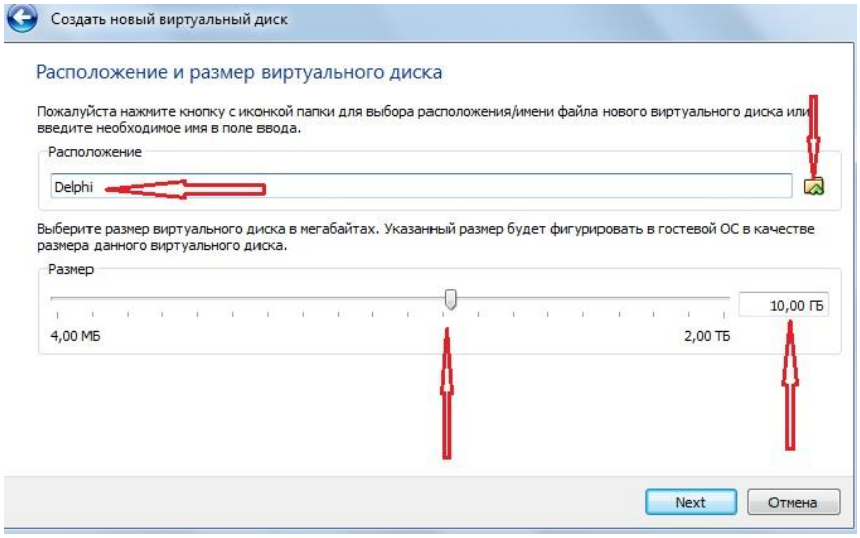
На данном этапе нужно определиться с типом создаваемого виртуального диска. Предлагается выбор между динамическим и фиксированным виртуальным диском.

Динамический диск имеет свойство увеличиваться по мере заполнения информацией. Но нужно иметь ввиду, что действие это однонаправленное. Если размер диска увеличился по мере роста информации, а затем часть информации вы удалили, то диск автоматически в размере не уменьшится, а останется прежнего размера.

Фиксированный диск сразу же обладает заданными размерами и к его преимуществу относится более высокое быстродействие по сравнению с динамическим.

Но в принципе можно для начала выбрать любой. Они оба хорошо работают. Впоследствии можно будет параметры изменить по своему вкусу.

Выбрав тип диска, переходим к следующему шагу, где должны будем выбрать расположение и размер виртуального диска.



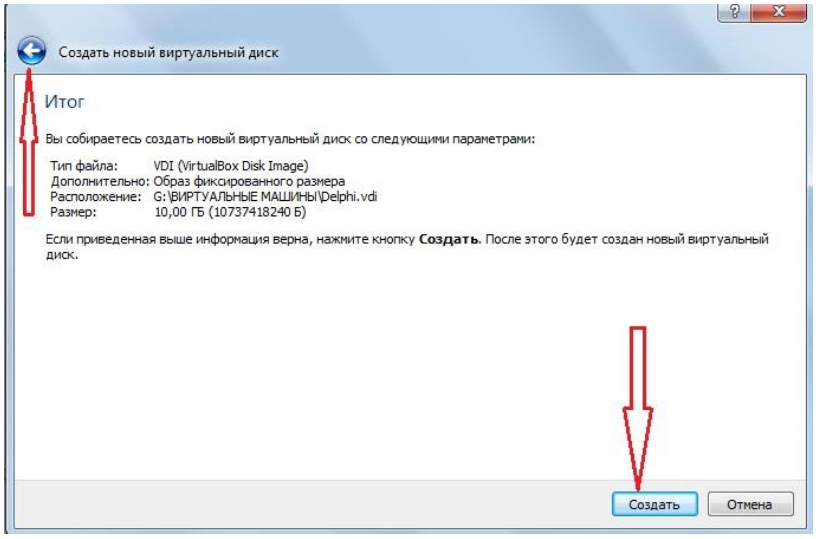
Здесь вам нужно выбрать имя и место расположения создаваемой виртуальной машины. Иными словами, место расположения папки в которой будут храниться файлы создаваемой виртуальной машины. По умолчанию имя совпадает с названием машины, которое задавалось вначале, но здесь при желании его можно изменить.

Место расположения виртуальной машины можно выбрать, щелкнув по иконке папки.

Удобнее не размещать создаваемый диск виртуальной машины на основном системном диске. В этом случае при переустановке основной ОС, созданная виртуальная машина будет сохранена. Чтобы ее запустить достаточно будет переустановить программу виртуализации, а не создавать машину заново.

Далее вам нужно выбрать размер создаваемого виртуального диска, перемещая ползунок или вводя цифры в окошко.

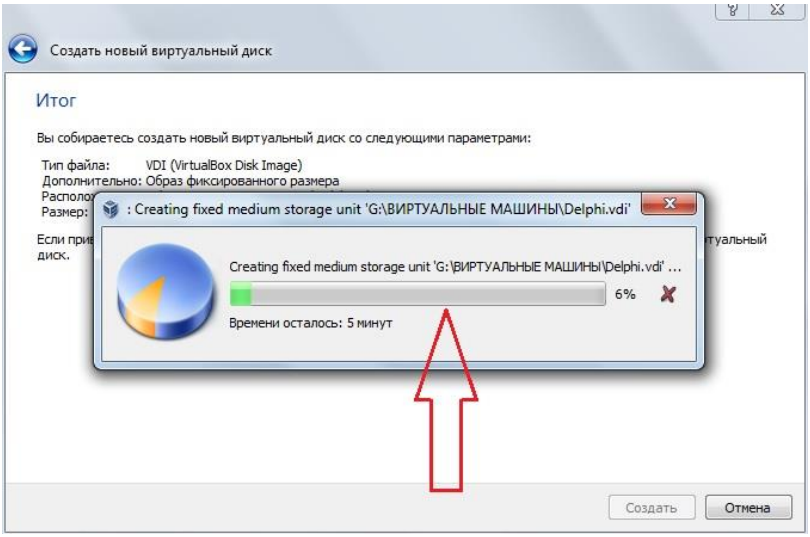
Следующий шаг итоговый. Здесь можно проверить введенную информацию и если все устраивает, создать новую виртуальную машину.



Если же выбранные параметры вас не устраивают, можно будет вернуться назад, щелкнув кнопке возврата.

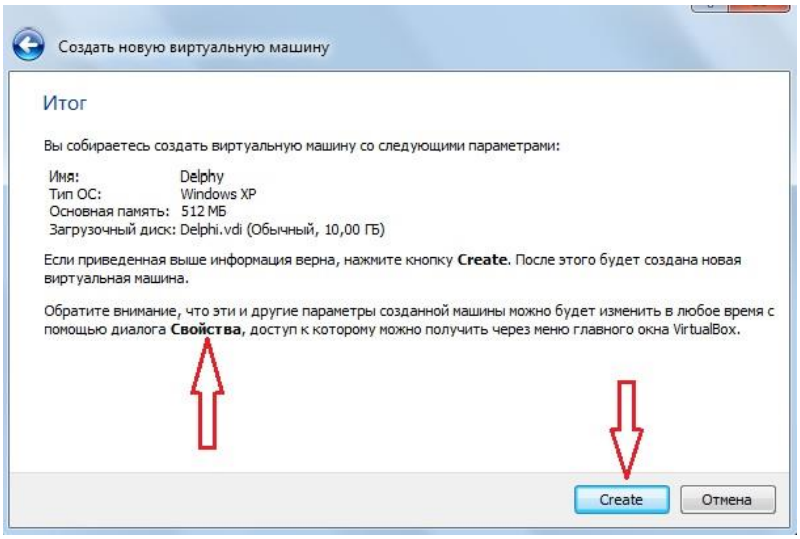
Следует иметь ввиду, что возможность возврата назад предусмотрена на любой стадии создания виртуальной машины, что очень удобно.

Щелкнув после проверки введенной информации на кнопке Создать, вы переходите к последней стадии создания виртуальной машины.

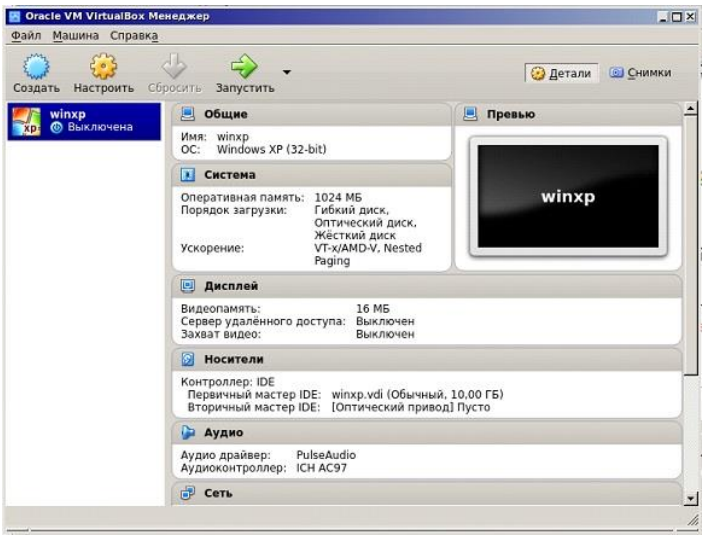


Процесс создания виртуальной машины **VirtualBox** займет некоторое время, но, впрочем, не очень продолжительное.

По окончании создания виртуального диска для виртуальной машины высветится итоговое диалоговое окно.



Здесь снова дается возможность ознакомиться, с какими параметрами будет создана виртуальная машина. И предлагается подтвердить их, нажав кнопку **Create**, после чего новая виртуальная машина будет создана.

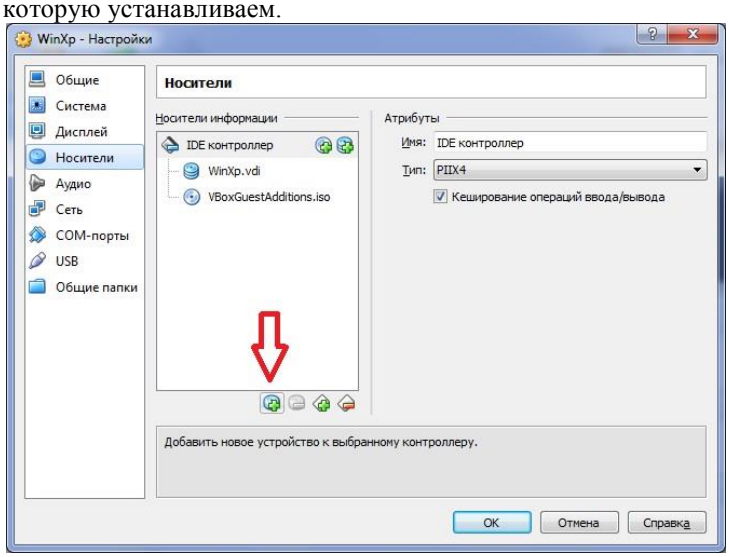


Обратите внимание на пункт, в котором говорится, что впоследствии параметры созданной виртуальной машины всегда можно изменить через диалоговое окно **Свойства** меню главного окна VirtualBox. Так что можно

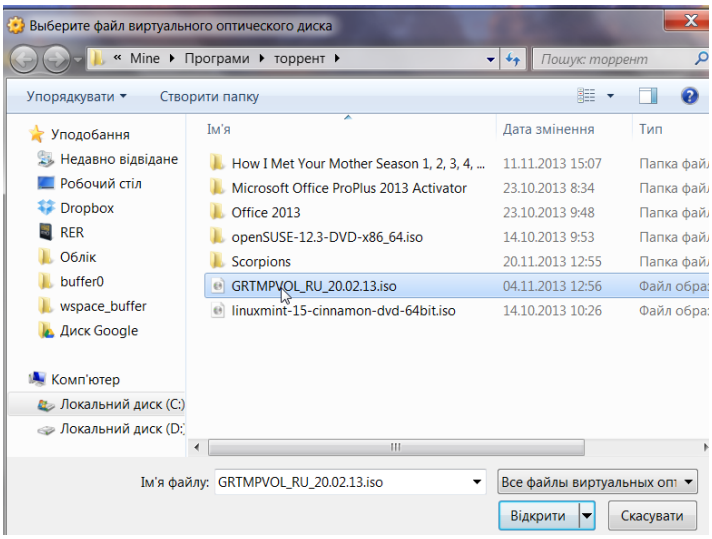
смело создавать машину с предварительными параметрами, а затем подстраивать ее под насущные нужды. Бесплатная, но очень хорошая, программа по созданию виртуальных машин VirtualBox позволяет сделать это в любое время.

Но нужно иметь ввиду, что изменять параметры можно, когда виртуальная машина находится в режиме **Выключено**, а не **Сохранено**.

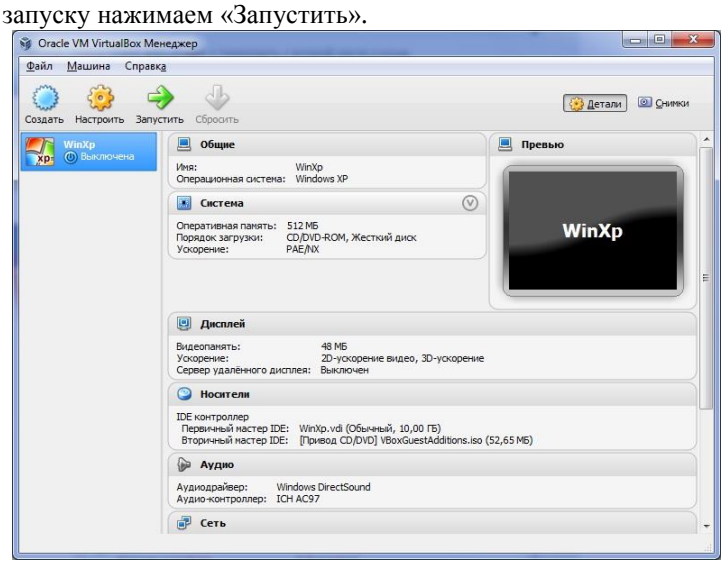
Заходим на вкладку «Носители». Добавляем новое устройство «Добавить привод оптических дисков» и выбираем образ системы, которую устанавливаем.



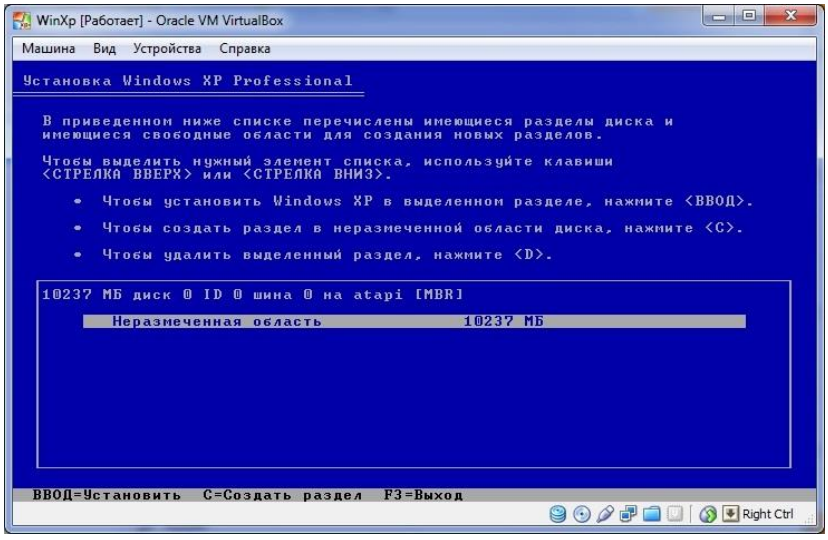
Откроется окно поиска образа. Вы должны найти ISO-файл XP. По сути, файл ISO - это и есть установочный диск для виртуальной машины.

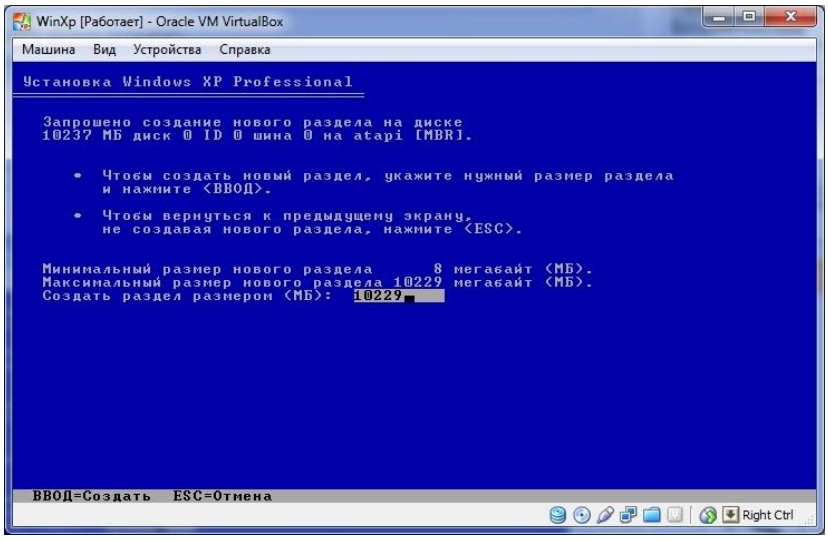


Машина готова к запуску нажимаем «Запустить».

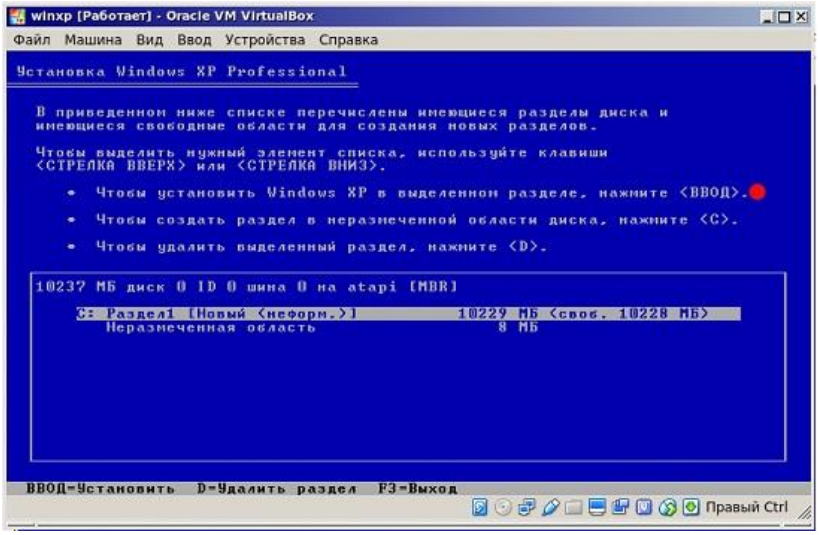


Запускается установщик Windows XP нажимаем «Enter». Попадаем в мастер раздела диска, Скорее всего, по умолчанию вы выбрали 10 Гб жесткого диска, поэтому сейчас система показывает неразмеченную область 10237 Мб., жмем «С».

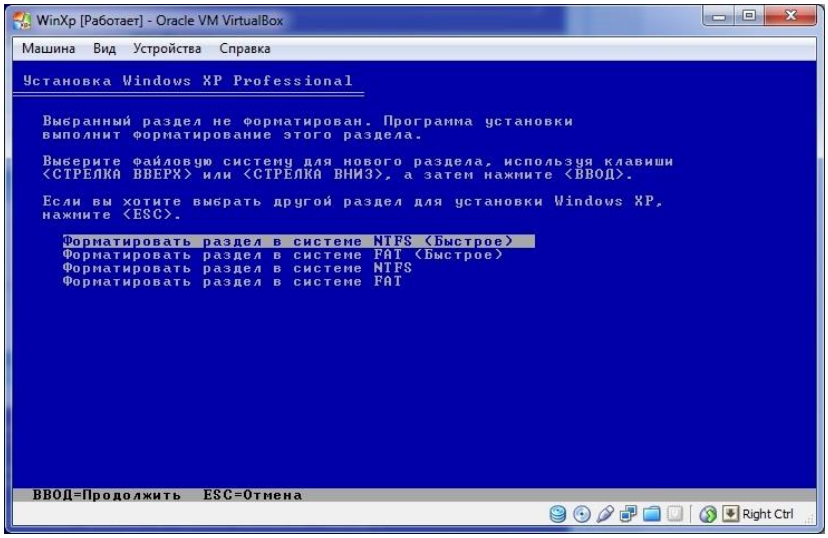




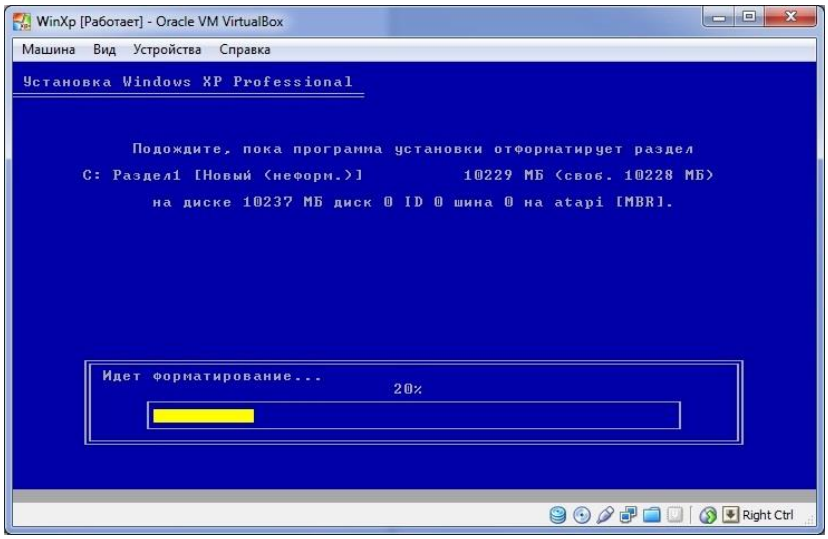
и нажимаем «Enter». Внизу экрана вы видите команды. Нажмите «С», чтобы создать раздел. Или «Enter», чтобы продолжить загрузку в выбранный раздел



Теперь система предлагает форматировать созданный раздел**.** Выбираем курсором «Форматировать раздел в системе NTFS <Быстрое>« и жмем «Enter».

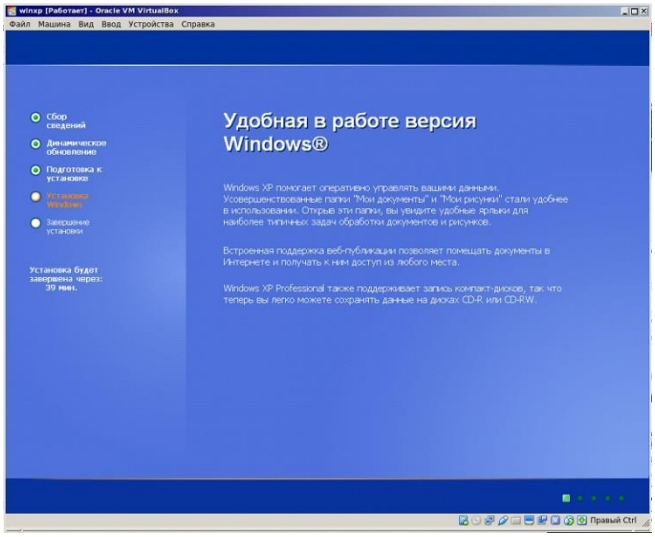


Ждем пока идет форматирование диска.



Далее установщик будет копировать ваши файлы на виртуальный жёсткий диск и после копирование перезагрузка виртуальной машины.

Начало установки самой Windows XP.



Далее начнется форматирование виртуального жесткого диска, копирование установочных файлов и перезагрузка, в момент которой нажимать ни на какие кнопки НЕ СЛЕДУЕТ. Только так гарантированно откроется, собственно, установка ОС на Virtualbox, а не начнется по новой подготовка к установке, как на скриншотах.

После завершения установки, В идеале после перезагрузки вы увидите примерно такое окно:

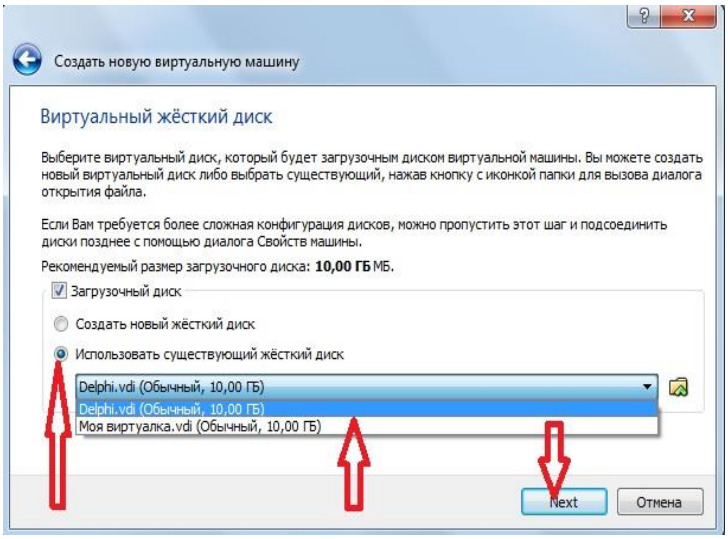
Извлеките образ диска, с которого мы делали установку.

Нажимаем на вкладку «Устройства» переходим на «Приводы оптических дисков» и выбираем «Изъять диск из привода». Установка Windows XP завершена.

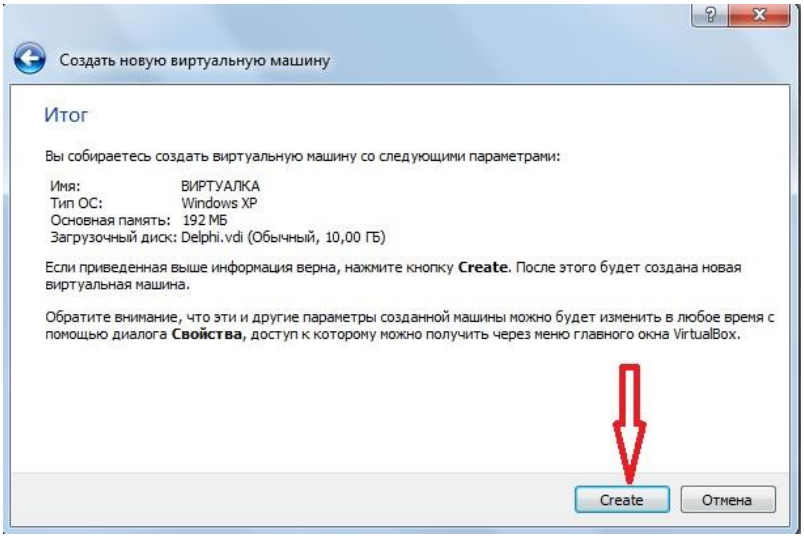
Стоит отметить только один момент. Окошко Virtualbox может захватывать курсор мыши (не всегда) и он становится недоступным для других действий. Чтобы «вернуть» его зажмите правый Ctrl на клавиатуре и пощелкайте левой кнопкой мыши.

1.2 Создание виртуальной машины VirtualBox с использованием существующего диска

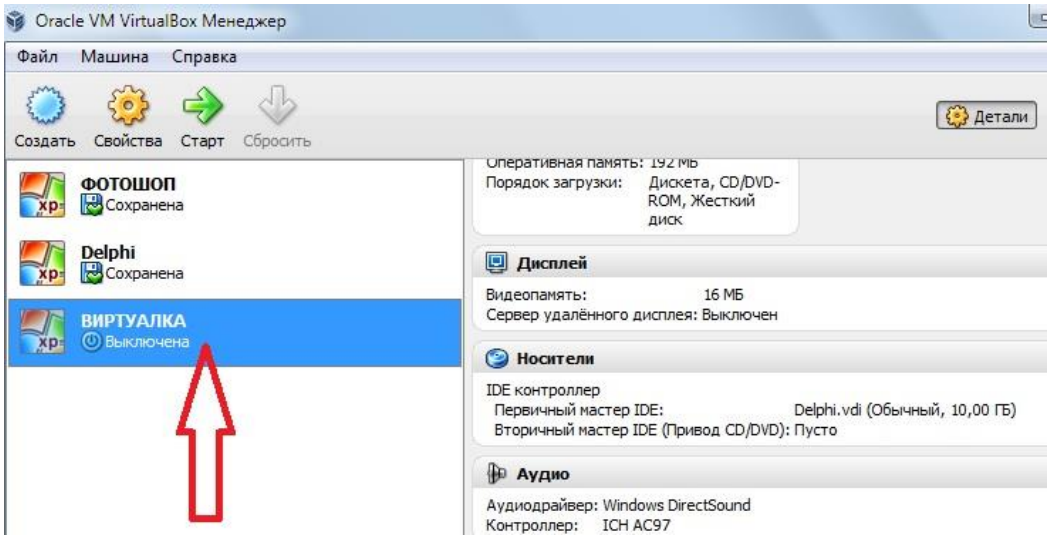
Можно значительно упростить и ускорить создание новой виртуальной машины VirtualBox Windows, если в процессе создания указать **Использовать существующий жесткий диск**.



В этом случае появится возможность указать в выпадающем списке, какой диск будет использован при создании виртуальной машины. Это диски уже существующих машин. Щелкнув кнопку Next, сразу же переходим к итоговому окну. Здесь подтверждаем предлагаемые параметры и щелкаем кнопку Создать (Create).



Виртуальная машина тут же будет создана и появится в списке существующих машин.



Причем созданная машина будет с установленной системой и программами той виртуальной машины, диск которой был использован. Практически будет создана копия существующей машины.

Нужно иметь ввиду, что эти две машины будут использовать один и тот же диск. Изменения внесенные на одной машине, будут отражены на второй.

После создания виртуальной машины нужно будет произвести некоторую настройку ее, установить операционную систему и Дополнения гостевой ОС, а затем создать общие папки, позволяющие обмениваться информацией между основной ОС и виртуальной машиной.

1.3 Как создать общую папку VirtualBox

Создание общей папки VirtualBox начинается с того, что такая папка создается на основной ОС.

Делается это обычным способом. То есть создается обычная папка через функцию **Создать**.

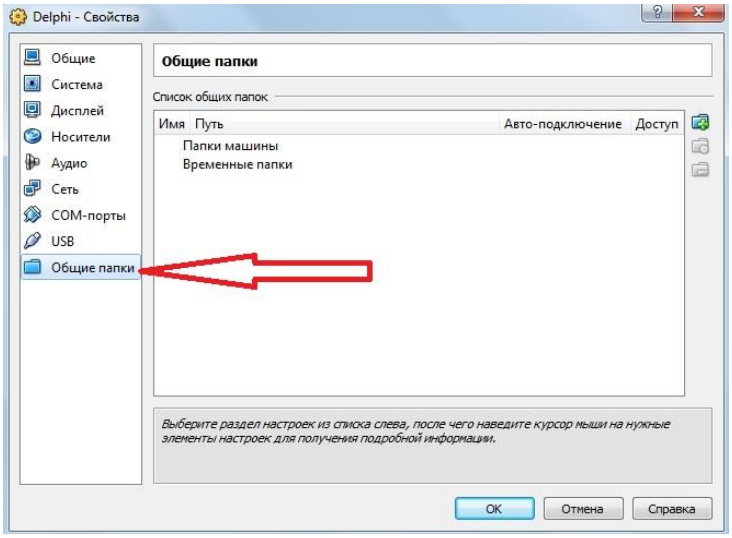
Итак, предварительно создаем папку на основной машине. В эту папку впоследствии мы будем сохранять, перемещать или копировать файлы из основной ОС для доступа к ним из виртуальной машины. Причем процесс этот двунаправленный, т.е. информация, созданная на виртуальной машине VirtualBox и сохраненная в созданной общей папке будет доступна из основной ОС.

Даем папке удобное, понятное название, например, **ПАПКА ДЛЯ ВИРТУАЛКИ**. Причем нет необходимости объявлять созданную папку общей т.е. открывать к ней общий доступ. Это обычная папка. Кстати, можно воспользоваться уже существующей папкой, а не создавать новую. Это не имеет значения. Результат будет идентичным.

После того, как создали будущую общую папку для машины VirtualBox Windows на основной ОС, переходим в саму виртуальную машину. Дальнейшее создание и настройка общей папки VirtualBox осуществляется в непосредственно здесь.

Запускаем виртуальную машину VirtualBox (ранее созданную), в Главном меню переходим *Машина – Свойства*.

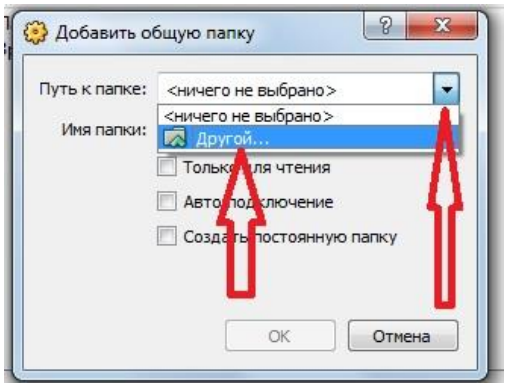
Открывается окно *Имя машины-Свойства*. В нем щелкаем кнопку *Общие папки* (внизу списка слева). Кнопка должна стать активной (окраситься в синий цвет). Затем щелкаем справа на иконке с изображением папки с зеленым крестиком.



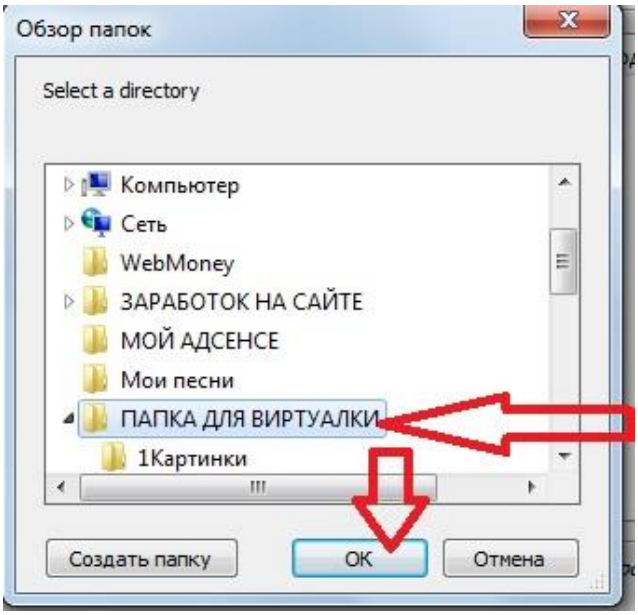
Обратите внимание на имеющийся список общих папок по умолчанию. Это **Папки машины** и **Временный папк**и. В зависимости от того, где будет расположена создаваемая в VirtualBox общая папка, зависит время ее жизни.

Далее откроется диалоговое окно ***Добавить общую папку.***

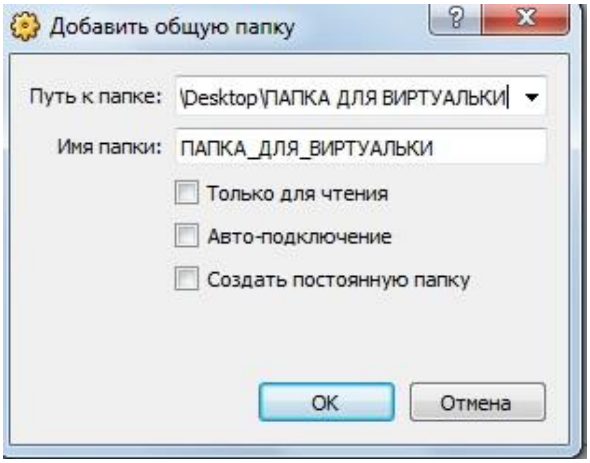
В окошке ***Путь к папке*** щелкаем на черном треугольнике, а затем на строчке ***Другой***.



Откроется диалоговое окно ***Обзор папок***, в котором мы ищем созданную ранее общую папку на основной ОС. Я ее назвал ранее **ПАПКА ДЛЯ ВИРТУАЛКИ**, активируем ее и щелкаем **ОК**.



В следующем диалоговом окне будет автоматически указан путь к выбранной папке и ее имя. Поставке галочку на Авто-подключении.



Здесь же мы можем проставить необходимые галочки в окошках.

Если поставить галочку в окошке *Только для чтения*, то здесь движение будет однонаправленное: только из основной ОС в гостевую. Из виртуальной машины уже нельзя будет сохранять или перемещать в эту папку файлы. Общая папка будет создана и расположена во Временных папках.

Если выбрать *Автоподключение*, то при запуске виртуальной машины, она будет пытаться подключиться к общей папке VirtualBox window. Но именно пытаться. Совсем не факт, что она подключится.

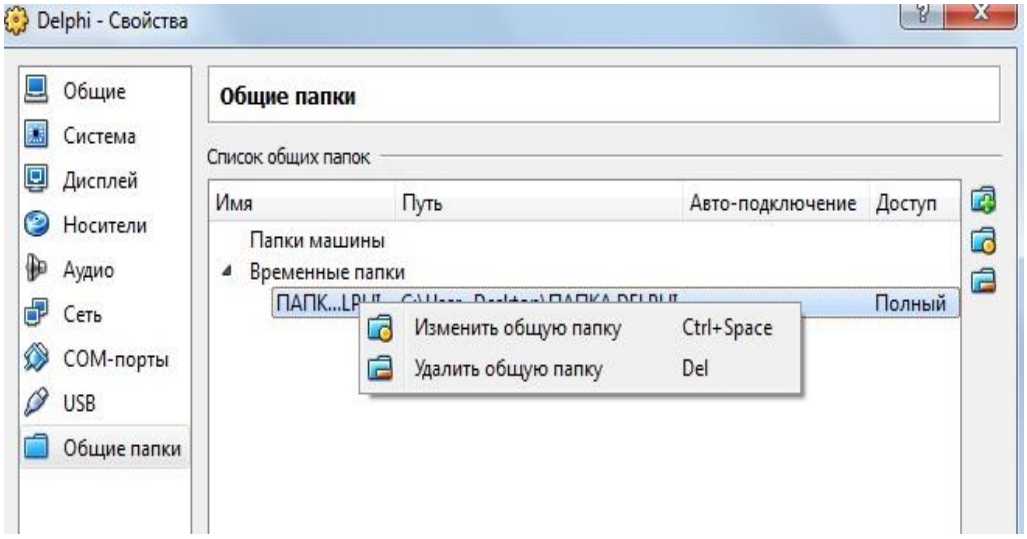
Если выбрать ***Создать постоянную папку***, то эта папка будет постоянной для данной виртуальной машины. Сохранена будет в разделе **Постоянные папки.**

Можно вообще не делать выбор и тогда данная папка будет сохранена в разделе *Временные папки* для данной виртуальной машины.

1.4 Настройка общей папки VirtualBox

Настройка и управление общей папкой в виртуальной машине `VirtualBox windows не представляет никаких сложностей.

Созданную папку можно всегда изменить или удалить, щелкнув на ее названии правой кнопкой. В появившемся контекстном меню достаточно выбрать соответственно ***Изменить общую папку*** или ***Удалить общую папку***.



Как только мы создадим общую папку, она сразу же появится в Проводнике в разделе ***Сетевое окружение – Вся сеть – shared folder.*** И мы можем через Проводник получить к ней доступ.

Но время жизни этой общей папки не постоянно, а только до закрытия виртуальной машины. При следующем включении виртуальной машины, мы доступ к созданной папке не получим. Она будет удалена. Нужно будет снова создавать общую папку и через проводник получать к ней доступ.

Дело в том, что эта общая папка была создана как временная и при закрытии машины она удаляется как из раздела *Общие папки – Временные папки*, так и из **Проводника.**

Кстати, таким образом можно получать доступ к любой папке на основной ОС, если, конечно это не запрещено правилами безопасности, а не только к специально созданной общей папке. Но это временный доступ, только на период работы виртуальной машины.

Даже если мы создадим временную общую папку virtualbox, а затем подключим ее, используя функцию *Подключить сетевой диск*, у нас появится новый сетевой диск в папке **Мой компьютер**, но при выключении виртуальной машины, а затем новом включении ссылка с этого сетевого диска перестанет работать.

1.4.1 Подключение и настройка постоянной общей папки VirtualBox Windows

Для того чтобы можно было постоянно пользоваться созданной общей папкой для виртуальной машины VirtualBox, необходимо ее определенным образом настроить.

Добавляем общую папку и ставим галку ***Создать постоянную папку*** (рис.4). Как только нажмем **ОК**, она появляется в списке *Постоянные папки*.

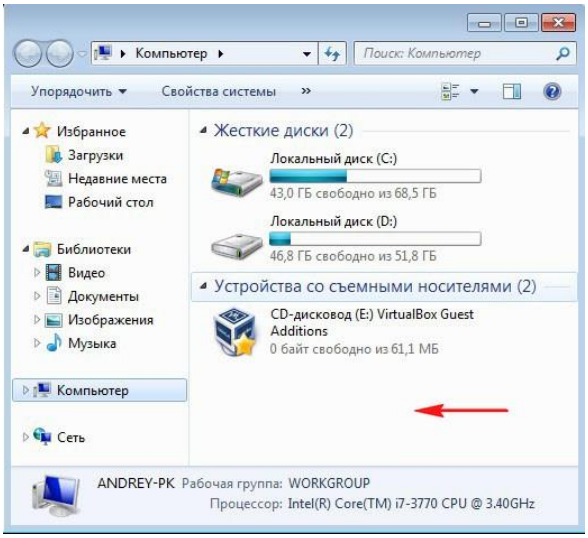
Теперь доступ к этой общей папке возможен через **Проводник - Сетевое окружени**е или **Главное меню – Сетевое окружение** и она будет существовать постоянно.

После этого можно пользоваться папкой. Она будет работать в обеих направлениях. То есть если вы перемещаете в нее из виртуалки документ, он тоже сохраняется.

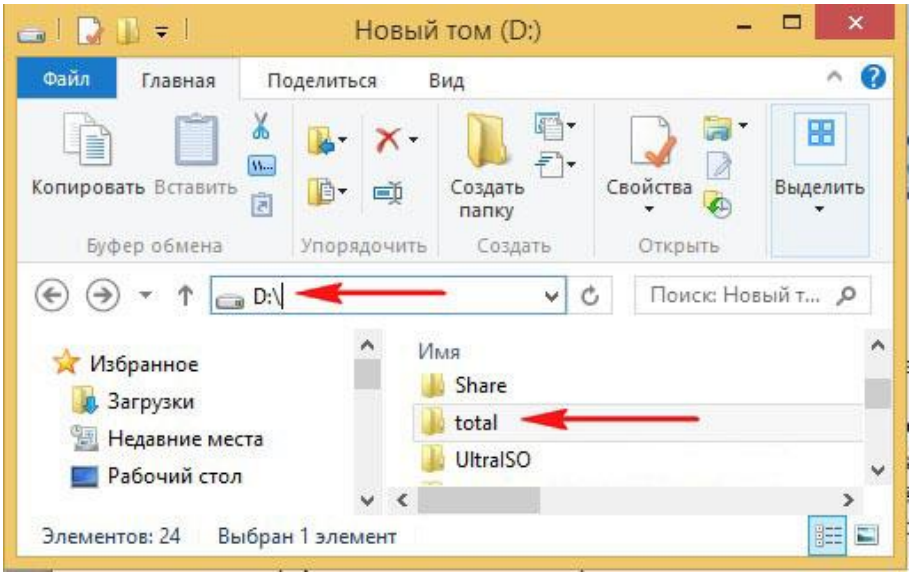
Для удобства пользования можно продолжить дополнительную настройку общей папки, воспользовавшись функцией **Подключить сетевой диск** и подключить эту папку. В папке **Мой компьютер** появится новый сетевой диск. Затем можно создать ярлык со ссылкой на него и прямо с рабочего стола открывать созданную папку.

1.4.2. Если общей папки нет?!

После установки на виртуальную машину операционной системы заходим в окно Компьютер и видим, что общей папки нет.

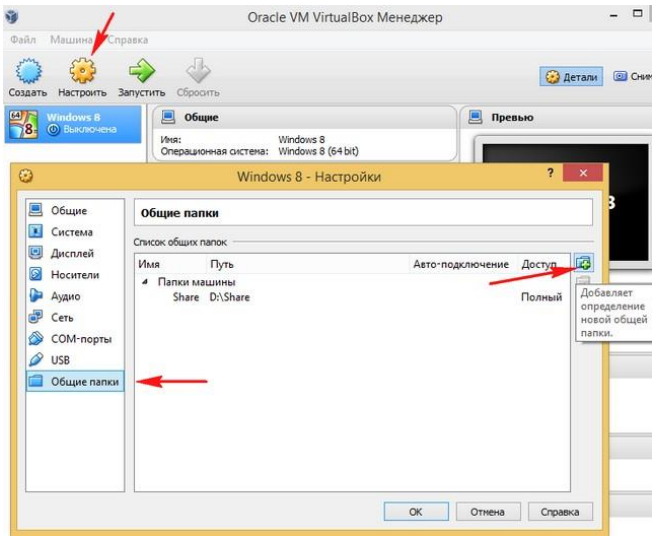


В первую очередь создаём общую папку, к примеру я создаю папку под название total на диске D: своей основной операционной системы в которую установлена виртуальная машина.

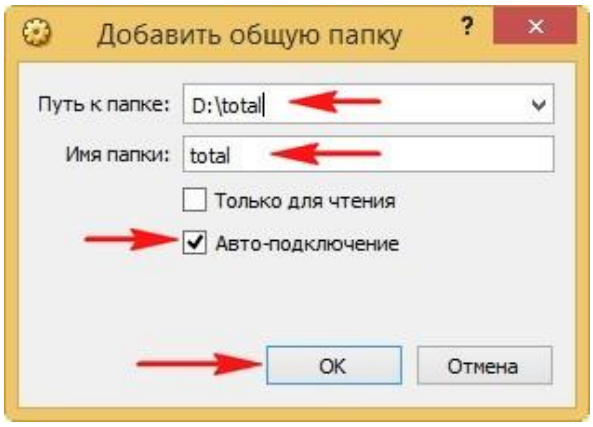


Любой скопированный в эту папку файл будет доступен для работы в установленной на виртуальную машину Windows.

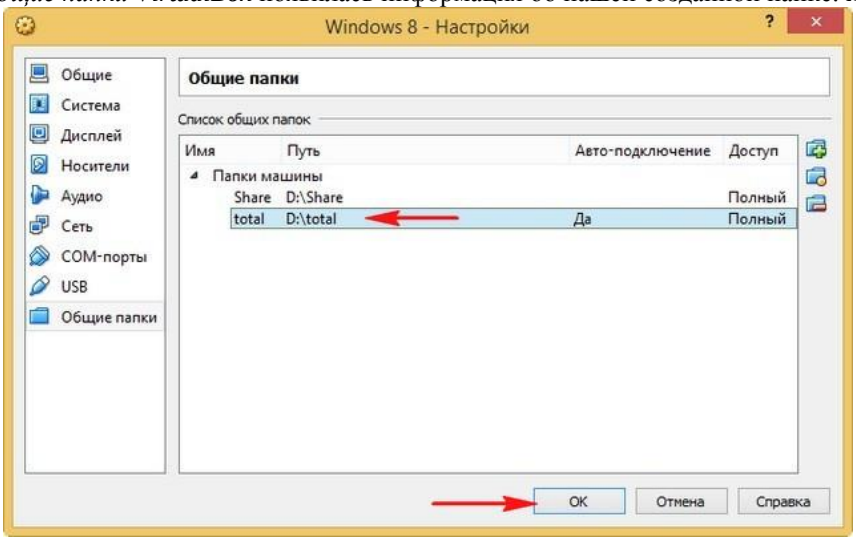
Теперь вносим информацию о созданной папке в настройки виртуальной машины. Запускаем виртуальную машину и жмём на кнопку Настроить, затем выбираем пункт Общие папки и жмём на плюсик.



В данном окне вводим:
Полный путь к созданной папке D:\total
Имя total
Можете поставить галочку па пункте Автоподключение.
Нажимаем ОК.

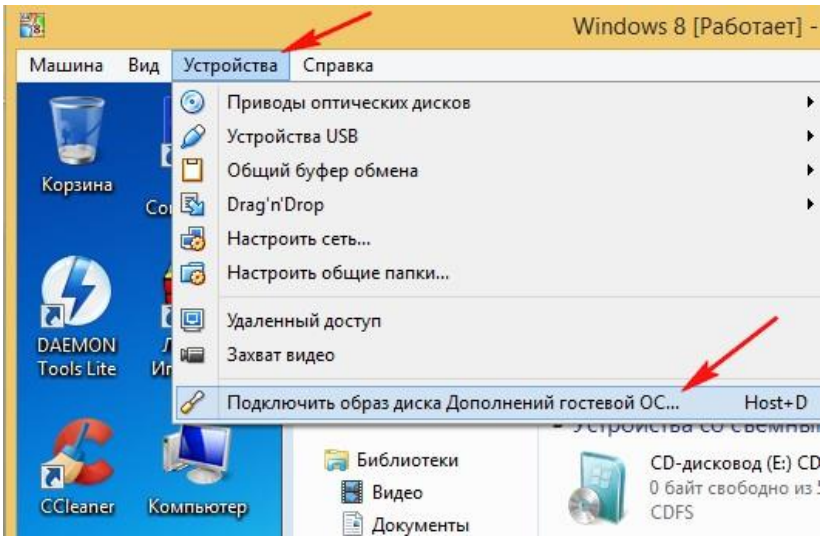


В разделе *Общие папки VirtualBox* появилась информация об нашей созданной папке. Жмём ОК.

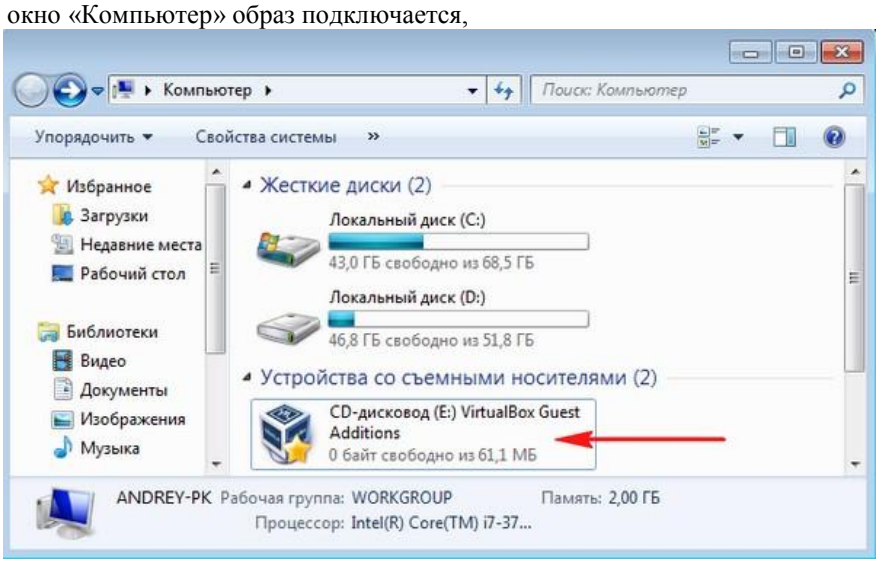


Запускаем установленную на виртуальную машину операционную систему или устанавливаем Windows на виртуальную машину, если она у вас ещё не установлена.

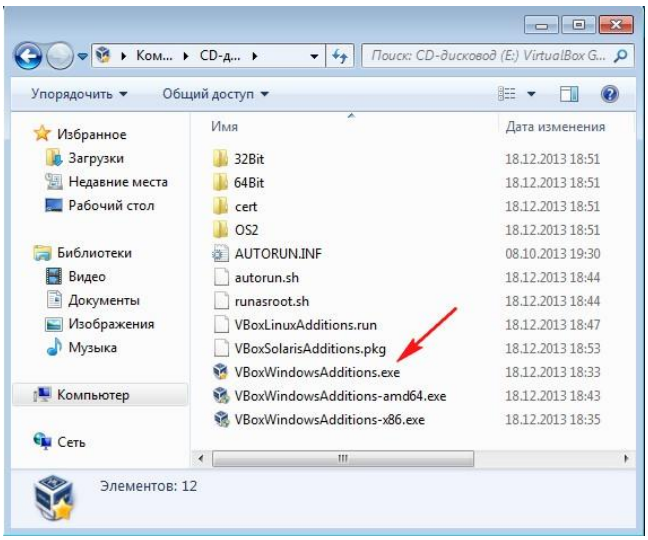
После установки операционной системы идём в Устройства, далее Подключить образ диска Дополнений гостевой ОС.



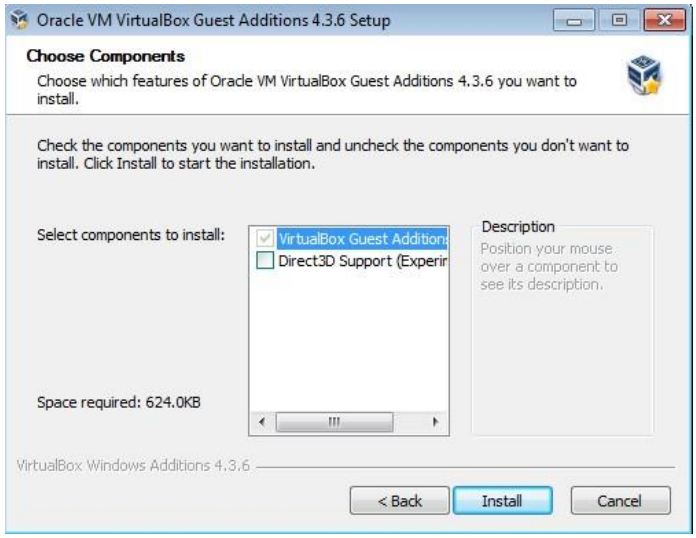
Открываем окно «Компьютер» образ подключается,



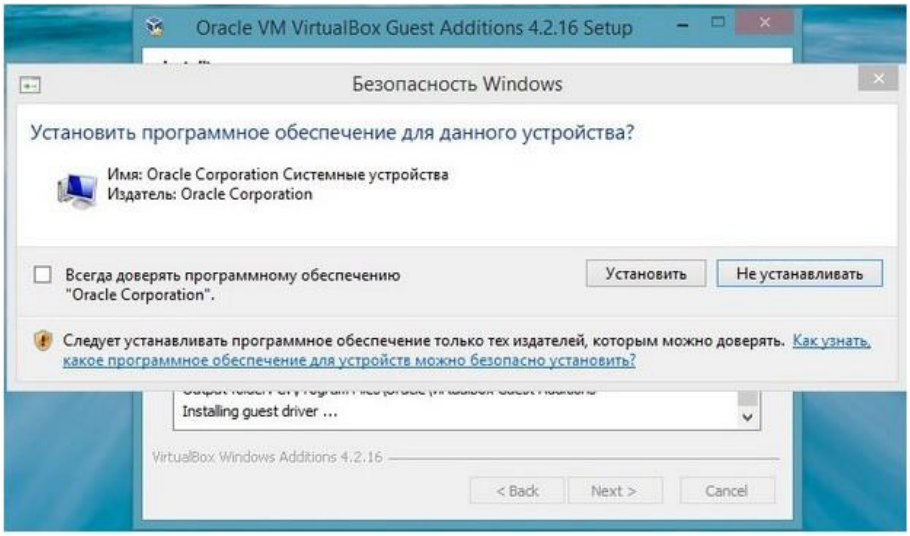
Заходим внутрь. Запускаем установку обновлений для гостевой ОС, щёлкаем двойным щелчком левой мыши на файле VBoxWindowsAdditions,



начнётся установка дополнений для гостевой ОС. Next.

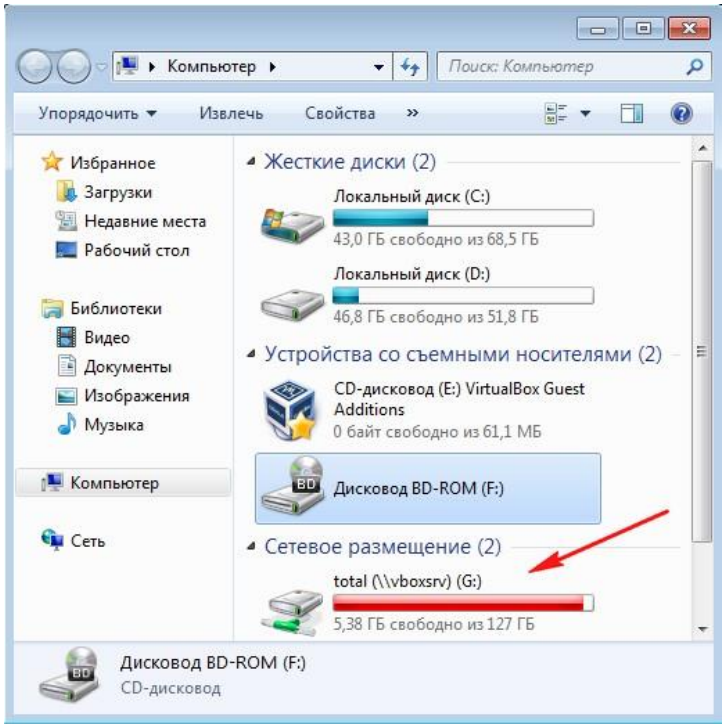


Установить

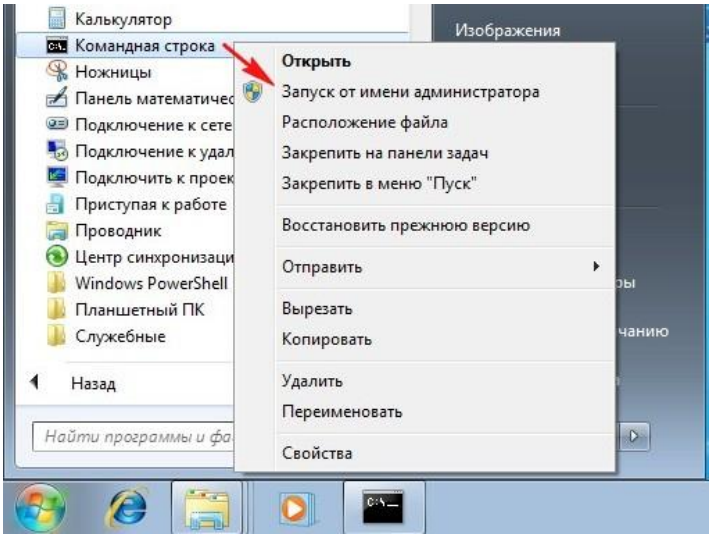


Нажимаем Финиш и Windows установленная на виртуальную машину перезагружается.

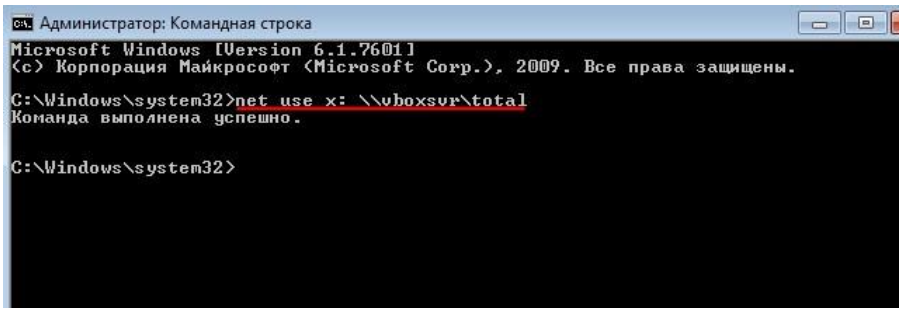
После перезагрузки у Вас должна появиться общая папка «total».



Если после этого общая папка у вас не появилась, тогда в операционной систему установленной на виртуальную машину запускаем командную строку от имени администратора.



В появившейся командной строке вводим команду net use x: \\vboxsvr\total (где total название общей папки) и жмём Enter на клавиатуре.

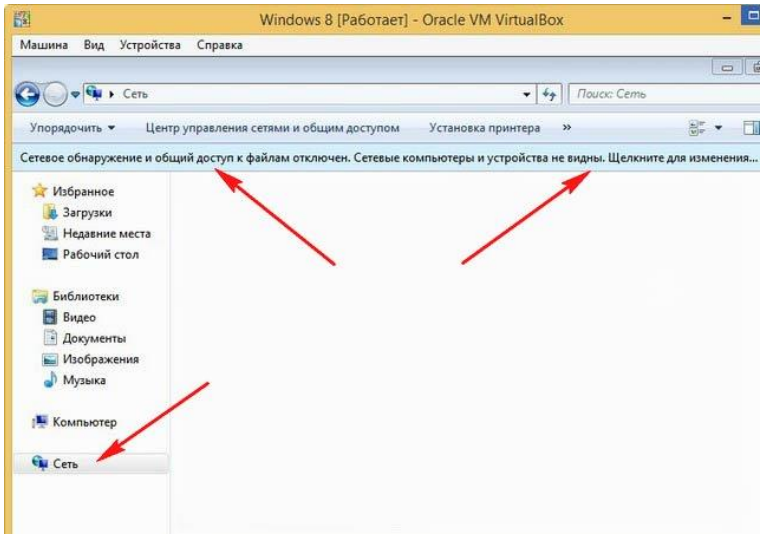


Общая папка VirtualBox должна появится в окне Компьютер. Если у кого-то здесь возникнет ошибка, значит Вы поленились установить дополнения для гостевой ОС.

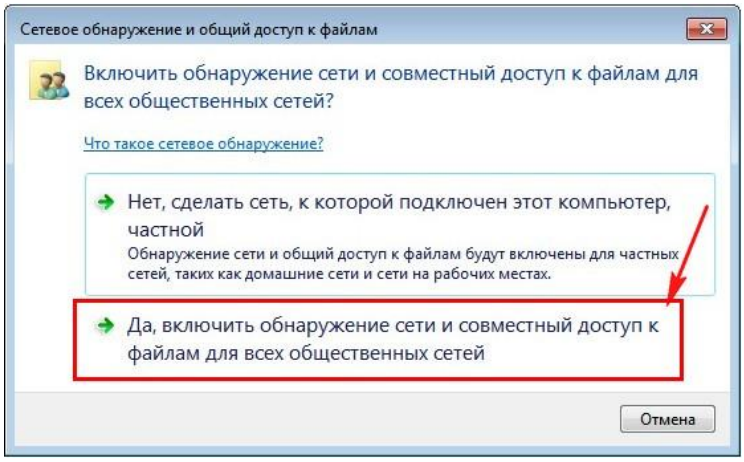
Если и после данных действий общая папка VirtualBox не появится в окне «Компьютер», значит щёлкаем мышью на значке Сеть.

В верхней части окна появится сообщение. «Сетевое обнаружение и общий доступ к файлам отключен. Сетевые устройства и компьютеры не видны. Щёлкните для изменения...»

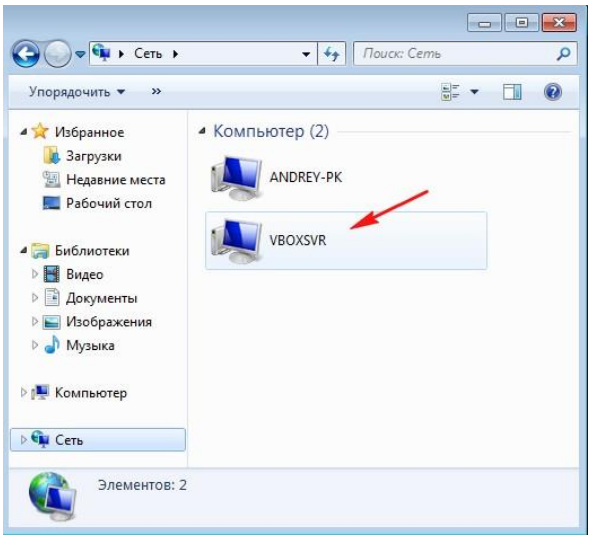
Щёлкаем на сообщении



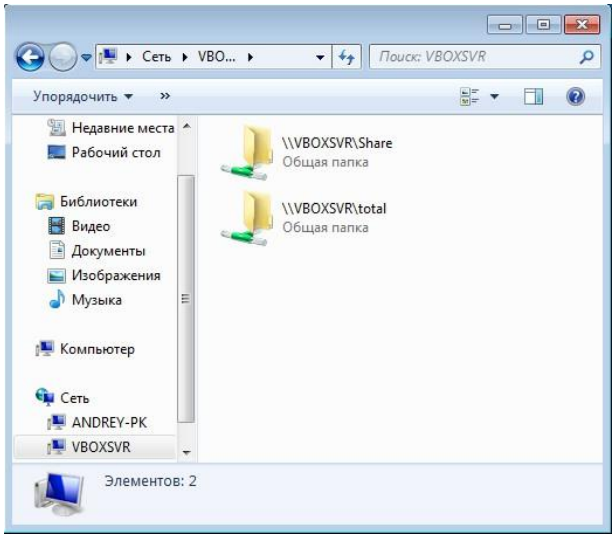
и в следующем открывшемся окне нажимаем на «Да включить обнаружение сети и совместный доступ к файлам для всех общественных сетей»



В данном окне появляется папка \\VBOXSVR, заходим в неё и видим все созданные нами общие папки VirtualBox.



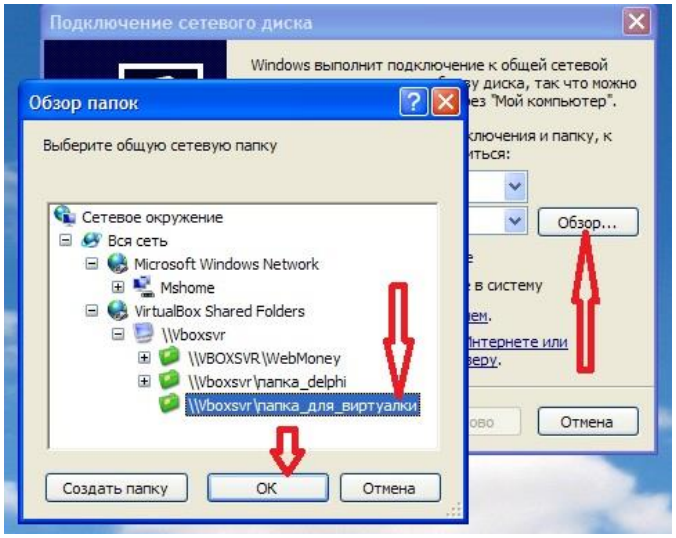
В данных общих папках находятся файлы доступные для установленной на виртуальную машину операционной системы и основной операционной системы.



1.4.3. Как настроить общую папку через подключение сетевого диска

Дальнейшая настройка общей папки VirtualBox заключается в подключении ее через функцию **Подключить сетевой диск.**

Для этого в виртуальной машине VirtualBox windows щелкаем правой кнопкой **Мой компьютер**. В контекстном меню выбираем ***Подключить сетевой диск***. Открывается диалоговое окно ***Подключение сетевого диска***.



Через кнопку ***Обзор*** входим в ***Обзор папок*** и выбираем ту папку, которую мы хотим подключить. В данном случае это созданная ранее общая *Папка для виртуалки*. Щелкаем **ОК** и у нас будет подключен новый сетевой диск.

Дальнейшая настройка общей папки заключается в том, что заходим в папку Мой компьютер, находим там созданный сетевой диск.

Правой кнопкой щелкаем на этом диске и далее **Создать ярлы**к. Этот ярлык будет создан на рабочем столе.

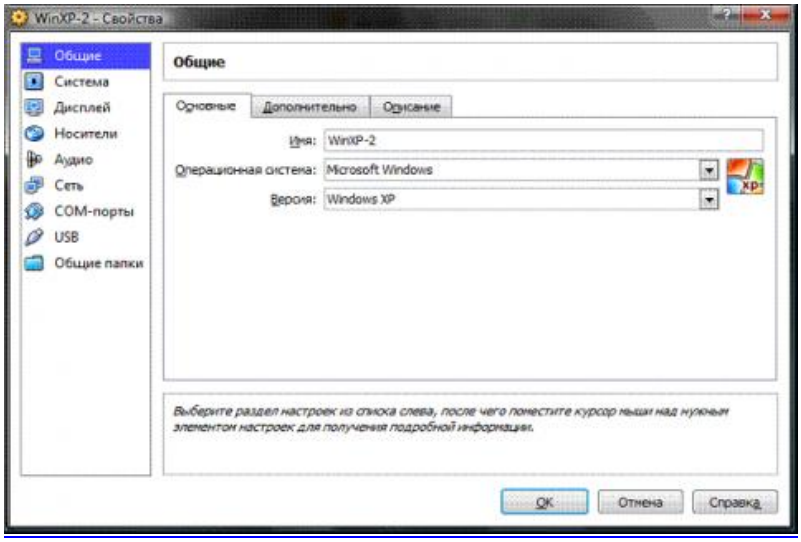
На этом собственно **подключение и настройка общей папки VirtualBo**х можно считать законченной. Через созданный ярлык на рабочем столе мы можем пользоваться этой папкой как обычно.

Нужно иметь ввиду, что расположена эта общая папка будет на основной ОС.

С одной стороны, это хорошо. Она не занимает место на виртуальном диске. Но с другой стороны ее желательно не размещать на системном разделе жесткого диска. Это на случай переустановки операционной системы. Тогда вся информация в этой папке и сама папка будут сохранены.

1.5 Настройка аппаратной части виртуальной машины

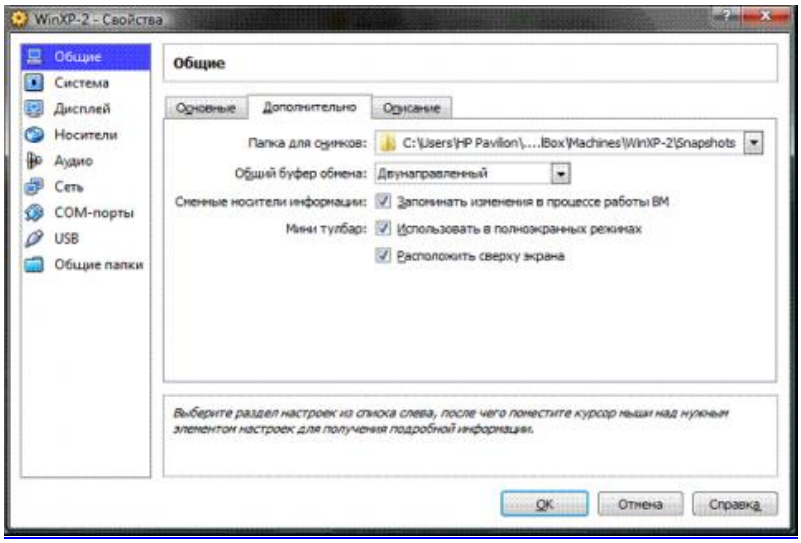
В колонке слева выберем нашу WinXP-2 и откроем её свойства, где колонка с левой стороны напоминает диспетчер устройств. На первой вкладке раздела «Общие» мы видим основные параметры нашей виртуальной машины:



Перейдем на вкладку дополнительно и посмотрим, какие настройки системы мы можем произвести:

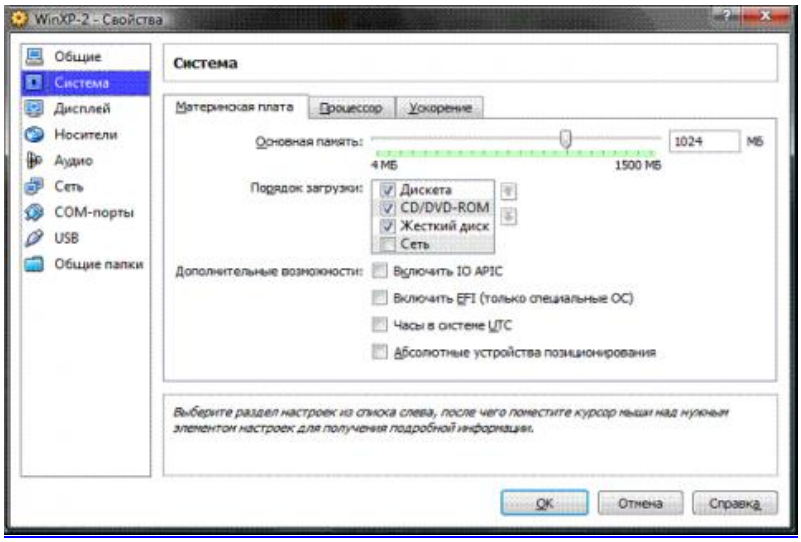
- «Папка для снимков». Если Вы разместили Ваш жесткий в собственном расположении, то лучше и эту папку перенести туда же, т.к. снимки имеют большой вес и, опять-таки, не стоит перегружать Ваш системный диск. Моя рекомендация – создавать снимки перед каждым значительным изменением, которые Вы хотите произвести в виртуальной системе, причем даже на одну виртуальную машину Вы можете создать несколько снимков, содержащих отличные друг от друга настройки и установленные приложения;
- «Общий буфер обмена» – определение того, как будет работать буфер обмена между Вашей host-системой и виртуальной машиной. Вариантов работы буфера предоставлено четыре – «выключено», «только из гостевой ОС в основную», «только из основной ОС в гостевую», «двунаправленный». Мы выберем последний вариант, т.к. это обеспечит нам максимальное удобство в работе;

- «Сменные носители информации запоминать изменения в процессе работы», тут мы ставим флажок в знак согласия, т.к. данная опция позволит системе запомнить состояние CD\DVD-приводов;
- «Мини тулбар» – это небольшая консоль, содержащая элементы управления виртуальной машиной. Её лучше применять только в полноэкранном режиме, т.к. она полностью дублируется главным меню рабочего окна виртуальной машины. Располагать её действительно лучше сверху просто потому, что можно случайно нажать на какой-нибудь элемент управления, пытаясь, например, развернуть окно из панели задач виртуальной машины.

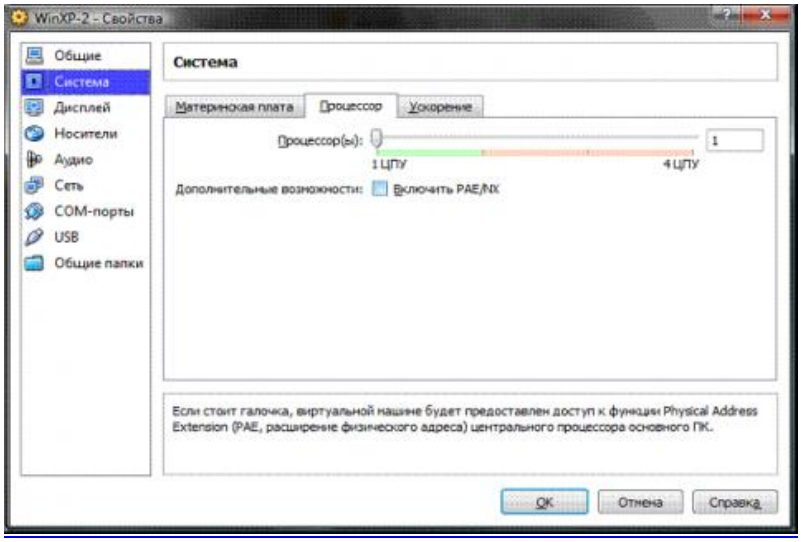


Перейдем к разделу система и на первой вкладке материнская плата произведем следующие настройки:

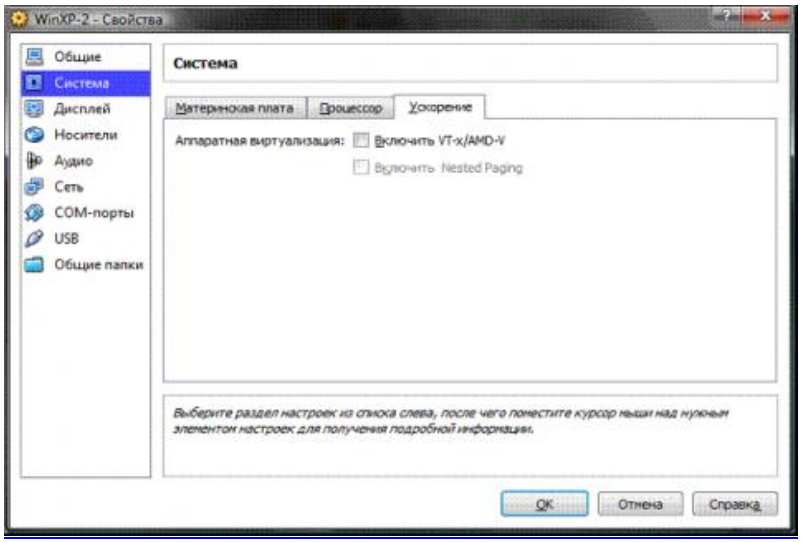
- если нужно, откорректируем размер оперативной памяти Вашей виртуальной машины, хотя окончательно убедится в правильности выбранного объема Вы сможете только после запуска виртуальной машины. Выбирать размер Вы можете, исходя из объема доступной физической памяти, установленной на Вашем ПК. Например, при наличии 4ГБ ОЗУ оптимальным будет выделение 1ГБ, т.е. одной четвертой части, что позволит Вашей виртуальной машине работать без малейших зависаний;
- откорректируем порядок загрузки - дисковод гибких дисков («дискета) можно вообще отключить, а первым обязательно поставьте CD/DVD-ROM, чтобы обеспечить возможность установки ОС с загрузочного диска. При этом в роли загрузочного диска может выступать как и компакт-диск, так и образ ISO;
- все остальные настройки описаны в динамической справке снизу, и их применение зависит от аппаратной части вашего реального ПК, причем если Вы выставите настройки неприменимые к Вашему ПК система виртуальной машины просто не запуститься;



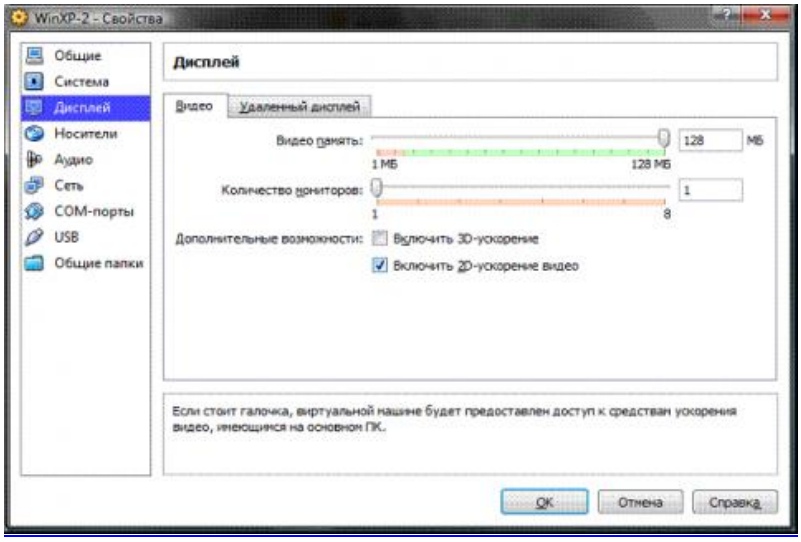
Перейдем к вкладке «Процессор», тут Вы можете выбрать количество процессоров, установленных на Вашу виртуальную материнскую плату. Обратите внимание, что это опция будет доступна только при условии поддержки аппаратной виртуализации AMD-V или VT-x, а также включенной опции OI APIC на предыдущей вкладке.



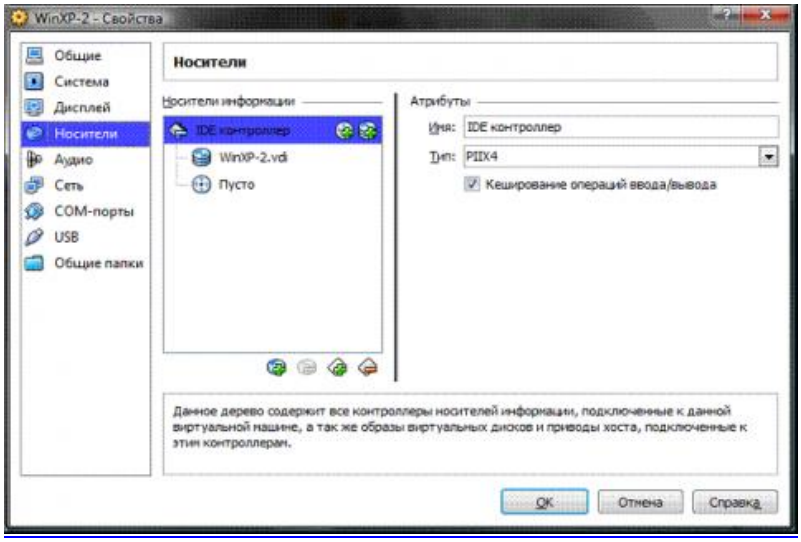
Здесь я снова обращу Ваше внимание на настройки аппаратной визуализации AMD-V или VT-x. Перед включением этих настроек, нужно выяснить, поддерживает ли эти возможности Ваш процессор и включены ли они по умолчанию в BIOS (нередко они отключены).



Перейдем к разделу «Дисплей». В данном разделе на вкладке «Видео» Вы можете установить размер памяти виртуальной видео карты, а также включить 2D и 3D ускорение, причем включение 2D ускорения желательно, а 3D необязательно. На вкладке «Удаленный дисплей» Вы можете включить опцию, при которой Ваша виртуальная машина будет работать как сервер удаленного рабочего стола (RDP).



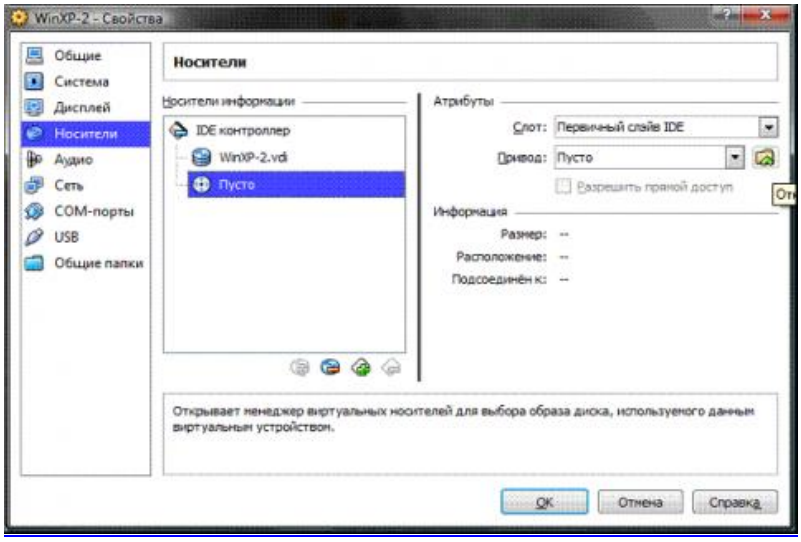
Переходим к разделу носители. Тут Вы можете увидеть созданной ранее виртуальный жесткий диск и позицию с надписью пусто. Выделяем эту позицию и осуществляем настройку.



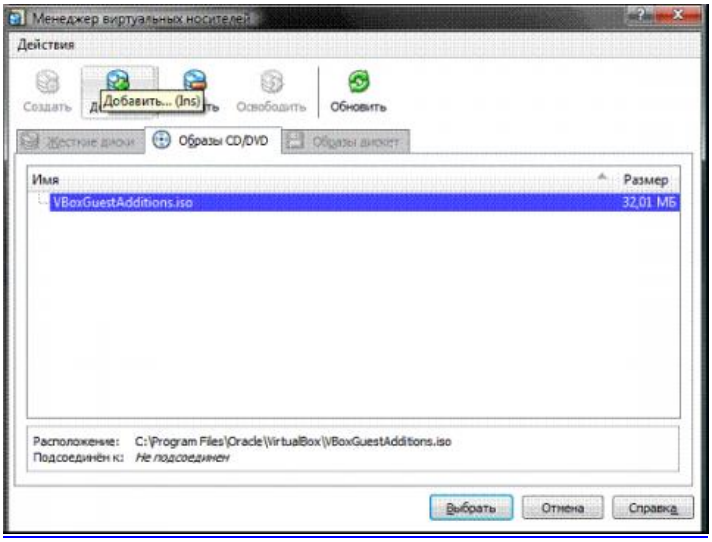
Для настройки виртуального привода компакт-дисков можно пойти двумя путями:

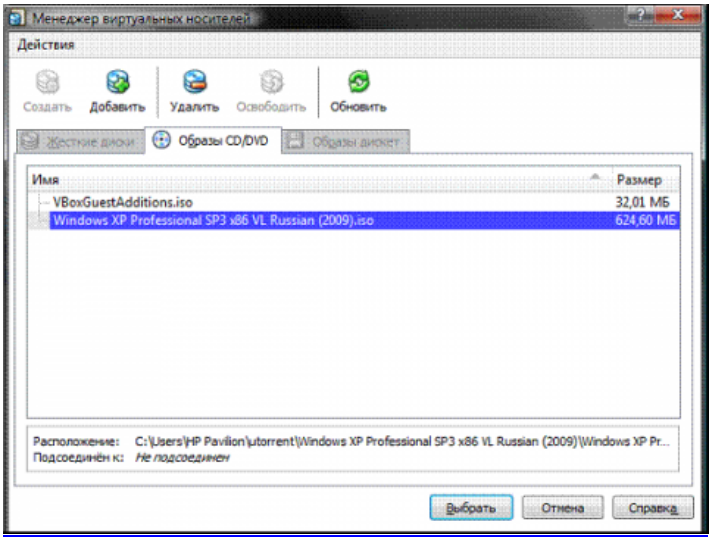
- первый вариант - в раскрывающемся меню «Привод» выбираем Ваш реальный или виртуальный CD/DVD-ROM (существующие в реальной системе) и загружаем в него физический диск с дистрибутивом Windows XP или ISO-образ, если это эмулятор;
- второй вариант - щелкаем значок так, как показано на рисунке ниже и в отрывшемся окне добавляем ISO-образ загрузочного диска Windows XP, этим путем мы и пойдем.

Примечание. В данном пункте Вы уже не можете выбрать дистрибутив другой операционной системы, т.к. версия ОС уже была определена в самом начале процесса настройки виртуальной машины.

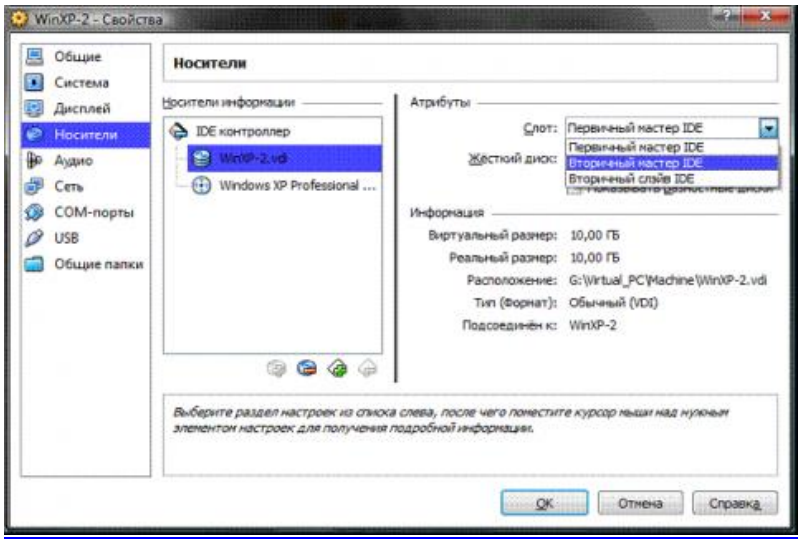


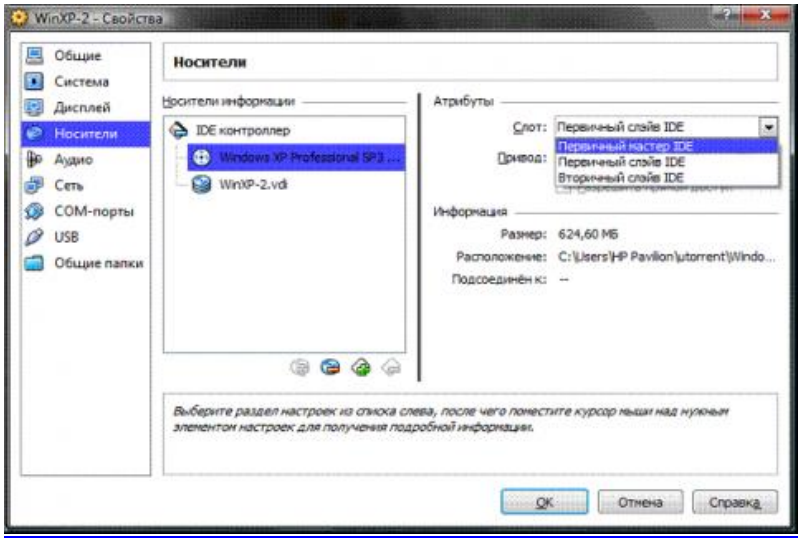
На рисунке ниже представлена процедура добавления ISO-образов в менеджер виртуальных носителей. В него Вы можете внести любое количество образов различного назначения, например, игры, дистрибутивы приложений, базы данных и пр., которые Вы сможете потом быстро переключать через главное меню окна виртуализации VirtualBox.



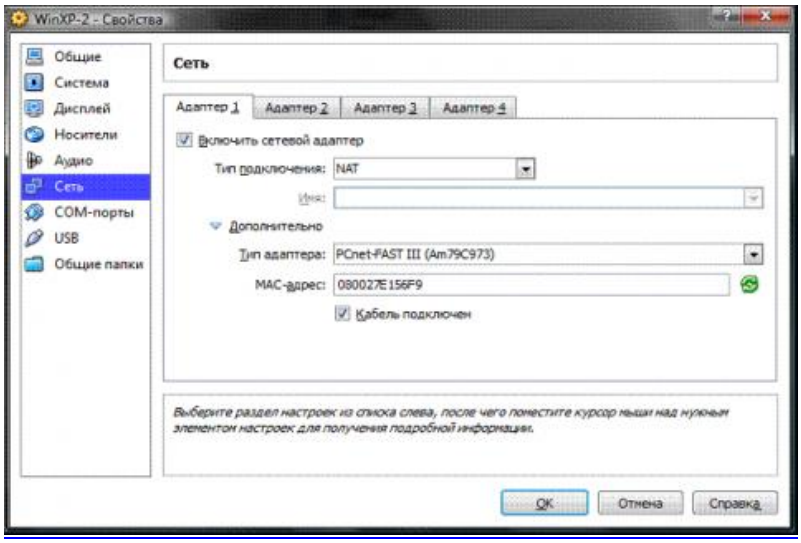


Далее Вы можете настроить слоты подключения накопителей, для упрощения описания привожу скриншоты, по которым Вы можете произвести действия по настройке. По привычке, я устанавливаю привод компакт-дисков как «Первичный мастер IDE», жесткий диск, содержащий загрузочный раздел, как «Вторичный мастер IDE», а дополнительный виртуальный жесткий диск «Первичный слейв IDE».



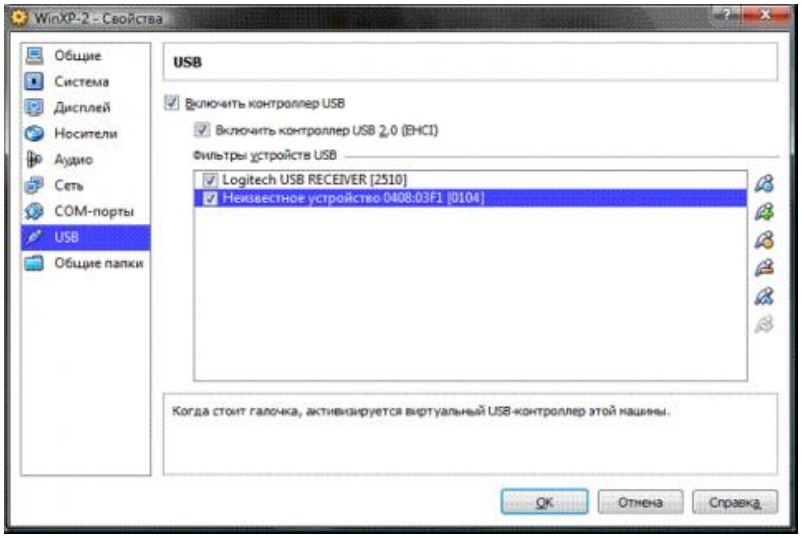


Настройка сети и сетевого взаимодействия не рассматривается в рамках данной статьи, поэтому замечу лишь то, что сетевой адаптер типа NAT включен по умолчанию, а этого уже достаточно для предоставления Вашей виртуальной машине доступа в Интернет. Тип выбираемого адаптера должен быть «Pcnet-Fast III (Am79C973)», т.к. только для этого адаптера присутствуют драйверы в ОС Windows XP.

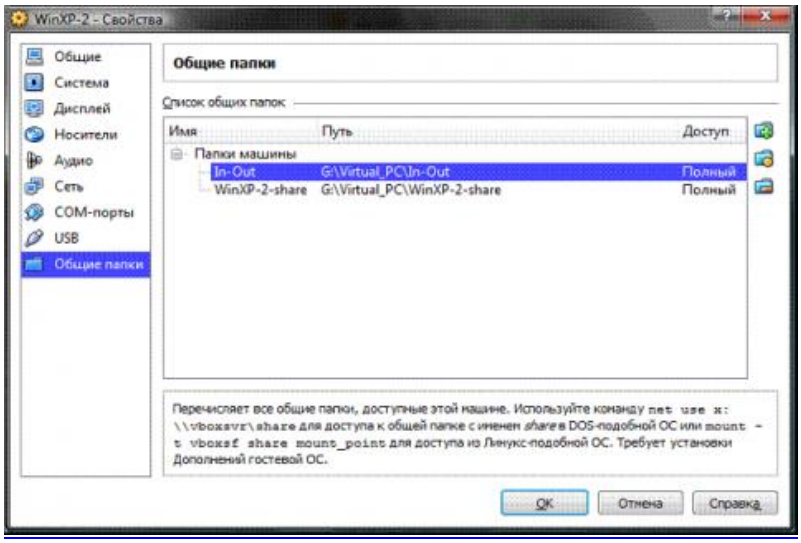


Раздел COM я подробно не описываю, т.к. подключать к портам данного типа уже нечего. В случае если Вам все же потребуется подключить устройство с интерфейсом RS-232C, то наиболее удобным будет включить COM-port виртуальной машины в режиме «хост-устройство», а в качестве «пути к порту» использовать имя реально порта Вашего ПК, которое Вы можете посмотреть в диспетчере устройств.

Переходим к разделу USB, здесь ставим оба доступных флажка, а затем, используя кнопку с изображением «вилки» USB и «плюса», добавляем все доступные контроллеры.



Переходим к разделу «Общие папки» и выбираем папки, которые нужно сделать доступными для виртуальной машины.



Примечание. Обратите внимание на динамическую справку снизу – именно таким образом, через окно командной строки, Вы сможете подключить общие папки к Вашей виртуальной машине.

Задание

1. Установить Виртуальную машину;
2. Изучить интерфейс и возможности VirtualBox.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3:

1. Что такое и для чего используется виртуальная машина?
2. Как создать общую папку? Для чего она используется?
3. Расскажите о возможностях ВМ?

Лабораторная работа №4
«Арифметические операции в двоичной системе счисления»

Цель работы:

1. Научиться выполнять арифметические операции над числами в двоичной системе счисления.

1 Основные понятия и термины

В двоичной системе счисления арифметические операции выполняются по тем же правилам, что и в десятичной системе счисления, т.к. они обе являются позиционными (наряду с восьмеричной, шестнадцатеричной и др.).

1.1 Сложение

Сложение одноразрядных двоичных чисел выполняется по следующим правилам:

```
0 + 0 = 0
1 + 0 = 1
0 + 1 = 1
1 + 1 = 10
```

В последнем случае, при сложении двух единиц, происходит переполнение младшего разряда, и единица переносится в старший разряд. Переполнение возникает в случае, если сумма равна основанию системы счисления (в данном случае это число 2) или больше его (для двоичной системы счисления это не актуально).

Сложим для примера два любых двоичных числа:

```
  1101
+  101
 ------
 10010
```

1.2 Вычитание

Вычитание одноразрядных двоичных чисел выполняется по следующим правилам:

```
0 - 0 = 0
1 - 0 = 1
0 - 1 = (заем из старшего разряда) 1
1 - 1 = 0
```

Пример:

```
  1110
-  101
  ----
  1001
```

1.3 Умножение

Умножение одноразрядных двоичных чисел выполняется по следующим правилам:

```
0 * 0 = 0
1 * 0 = 0
0 * 1 = 0
1 * 1 = 1
```

Пример:

```
  1110
*10
------
+ 0000
```

```
 1110
------
 11100
```

1.4 Деление

Деление выполняется так же как в десятичной системе счисления:

```
1110 | 10
  |----
10| 111
----
 11
 10
----
  10
  10
----
0
```

Сложение	Вычитание	Умножение
$0+0=0$	$0-0=0$	$0*0=0$
$0+1=1$	$1-0=0$	$1*0=0$
$1+0=1$	$1-1=0$	$0*1=0$
$1+1=10$	$10-1=1$	$1*1=1$

Задание

Выполните арифметические действия:

1	1010000100+1101110	29	110110101+1010001111
2	1111010111-1011100011	30	1110110010-1110011100
3	1001110*101000	31	100101*11000
4	11110101000/10100	32	10000000010/110110
5	1010001011+1111001010	33	1110011101+1100001101
6	1011100110-1010111011	34	1101100010-11111001
7	1001010*11100	35	11010*1010001
8	1010000000/100000	36	111101100000/1010010
9	1011111111+111001001	37	11110100+1011101100
10	1011001000-1000010001	38	1101100110-100100101
11	1001000*1010110	39	111011*1101
12	110101001/10001	40	1000011000100/111010
13	1010000100+1101110	41	110000101+1010101111
14	1111010111-1011100011	42	1010110110-1001001100
15	1001110*101000	43	100010*10100
16	11110101000/10100	44	1110111011100/1011011
17	1010001011+1111001010	45	11111011+111010010
18	1011100110-1010111011	46	1101000011-1100100000
19	1001010*11100	47	110000*1000101
20	1010000000/100000	48	11100001000/11001
21	111111110+1011010001	49	1011110001+1111000011
22	10111101-10000001	50	111001010-110100111
23	110100*1000001	51	1011001*1010010
24	101110001110/110011	52	11000111011/11101
25	11100000+1010011110	53	110011111+1010111001
26	1001001011-111111110	54	1100011110-110010101
27	101000*1000011	55	100110*1100001
28	100000101010/10110	56	1110000110000/1010010

57	1010000101+1000111011	73	1110011100+1100000000
58	1101110000-1001000001	74	1101110100-100011001
59	111101*11001	75	1001111*11110
60	1000001110100/110100	76	1000101001100/1010010
61	1000010000+1001100110	77	1110010010+101000001
62	1001101100-111110010	78	1011000111-111100100
63	1011*1000100	79	11001*100101
64	10110000010/1111	80	1000110001000/1000010
65	111111111+110001100	81	1111010110+10111111
66	1100000011-1001110111	82	101000110-100111110
67	1000110*1000111	83	1001001*100110
68	101101011000/101100	84	11111001011/100011
69	11001010+1101011011	85	110100010+1111001011
70	1011101111-1011011100	86	1000111111-100000010
71	111011*110100	87	110110*111000
72	11000111110/100010	88	1100000000/100000

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4:

1. Как выполнить арифметическое сложение двоичных чисел?
2. Как выполнить арифметическое вычитание двоичных чисел?
3. Как выполнить арифметическое умножение двоичных чисел?
4. Как выполнить арифметическое деление двоичных чисел?

Лабораторная работа №5
«Изучение основных команд ОС UNIX»

Цель работы:

1. Изучить принципы построения интерфейса и основные команды ОС UNIX;
2. Получить навыки работы в ОС UNIX, с использования интерфейса командной строки.

1 Основные понятия и термины

1.1 Интерфейс пользователя. Основные сведения

Одним из важных элементов операционной системы общего назначения, является интерфейс пользователя. Интерфейс пользователя вычислительной системы представляет собой совокупность средств, методов, и правил, при помощи которых пользователь взаимодействует с вычислительной системой. Интерфейс пользователя реализуется как аппаратными средствами, так и программным обеспечением. При выполнении цикла лабораторных работ по курсу «Операционные системы» мы будем взаимодействовать с ОС LINUX посредством интерфейса командной строки (Command Line Interface - CLI).

Этот интерфейс требует минимальных технических средств, хотя может имитироваться с использованием современной аппаратуры. Простейшая схема аппаратных средств, необходимых для CLI показана на рисунке 1.

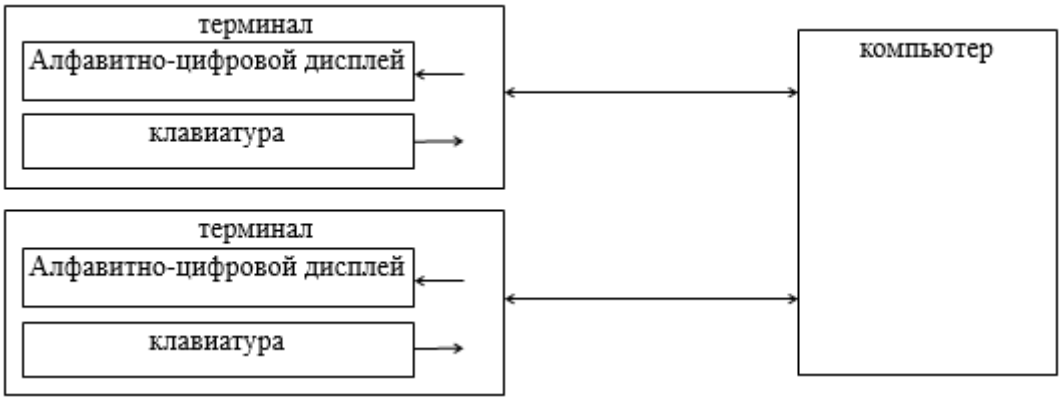


Рисунок 1 — Аппаратные средства CLI

Пользователь набирает на клавиатуре текстовые команды, которые последовательно по одному символу поступают в вычислительную систему по линии связи. Вычислительная система последовательно по одному символу выводит через линию связи текстовые сообщения, которые отображаются на алфавитно цифровом дисплее. Как видно на схеме, к компьютеру могут быть одновременно подключены несколько терминалов, за каждым из которых работает пользователь. На стороне компьютера, с терминалами могут взаимодействовать несколько параллельно выполняющихся программ, в том числе программы типа 'командный интерпретатор' или оболочка (shell), считающаяся частью ОС.

В вычислительных системах на базе современных персональных компьютеров используемых в цикле лабораторных работ, аппаратные средства представлены, как правило, единственным графическим дисплеем и клавиатурой. При этом, наличие нескольких терминалов, между которыми может переключаться пользователь, имитируется.

1.2 Сеанс работы пользователя в ОС UNIX

Работа пользователя в ОС UNIX с интерфейсом CLI начинается после завершения начальной загрузки при включении питания. В процессе загрузки выводятся многочисленные текстовые сообщения, информационного и контрольного характера.

В ОС LINUX с одним графическим дисплеем и клавиатурой, пользователь может войти в систему одновременно с нескольких терминалов, используя учётные данные (login, password) одного или разных пользователей. Переключаться между 6-тью доступных виртуальными терминалами можно одновременным нажатием клавиш Alt и F1, Alt и F2, … , Alt и F6. Работа в каждом терминале начинается с регистрации в системе, завершить работу в сеансе пользователя можно (переключившись на нужный терминал) выполнив команду exit.

Завершить работу компьютера может только пользователь обладающий специальными правами — 'правами администратора'. Для выключения компьютера будем пользоваться командой sudo halt – выполнить команду halt – стоп системы - от имени администратора.

После входа в систему, пользователь может вводить команды. Эти команды будут исполняться операционной системой. Результаты исполнения команд будут отображаться на дисплее.

1.3 Команды

Будем использовать текстовые команды для:

−манипулирования данными, хранящимися в долговременной памяти (в виде файлов на диске) с помощью команд ОС;

−запуска прикладных и системных программ, в том числе средств разработки программного обеспечения;

−выполнения других действий.

Исполнение команд происходит по нажатию клавиши Enter. Команда ОС UNIx представляет собой последовательность символов — латинских букв, цифр или некоторых знаков как правило представляющих слово или аббревиатуру. Команды могут иметь один или несколько иметь аргументов (параметров), разделяемых символом-разделителем. Как правило в качестве разделителя используется пробел. Аргументы представляют данные, необходимые для выполнения команд. Одна и та-же команда может выполняться с разным набором параметров. Если некоторые параметры не указываются, используются их значения «по умолчанию». Для ограничения числа команд при увеличении гибкости их применения, большинство команд могут снабжаться необязательными аргументами - «ключами» или «опциями». Ключ начинается с символа – или --, за которым следует символ латинской буквы, цифры или слово. Несколько ключей могут применяться совместно. Возможность применения конкретного набора ключей рассматривается в документации на каждую команду. В качестве параметра приведены разные формы команды вывода содержимого каталога на дисплей.

ls - выводит содержимое текущего каталога

ls /home/student - выводит содержимое каталога /home/student

ls -l - выводит содерж. текущего каталога (подробный формат)

ls -s /etc - выводит содержимое каталога /etc и размер каждого элемента каталога в блоках

1.3.1 Справочная подсистема

CLI интерфейс требует для работы запоминания большого количества информации. В связи с этим используются разные интерактивные справочные системы. Одна из популярных систем носит название manual pages - страницы руководства или man. Система позволяет иметь в системе man – страницу по каждому объекту, например по команде. Просмотреть справочную информацию можно по команде.

man <имя объекта>

например man ls выдаёт информацию о команде ls, а man man – информации по команде man. Команда man может выдать информацию только о тех объектах, для которых ранее эта информация была внесена в справочную систему, например при установке системы или инсталляции соответствующего пакета.

1.3.2 Команды для работы с файловой системой

Информация в долговременной памяти, которая как правило является магнитным диском, организована в виде иерархической структуры файлов и подкаталогов, называемой файловой системой (ФС). Объекты ФС — файлы и каталоги — определяются именем. Могут использоваться три формы записи имени:

1. Полная форма, в которой перечислены все подкаталоги которые нужно пройти для доступа к файлу (подкаталогу) от корневого каталога (каталога самого верхнего уровня),разделённые символом-разделителем (/):
2. Краткая форма — это имя файла (подкаталога) в родительском каталоге

3. Относительная форма , в которой перечислены все подкаталоги которые нужно пройти для доступа к файлу (подкаталогу) относительно текущего каталога:

Если полное имя файла /home/student/file1.txt, краткое имя file1.txt. имя файла относительно текущего каталога /bin будет ../home/student/file1.txt, а относительно текущего каталога /home - student/file1.txt.

Имя .. означает имя родительского каталога, а имя . -имя текущего каталога.

Для смены текущего каталога используется команда

cd <имя подкаталога>

Для вывода на дисплей текущего каталога используется команда

pwd

Для вывода на дисплей содержимого каталога

ls <имя подкаталога>

Для создания пустого файла используется команда

touch <имя файла>

Для удаления файла используется команда

rm <имя файла(непустого подкаталога -R)>

Для создания каталога используется команда

mkdir <имя подкаталога>

Для удаления каталога (пустого) используется команда

rmdir <имя подкаталога>

Для перемещения и переименования файла или подкаталога используется команда

mv <исходное имя файла (подкаталога)> <новое имя (подкаталога)>

Для копирования файла или подкаталога используется команда

cp <исходное имя файла (подкаталога)> <новое имя (подкаталога)>

1.3.3 Запуск программ

Для запуска любой программы нужно набрать её полное или относительное имя. Краткое имя можно использовать для программ в каталогах, перечисленным в переменной окружения PATH, например для стандартных команд, большинство из которых представляют собой отдельные программы — утилиты расположенные в каталоге /bin или /sbin. Например вместо /bin/ls набирают просто ls. Для остальных программ, даже если они находятся в текущем каталоге, нужно набрать ./имя_программы.

Программы можно запускать в так называемом фоновом режиме. При этом можно вводить очередную команду ОС, не дожидаясь, пока запущенная программа завершится. Это полезно для программ, которые работают длительное или неопределённое время, например для программ проверки диска, или программы отслеживающей получение электронной почты. Для запуска в фоновом режиме достаточно после имени программы добавить символ &.

ls / -R& вывести содержимое всех подкаталогов рекурсивно

Выполнение программы запущенной не в фоновом режиме можно прервать, нажав вместе клавиши Ctrl+C. Программы запущенные в фоновом режиме можно завершить в два этапа:

1 Определить идентификатор процесса (PID), который соответствует запущенной программе, которую нужно завершить - с помощью программы ps;

2 Завершить программу командой kill -9 <PID>, где PID определён в п.1.

1.3.4 Команды применяемые при разработке программ

Минимальный набор команд, который будет применяться при разработке программ включает следующие команды:

Запустить текстовый редактор nano для набора текста новой программы или редактирования существующего файла программы:

nano имя_файла.c

Запустить программу компиляции

gcc имя_программы.c -o имя_программы

Запустить откомпилированную программу в режиме отладки («под отладчиком»)

gdb имя_программы

Эти и другие команды и соответствующие средства разработки подробнее описаны в [1].

1.3.5 Другие часто применяемые команды

Вывести на дисплей сообщение или переменную окружения

echo <сообщение>, или echo <$имя_переменной_окружения>

Очистить дисплей

clear

Вывести на дисплей содержимое файла

cat имя файла

Рассмотренные в данных методических указаниях команды входят в минимально-необходимый набор команд при работе в ОС UNIX с использованием CLI в данном лабораторном практикуме.

1.3.5 Дополнительные возможности CLI ОС UNIX

Для увеличения гибкости и удобства использования CLI, в ОС UNIX реализованы дополнительные возможности.

1. Перенаправление ввода-вывода позволяет направить поток символов, которые генерирует программа не на терминал, а в текстовый файл (см. рисунок1). Это поможет, например, создать 'скриншот' результатов работы программы при оформлении отчёта по лабораторной работе. Аналогично можно перенаправить поток ввода от клавиатуры, 'заставив' программу получать входной поток символов из файла;
2. Генерация строк (имени файла или каталога), с использованием шаблона. Шаблон - специального вида строка, описывающую целый класс строк. Таким образом, пользователь может применять групповые команды, манипулирующие не одним, а множеством объектов файловой системы (удалять, копировать, переименовывать, создавать и.т.д). Например команда rm *.o удалит все объектные файлы (с расширением .o) в текущем каталоге, так как * заменяет любое количество любых символов в имени файла.
3. Повтор, редактирование ранее введённых команд. Командный интерпретатор сохраняет в буфере все введённые пользователем команды. Нажатием клавиш «курсор вверх» и «курсор вниз» пользователь может выбрать нужную команду, из буфера и повторно исполнить эту команду нажатием «ENTER». Команда также может быть отредактирована для изменения параметров и/или исправления ошибок. Перемещение курсора в пределах строки выполняется клавишами «курсор влево» и «курсор вправо». Эта возможность особенно важна при редактировании громоздких строк, содержащих большое число аргументов и ключей.
4. Автоматическое достраивание выражений. Заключается в том, что командный интерпретатор добавляет к уже введённой пользователем части строки (имени файла) его возможное окончание — имя существующего файла. Это происходит при нажатии клавиши TAB. Если к введённой пользователем части имени подходит несколько окончаний, очередные варианты просматривается повторными нажатиями TAB.
5. При разработке программного обеспечения, удобно запустить текстовый редактор для редактирования исходного кода программы в одном терминале, а команды компиляции программы этой же программы выполнять в другом терминале, переключаясь между текстовыми 'окнами' с помощью Alt + Fx, например Alt+F1 <=> Alt+F2.
6. Компиляцию и запуск программы удобно запускать одной составной командой, реализующей 'условную операцию' gcc prog.c -o prog или ./prog. Здесь команда запуска программы ./prog будет выполняться, только если при выполнении команды компиляции gcc prog.c -o prog не было обнаружено ошибок.

sudo halt	выключение компьютера	*mkdir <имя подкаталога>*	создание каталога
halt	стоп системы	*rmdir <имя подкаталога>*	удаление каталога (пустого)
ls	содержимое текущего каталога	*mv <исходное имя файла (подкаталога)> <новое имя (подкаталога)>*	перемещение и переименование файла или подкаталога
ls /home/student	содержимое каталога /home/student	*cp <исходное имя файла (подкаталога)> <новое имя (подкаталога)>*	копирование файла или подкаталога
ls -l	содержимое текущего каталога (подробный формат)	*ps*	определить идентификатор процесса (PID), который соответствует запущенной программе
ls -s /etc	содержимое каталога /etc и размер каждого элемента каталога в блоках	*kill -9 <PID>*	завершить программу
ls <имя подкаталога>	вывода на дисплей содержимого каталога	*nano <имя файла>. <расширение файла>*	запустить текстовый редактор nano
ls / -R&	содержимое всех подкаталогов рекурсивно	*gcc <имя программы>.c –o <имя программы>*	запустить программу компиляции
man <имя объекта>	справочная информация (man ls – информация о команде ls)	*gdb <имя программы>*	запустить откомпилированную программу в режиме отладки («под отладчиком»)
cd <имя подкаталога>	смена текущего каталога	*clear*	очистить дисплей
pwd	вывод на дисплей текущего каталога	*cat <имя файла>*	вывести на дисплей содержимое файла
touch <имя файла>	создание пустого файла	*./ <имя программы>*	команда запуска программы
rm <имя файла>	удаление файла	echo <сообщение>, или echo <$имя_переменной_окружения>	Вывести на дисплей сообщение или переменную окружения
>>	Команда вывода в файл	Cd ..	Переход на каталог выше

1.4 Текстовый редактор nano

Предназначен для редактирования текстовых файлов с использованием алфавитно-цифровых терминалов. Чтобы открыть файл для редактирования или просмотра, выполните команду:

nano <имя файла>

или

nano -v <имя файла>

Команда nano без параметров запускает редактор без загрузки любого файла. После старта, пользователь управляет редактором, в основном, одновременным нажатием клавиши CTRL и о одной из алфавитно-цифровых клавиш. Наиболее употребительные сочетания, выведены в строке подсказки в нижней части экрана. Полный набор команд и сочетаний клавиш можно просмотреть с помощью сочетания CTRL+G. Выход из режима просмотра подсказки (и других режимов) — сочетание CTRL+C. Для подтверждения выбора используется клавиша ENTER, а также Y – положительный ответ, или N – отрицательный ответ.

Некоторые команды приводят к переходу во вложенное меню. Это меню также отображается в нижней части экрана.

Таблица 1 - Часто применяемые команды редактора nano

№	Действие	Сочетание клавиш	Примечание
1	Загрузить файл (вставить содержимое файла в текст)	CTRL + R	Ввести имя файла для чтения и нажать ENTER или
			Нажать CTRL+ t для перехода к выбору файла из списка и нажать ENTER
2	Записать отредактированный текст в файл	CTRL+ O	Если редактируется уже существовавший файл, будет предложено сохранить его под старым именем, нажав ENTER. Для сохранения редактируемого файла под другим именем — наберите это имя и нажмите ENTER.
3	Вырезать из текста строку на которой находится курсор и поместить в буфер	CTRL + K	
4	Вставить строку из текстового буфера на строку выше той, на которой находится курсор	CTRL + U	
5	Начать выделение текста	CTRL + ^	для отмены выделения — повторно нажать CTRL + ^
			для записи выделенного фрагмента в буфер - CTRL + K
6	Найти строку	CTRL + W	Для 1-го поиска — ввести искомую строку и нажать ENTER
			Для следующего поиска - нажать ENTER
7	Перейти на строку номер N	CTRL + _	Ввести номер строки и нажать ENTER
8	Включить/выключить постоянную индикацию позиции курсора	ESC , C	Если позиция курсора не отображается, нажать и отпустить escape, потом нажать и отпустить C
9	Включить/ выключить «сглаживание при прокрутке»	ESC, S	Для повышения плавности «прокрутки» текста нажать и отпустить escape, потом нажать и отпустить S
10	Выйти из редактора	CTRL + X	Если файл не был записан после изменения, появится запрос на запись. Ответить Y или N или отменить выход (CTRL + C)

Задание

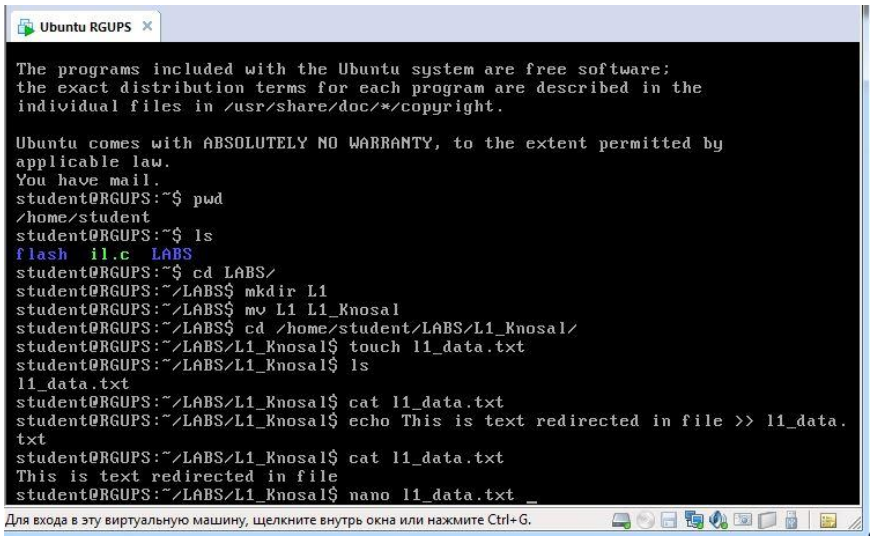
Выполните команды для реализации следующих действий:

1. Включите компьютер и войдите в систему
2. Выведите на дисплей имя текущего каталога
3. Выведите на дисплей содержимое текущего каталога
4. Сделайте каталог LABS текущим каталогом
5. Создайте в текущем каталоге подкаталог L1
6. Переименуйте созданный каталог в L1_NAME (где name – фамилия пользователя)
7. Сделайте каталог L1 текущим каталогом
8. Создайте пустой файл l1_data.txt
9. Выведите строку «Этот текст перенаправлен в файл» с помощью команды echo и механизма перенаправления в файл l1_data.txt
10. Отредактируйте текст в файле l1_data.txt, добавив «перенаправлением вывода команды echo», выйдите из редактора по Ctrl+X, сохранив изменения.
11. Очистите дисплей
12. Выведите содержимое файла l1_data.txt на дисплей (не редактором nano)
13. Скопируйте файл l1_il.c из каталога /home/student/LABS/DATA в текущий каталог
14. Удалите все файлы в каталоге L1_NAME

15. Удалите каталог L1_NAME
16. Выключите компьютер (с помощью команды)

Пример выполнения

1. Включите компьютер и войдите в систему
2. Выведите на дисплей имя текущего каталога
3. Выведите на дисплей содержимое текущего каталога
4. Сделайте каталог LABS текущим каталогом
5. Создайте в текущем каталоге подкаталог L1
6. Переименуйте созданный каталог в L1_NAME (где NAME – фамилия пользователя)
7. Сделайте каталог L1 текущим каталогом
8. Создайте пустой файл l1_data.txt
9. Выведите строку «Этот текст перенаправлен в файл» с помощью команды cat, echo и механизма перенаправления в файл l1_data.txt



10. Отредактируйте текст в файле l1_data.txt, добавив «перенаправлением вывода команды echo», выйдите из редактора по Ctrl+X, сохранив изменения.

11. Очистите дисплей

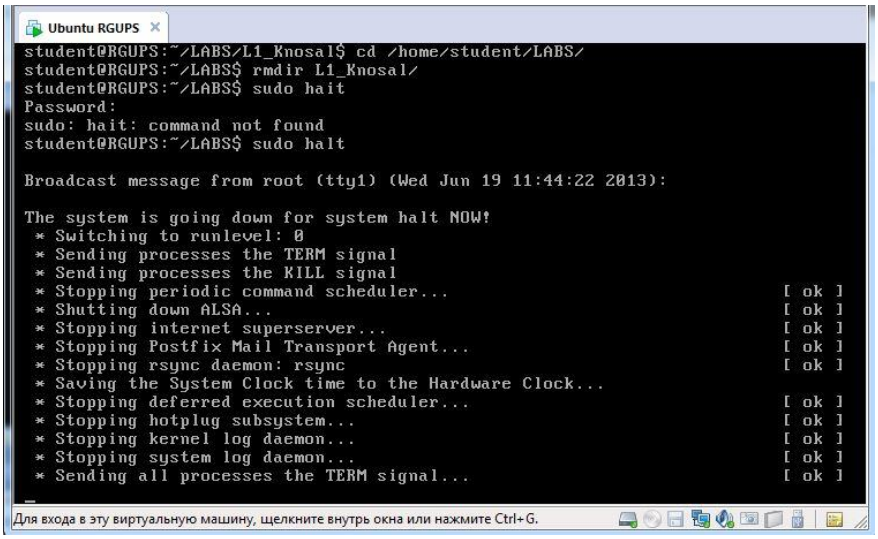
12. Выведите содержимое файла l1_data.txt на дисплей (не редактором nano)
13. Скопируйте файл l1_il.c из каталога /home/student/LABS/DATA в текущий каталог

```
student@RGUPS:~/LABS/L1_Knosal$ cat l1_data.txt
This is text redirected to a file redirect the output of the echo
command
student@RGUPS:~/LABS/L1_Knosal$ cp /home/student/LABS/DATA/l1_il.c /home/student
/LABS/L1_Knosal/
```

14. Удалите все файлы в каталоге L1_NAME
15. Удалите каталог L1_NAME

```
student@RGUPS:~/LABS/L1_Knosal$ ls
l1_data.txt  l1_il  l1_il.c
student@RGUPS:~/LABS/L1_Knosal$ rm l1_il
student@RGUPS:~/LABS/L1_Knosal$ rm l1_il.c
student@RGUPS:~/LABS/L1_Knosal$ rm l1_data.txt
student@RGUPS:~/LABS/L1_Knosal$ cd ..
student@RGUPS:~/LABS$ rmdir L1_Knosal/
student@RGUPS:~/LABS$ _
```

16. Выключите компьютер (с помощью команды)



КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

1 Что такое интерфейс командной строки (CLI)?

2 Какой формат имеют команды в ОС UNIX?

3 Какие дополнительные средства CLI ОС UNIX вы знаете?

Лабораторная работа №6-7
«Редактирование, резервное копирование и восстановление реестра»

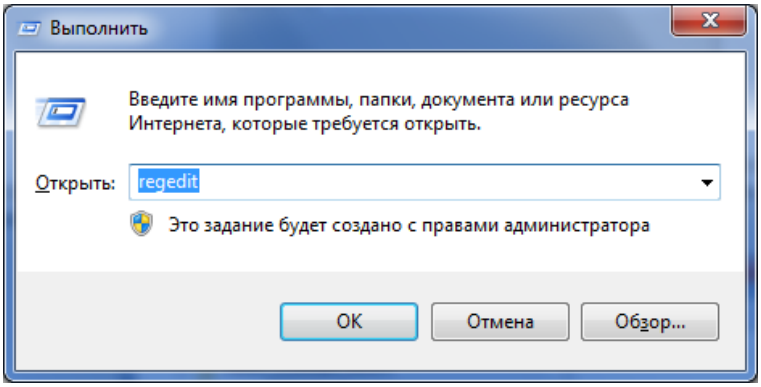
Цель работы:

1. Изучение структуры и назначения реестра Windows.
2. Овладение средствами управления реестром.
3. Овладение средствами резервного копирования и восстановления реестра.
4. Овладение навыками редактирования и очистки реестра.
5. Приобретение навыков по управлению системными службами.

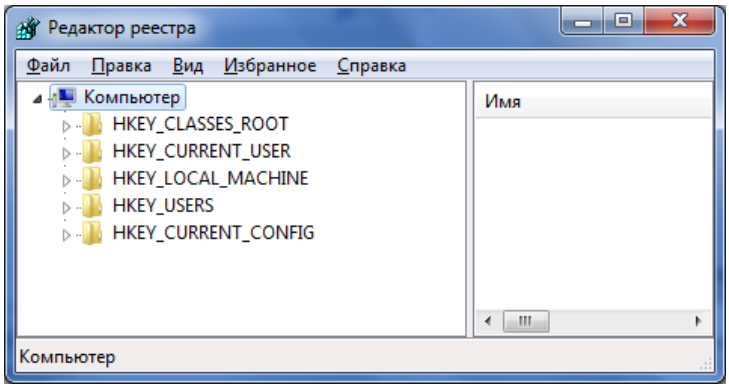
1 Краткие теоретические сведения

1.1. Назначение и структура реестра

Для просмотра, внесения изменений и очистки реестра компьютера можно использовать программу редактор реестра, который находится по следующему пути: C:\WINDOWS\system32\regedt32.exe.

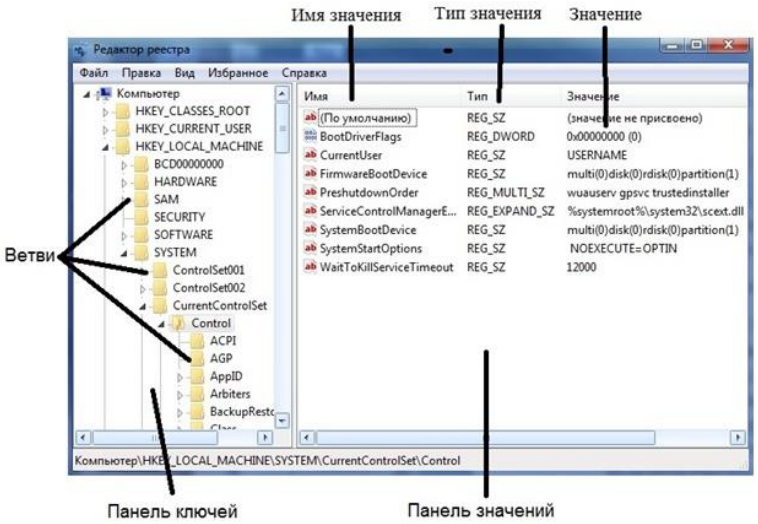


Редактор реестра позволяет просматривать все ключи и значения, которые находятся в реестре, а также по необходимости изменять настройки Windows, программ, или драйвера устройств.



В появившемся окне «Редактора Реестра», обратите внимание на то, что с левой стороны окна расположена панель ключей, а с правой стороны – панель значений. Панель ключей отображает корневые ключи и подключи Реестра. Щелкая манипулятором мышь по корневым ключам Реестра, отобразите слева его иерархию. Панель значений справа демонстрирует настройки, содержащиеся в каждом из подключей. Щелкните по одному из них на панели ключей и найдите его значения на панели значений.

Термин «Ветвь» указывает на ключ и все его подключи.



Системный реестр имеет иерархическую структуру, которая подобна структуре каталогов на Вашем жестком диске, а Regedit подобен Проводнику Windows. Реестр содержит три типа объектов: ключи, параметры и значения.

Ключи - вершина иерархической структуры реестра. Под ключами реестра могут располагаться другие узлы иерархического дерева (подключи). Кроме этого, каждый ключ может содержать один или несколько параметров. Все ключи и параметры в пределах подключа должны иметь уникальные имена.

Параметры имеются у каждого ключа и подключа. У каждого ключа обязательно есть хотя бы один параметр – «По умолчанию». Если значения параметров не заданы, то они имеют значение Null.

Параметры состоят из трех частей: тип параметра, имя параметра и его значение. Допустимы следующие **типы параметров**: двоичные, двойное слово и строковые. Каждому типу параметров соответствует своя пиктограмма в окне редактора реестра.

String (строковое). Представляет из себя ASCIIZ–строку (заканчивается символом с кодом 0). Имеет переменную длину, максимальный размер 64 кБ. Значение строки всегда заключается в кавычки.

Binary (двоичное). Максимальный размер 64 кБ. В окне редактора реестра представлено в виде 16-ричного значения.

DWORD (двойное слово). Представляет собой число размером 32 бита (в реестре 8-значное шестнадцатеричное число). Чтобы отличить этот тип данных от двоичного, перед численным значением DWORD всегда есть два символа: 0x.

Вся база системного реестра разделена на шесть основных разделов, которые принято называть ветвями. Каждая ветвь содержит в себе параметры, относящиеся к определенному набору ключей. Ниже кратко описано назначение этих разделов.

Вид ключа	Описание
HKEY_CLASSES_ROOT (HKCR)	Описывает тип файла, расширение файла, и OLE информацию.
HKEY_CURRENT_USER (HKCU)	Содержит имена пользователей, вошедших в систему Windows, и их настройки.
HKEY_LOCAL_MACHINE (HKLM)	Содержит специфическую информацию об установленном оборудовании, настройках программного обеспечения и другую информацию. Эта информация используется для всех пользователей, зарегистрированных на данном компьютере, и является одним из наиболее часто обращаемых мест в реестре.
HKEY_USERS (HKU)	Содержит информацию обо всех пользователях, зарегистрированных на компьютере, в том числе как общую, так и специфическую информацию по каждому пользователю.
HKEY_CURRENT_CONFIG (HKCC)	Содержит подробности о текущей конфигурации оборудования подключенного к компьютеру.
HKEY_DYN_DATA (HKDD)	Используется только в операционной системе Windows 95, 98 и NT. Данный раздел реестра содержит информацию о динамическом статусе и информацию по устройствам Plug-и-Play. Раздел может изменяться в связи с добавлением или удалением устройств из компьютера. Информация для каждого устройства включает в себя соответствующий аппаратный ключ, текущее состояние устройства, в том числе возможные проблемы в его работе.

1.1.1 Состав основных разделов

Каждый из вышеперечисленных разделов содержит в себе другие разделы — как и файловая система, Registry имеет структуру дерева. Каждый узел (раздел или подраздел) называется ключом. Вы можете открывать новые ветви до тех пор, пока не доберетесь до уровня, на котором находятся только параметры.

1. **Hkey_Classes_Root**

Структура раздела несколько отличается от всех остальных. Для каждого зарегистрированного расширения файла имеется подключ (например, .bmp).

Значение этого ключа "По умолчанию" указывает на подключ описания документа («ACDC_BMP»), который расположен в той же ветви основного раздела. В подкл. описания документа и содержится цепочка ключей, хранящих информацию об ассоциациях, OLE, DDE.

2. **Hkey_Local_Machine**

Информация, сохраненная здесь, используется приложениями, устройствами и системой, и не зависит от того, кто был заявлен в качестве пользователя. Устройства могут помещать информацию в системный реестр с помощью P1ug&Play-интерфейса, программные средства — посредством стандартного API. Hkey_Local_Machine содержит ряд подразделов.

Раздел	Назначение
Config	Различные конфигурации компьютера.
Enum	Информация о подключенных к данному компьютеру устройствах.
Hardware	Информация о последовательных интерфейсах и модемах, которые используются программой HyperTerminal.
Network	Хранящаяся здесь сетевая информация создается при входе пользователя в сеть: имя пользователя, регистрационная информация, первичный поставщик услуг и другие сведения.
Security	Информация о том, какой компьютер в сети следит за безопасностью сети и поддерживает ли (допускает ли) данный компьютер удаленное управление.
Software	Информация о программных средствах, установленных на данном компьютере, и различные конфигурационные данные программ.
System	Информация данного раздела управляет запуском системы, загрузкой драйверов устройств, сервисом Windowsи поведением системы.

2.1 ***Подраздел Config***

− Содержит информацию о различных конфигурациях аппаратных средств.

− Каждая конфигурация имеет уникальное обозначение и хранится в отдельном подразделе с соответствующим именем.

− Конфигурации перечислены в списке в окне утилиты **Система.** Здесь же их можно обрабатывать.

− При запуске Windows проводится проверка конфигурации аппаратных средств. При этом может произойти следующее:

− В большинстве случаев конфигурационные данные позволяют Windows автоматически выбрать соответствующую конфигурацию.

− При первом после изменения оборудования запуске компьютера Windows создает новый элемент конфигурации для новых конфигурационных данных. В результате создается и новый Config-элемент в системном реестре.

− Когда конфигурационные данные не позволяют системе Windows однозначно решить, какую из описанных конфигураций следует выбрать, пользователю при загрузке системы предлагается меню, посредством которого он может выбрать подходящую конфигурацию.

2.2 ***Подраздел Enum***

− Windows располагает специальными программами, которые отвечают за построение дерева аппаратуры в системном реестре (например, Диспетчер устройств, вызываемый через Панель управления - Система-Устройства).

− Каждому устройству присваивается уникальный идентификационный код.

− В системном реестре хранится идентификационная информация о каждом устройстве, например, тип устройства, идентификационный код (ID) устройства, информация об изготовителе и информация, о драйвере.

2.3 ***Подраздел Software***

Содержит информацию о каждом программном средстве, установленном на компьютере.

Содержимое этого раздела является общим для всех пользователей данного компьютера.

Hkey_Local_Machine\Software содержит ряд подразделов и сведения о различных подразделах (их описание), которые могут появиться в системном реестре.

2.4 ***Подраздел System***

− Данные в подразделе System содержат все параметры драйверов устройств и служб, используемые при запуске Windows.

− Вся информация хранится в подразделе CurrentControlSet. Он содержит два следующих подраздела:

− Control: Подраздел включает информацию, используемую, при запуске системы, например, сетевое имя компьютера и запускаемые подсистемы.

− Services: Подраздел включает информацию, необходимую для контроля загрузки и конфигурирования драйверов, файловой системы, и др. Здесь также определяется, как отдельные службы вызывают одна другую.

4. **Hkey_Current_User и Hkey_Users**

− Содержит Default-подраздел и подразделы для всех пользователей, заявленных в системе.

− Информация из подраздела Default используется для того, чтобы создать конфигурацию для нового пользователя.

− Hkey_Current_User содержит информацию о пользователе, работающем на компьютере в текущем сеансе (см. табл.6).

Если существуют одинаковые параметры в Hkey_Local_Machine и Hkey_Current_User, то используются значения параметров, взятые из Hkey_Current_User.

Подраздел	Хранящаяся в подразделе информация
AppEvents	Пути и имена звуковых файлов, используемых для генерации звуков при определенных событиях в системе.
Control Panel	Установки из Панели управления.
Keyboard layouts	Текущая раскладка клавиатуры.
Network	Информация о текущем состоянии сети.
InstallLocationsMRU	Путь к установочным файлам.
Software	Установки активного пользователя, определяющие режимы работы программ (приложений).

5. **Hkey_Current_Config и Hkey_Dyn_Data**

− Hkey_Current_Config указывает на текущую системную конфигурацию, которая сохранена в Hkey_Local_Machine\Config.

− Часть системной информации в Windows должна постоянно присутствовать в оперативной памяти, поскольку системе необходим быстрый доступ к этой информации и Windows не может ожидать, пока нужные данные будут прочитаны с жесткого диска. Вся эта информация находится в Hkey_Dyn_Data.

− Подраздел Hkey_Dyn_Data\ConfigurationManager, называемый также деревом аппаратуры, представляет собой хранящееся в оперативной памяти описание текущей системной конфигурации.

− Дерево аппаратуры создается заново при каждом запуске системы и адаптируется, если в состав или конфигурацию аппаратуры были внесены изменения. Присутствующие в этом разделе данные можно просмотреть с помощью Редактора реестра, они всегда соответствуют текущему состоянию аппаратуры компьютера.

− Hkey_Dyn_Data содержит статистическую информацию о различных сетевых компонентах в системе. Она находится в подразделе PerfStats.

Реестр представляет собой реляционную базу данных, в которой аккумулируются вся необходимая для нормального функционирования компьютера информация о настройках операционной системы, а также об используемом совместно с Windows программном обеспечении и оборудовании.

С реестром взаимодействуют следующие компоненты операционной системы.

1. *Программы установки* (WindowsSetup). Каждый раз при запуске программы WindowsSetup или установочных программ для аппаратных и программных средств происходит добавление в реестр новых конфигурационных данных. Реестр позволяет приложениям совместно использовать конфигурационную информацию и предоставляет им больше возможностей взаимодействия между собой.

2. *Распознаватель* (Recognizer). При запуске компьютера распознаватель аппаратных средств помещает в реестр список обнаруженных устройств. На компьютерах с процессорами Intel распознавание аппаратных средств осуществляется программой Ntdetect.com и ядром операционной системыNtoskrnl.exe.

3. *Ядро системы*(Ntoskrnl.exe). При старте системы ядро извлекает из реестра сведения о загружаемых драйверах устройств и порядке их загрузки. Кроме того, ядро передает в реестр информацию о себе (номер версии и др.).

4. *Драйверы устройств*. Драйверы обмениваются с реестром параметрами загрузки и конфигурационными данными. Каждый драйвер устройства должен сообщить об используемых им системных ресурсах, включая аппаратные прерывания и каналы DMA. Приложения и драйверы устройств могут считывать эту информацию из реестра, предоставляя пользователям интеллектуальные программы инсталляции и конфигурирования.

5. *Административные средства*. Эти средства, в том числе утилиты панели управления и оснастки, собранные в меню *Администрирование*, представляют собой удобные и безопасные (в части внесения ошибок) средства модификации реестра. Редактор реестра также полезен для просмотра реестра и для внесения изменений в конфигурацию системы.

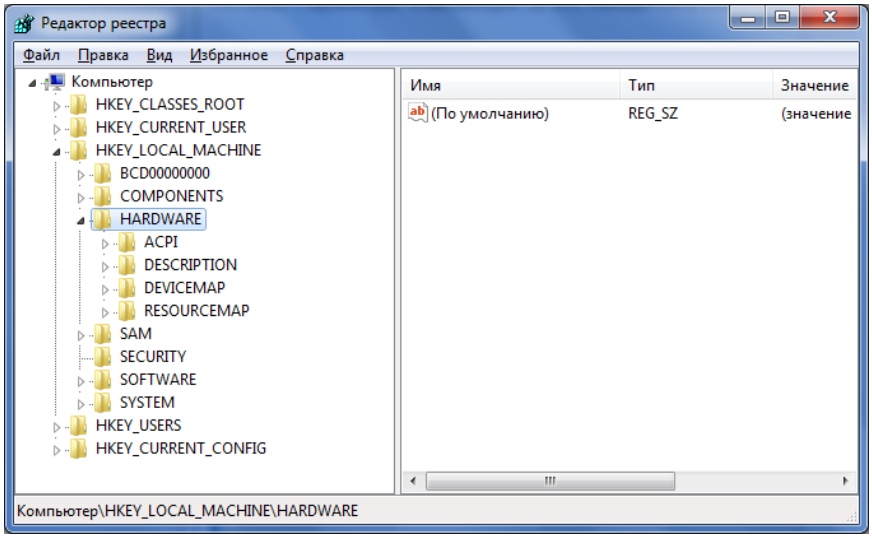
6. *Пользовательские профили* (userprofiles). WindowsNT/2000/XP обеспечивают возможность создания множества пользовательских профилей. Вся информация, относящаяся к конкретному пользовательскому имени и ассоциированным с ним правам, хранится в реестре.

7. *Аппаратные профили* (hardwareprofiles), или *профили оборудования*. Реестр позволяет хранить множественные аппаратные конфигурации. Аппаратный профиль представляет собой набор инструкций, с помощью которого можно указать операционной системе, драйверы каких устройств должны загружаться при запуске компьютера.

При редактировании системного реестра WindowsXP в специальных программах он представляется в виде единой базы данных, имеющей жесткую иерархическую структуру. Однако на физическом уровне реестр неоднороден и состоит из множества файлов, каждый из которых отвечает за собственный объем представленной в этой базе информации. Некоторые из отображаемых в реестре сведений вообще не хранятся в виде файлов на диске, а помещаются в оперативную память компьютера в процессе загрузки операционной системы и утрачиваются в момент отключения питания (энергозависимые разделы реестра).

В частности, к энергозависимым разделам реестра относится ветвь HKEY_LOCAL_ MACHINE\HARDWARE, в которой находятся сведения об оборудовании компьютера и системных ресурсах,

назначенных устройствам: о запросах на прерывание (IRQ), каналах прямого доступа к памяти (DMA) и диапазонах памяти ввода-вывода (I/ORange).



Другие части реестра, хранящие данные о базовой конфигурации операционной системы, ее настройках и параметрах, содержатся в системной папке **%SystemRoot%\System32\Config.**

Файлы, хранящие данные о профилях пользователей, находятся в папке **%SystemRoot%\Profiles**.

Данные, относящиеся к конкретным настройкам системы для каждого пользователя и данные об их персональной конфигурации рабочей среды, представлены в папках **%Drive%\DocumentsandSettings\%UserName%**, где **%Drive%** - имя раздела диска, на котором установлена операционная система, а **%User-Name%** - папка, имя которой соответствует имени одного из зарегистрированных в системе пользователей.

Дополнительные сведения о локальных пользователях системы по умолчанию содержатся в папке **%Drive%\DocumentsandSettings\LocalService**, а данные о настройках системы для удаленных пользователей - в папке **%Drive%\DocumentsandSettings\Net** - workService.

На логическом уровне реестр представляется иерархической структурой из четырех ступеней. Верхний уровень образуют так называемые *ветви* (HiveKeys), которые принято обозначать аббревиатурой HKEY_, где за символом подчеркивания следует название ветви. Всего в реестре пять ветвей:

1 HKEY_CLASSES_ROOT,
2 HKEY_CURRENT_USER,
3 HKEY_LOCAL_MACHINE,
4 HKEYJJSERS
5 и HKEY_CURRENT_CONFIG.

Второй ступенью являются *разделы,* или *ключи* (Keys). Они отображаются в программе *Редактора реестра* в виде подпапок ветвей HKEY_. Функционально ключи можно разделить на две условные категории: определяемые системой (их менять нельзя) и определяемые пользователем (эти имена могут быть изменены администратором, и такие изменения не приведут к каким-либо фатальным последствиям).

Ступенью ниже следуют *подразделы* (Subkeys). Их имена также могут быть определены системой или пользователем.

Последней ступенью в иерархической структуре системного реестра являются *параметры* (Values) - элементы реестра, содержащие саму информацию, определяющую работу операционной системы и компьютера в целом. Параметры представляют собой цепочку *имя параметра - тип данных -значение.*

Типы данных, определенные для параметров реестра, приведены в таблице ниже. Для значений параметров реестра вне зависимости от того, к какому типу данных они относятся, в программе *Редактор реестра* имеется набор встроенных мастеров, позволяющих легко изменять данные любого типа.

Таблица. Типы данных реестра**.**

Наименование	Тип данных	Описание
REG_BINARY	Двоичный	Аппаратные компоненты используют информацию в виде двоичных данных. Редакторы реестра отображают ее в шестнадцатеричном формате
REG_DWORD	Числовой	Числа (4 байта), параметры драйверов устройств и сервисов. Редакторы реестра отображают ее в двоичном, шестнадцатеричном и десятичном формате
REG DWORD LITTLE_ENDIAN	Числовой	Эквивалент REGDWORDc младшим байтом в начале числа
REG_DWORD_ BIG_ENDIAN	Числовой	Эквивалент REG_DWORDco старшим байтом в начале числа
REG_SZ	Строковый	Описания компонентов
REG_EXPAND_SZ	Строковый	Расширяемая строка данных. Текст, содержащий переменную, которая может быть заменена при вызове со стороны приложения
REG_MULTI_SZ	Многостроковый	Списки текстовых строк в формате, удобном для восприятия
REG_LINK	Строковый	Символическая ссылка Unicode. Предназначен для внутреннего использования.
REG_NONE	Нет типа	Параметр не имеет определенного типа данных
REG_RE- SOURCE_LIST	Строковый	Список аппаратных ресурсов
REG_RE- SOURCE_ REQUIRE- MENTS_LIST	Строковый	Список необходимых аппаратных ресурсов
REG_FULL_RE- SOURCE_ DESCRIPTOR	Строковый	Дескриптор (описатель) аппаратного ресурса

1.2 Как программы используют реестр

Программы используют реестр несколькими способами. Они могут добавить данные в реестр путем создания новых или использование уже существующих ключей. Когда программа добавляется в реестр, данные сортируются по типу данных или типу конкретного пользователя. Благодаря этому различию приложения могут поддерживать несколько пользователей и определять данные профиля каждого пользователя.

Программа может закрыть ключ и записать содержащиеся в нем данные в реестр, и так же приложение может удалить значение из ключа или очистить сам ключ.

1.3 Проблемы, связанные с реестром

Пользователи могут испытывать трудности в работе компьютера, вызванные ошибками в системном реестре по нескольким причинам. Проблемы с самим компьютером, как правило, происходят из-за недействительных или отсутствующих ключей в реестре Windows. Некоторые признаки неблагополучия включают в себя сбой операционной системы, «зависание», или заметное снижение быстродействия компьютера в целом. Конечно, если знать, что может вызвать ошибки в системном реестре, то можно избежать большинства проблем, связанных с этим.

В основной своей массе, проблемы с реестром возникают от действий самого пользователя, главным образом, связанных с установкой и удалением программного обеспечения и аппаратных средств на компьютер. Если вы часто устанавливайте и удаляйте программы, удаляете автозапуск программ, меняете оборудование и не удалять старые или поврежденные драйвера, удаляете программное обеспечение не надлежащим образом, или же быть может, устанавливаете на компьютер программы со встроенным вирусом, то у вас, скорее всего, возникнут проблемы с системным реестром Windows.

2 Средства управления реестром

Вы можете редактировать реестр непосредственно с помощью редактора реестра, поставляемый с операционной системой. Однако необходимо проявлять большую осторожность, поскольку возникшие ошибки в реестре могут привести к сбою в работе компьютера или привести к его полной неработоспособности. Если же это все-таки произойдет, то вы всегда можете восстановить реестр, который использовался при последнем успешном запуске компьютера.

Редактирование реестра может быть достаточно сложной задачей, и поэтому вы должны прочесть файлы справки операционной системы Windows, прежде чем приступить к редактированию реестра самостоятельно.

Основным инструментом администрирования реестра Windows является программа *Редактор реестра*. Кроме *Редактора реестра* в Windows имеются средства консоли управления - специальные средства, позволяющие управлять многими аппаратными, программными и сетевыми компонентами Windows.

Часть программных средств, позволяющих изменять настройки реестра, расположена в *Панели управления* Windows.

Практически все параметры операционной системы, связанные с окружением пользовательской среды, ее возможностями и ограничениями, можно изменить при помощи *Редактора системных политик*. Возможно и управление реестром Windows из командной строки посредством консольного интерпретатора команд CMD. Можно также создать командный (пакетный) файл, включающий в себя набор команд среды CMD.

Сам *Редактор реестра* поддерживает собственный набор команд, последовательность которых может быть записана в файл. Путем использования таких файлов та или иная информация может быть автоматически добавлена в реестр.

По умолчанию *Редактор реестра*(утилитаReggdit.exe) в процессе установки операционной системы копируется в каталог %SystemRoot%. Для его запуска нужно выполнить команды *Пуск-Выполнить*, а затем в поле *Открыть* набрать команду regedit и щелкнуть по кнопке *ОК*.

Рабочее окно *Редактора реестра* разделено на две части. В левой (*Панель разделов*) отображается дерево ветвей, разделов и подразделов, из которых состоит реестр Windows. В правой части отображаются параметры, назначенные для выбранного элемента реестра.

Перемещаясь при помощи мыши по иерархической структуре реестра в левой части окна, в правой части можно просматривать параметры каждого из разделов.

Чтобы изменить значение какого-либо параметра, нужно дважды щелкнуть мышью по его значку. В открывшемся окне можно установить требуемое значение параметра.

В нижней части окна расположена строка состояния, которая отображает размещение выделенного элемента реестра. Включение и отключение строки состояния осуществляется установкой или снятием флажка *Строка состояния* в меню *Вид*.

Строка меню содержит пять подменю: *Файл*, *Правка*, *Вид*, *Избранное* и *Справка*.

- Меню *Файл* позволяет осуществлять операции экспорта-импорта файла реестра и его отдельных компонентов, выводить содержимое какого-либо фрагмента реестра на печать или открывать для редактирования файлы реестра других компьютеров, расположенных в локальной сети.
- Меню *Правка* позволяет выполнять операции создания, удаления, редактирования, переименования логических элементов реестра, выполнять процедуры поиска, настраивать режимы безопасности.
- Меню *Вид* управляет настройками интерфейса *Редактора реестра*.
- Меню *Избранное* позволяет устанавливать закладки на различные разделы и подразделы реестра для последующего быстрого перехода к ним.
- Меню *Справка* вызывает встроенную справочную систему *Редактора реестра*.

Изменение значений параметров реестра осуществляется встроенными мастерами редактора. Таких мастеров четыре - по количеству типов данных, характерных для значений параметров реестра. Соответствующий мастер вызывается автоматически двойным щелчком по параметру реестра.

Работа с реестром при помощи программы *Редактор реестра* состоит из следующих действий:

- просмотр разделов и подразделов (в данном случае механизм работы сходен с порядком работы пользователя в программе *Проводник*);
- поиск информации в реестре (меню *Правка-Найти-Найти далее*);
- быстрый переход к выбранному разделу реестра (используется механизм закладок, меню *Избранное-Добавить в избранное*);
- добавление новых разделов и параметров реестра (меню *Правка-Создать-Раздел*) или (*Правка-Создать*, далее из меню выбирается тип создаваемого параметра, например *Строковый параметр*);
- переименование, удаление и копирование раздела или параметра реестра (щелчок правой клавишей мыши по объекту переименования и использование контекстного меню).

Для того чтобы назначить пользователям или группам пользователей разрешения на доступ к какому-либо разделу реестра, нужно выполнить следующие действия (обладая правами администратора).

1. Открыть редактор реестра.
2. Выделить ветвь, раздел или подраздел, для которого требуется настроить разрешения.
3. В меню *Правка* выбрать команду *Разрешения*. На экране появится диалоговое окно настройки параметров безопасности для выбранного раздела реестра.

4. Щелкнуть мышью по кнопке *Дополнительно*. Откроется окно *Дополнительные параметры безопасности*.

5. На вкладке *Разрешения* этого окна в списке *Элементы разрешений* выбрать компьютер, пользователя или группу. Чтобы добавить новый элемент в список, нужно воспользоваться кнопкой *Добавить*.

6. Если необходимо настроить все разрешения для выбранного в списке пользователя или группы вручную, запретив автоматическое наследование разрешений от родительского объекта, следует сбросить флажок *Наследовать* от родительского объекта применимые к дочерним объектам разрешения, добавляя их к явно заданным в этом окне.

7. Для изменения списка разрешений для выбранного пользователя или группы щелкнуть по кнопке *Изменить*. В списке *Разрешения* открывшегося окна установить (сбросить) флажки, соответствующие разрешениям для данного пользователя или группы. Если флажок затенен, это означает, что разрешение наследуется. Чтобы изменить настройки такого разрешения, нужно выполнить действия, указанные в п. 6.

Для того чтобы дочерние объекты реестра автоматически наследовали разрешения от родительских объектов, в диалоговом окне *Дополнительные параметры безопасности* следует установить флажок *Заменить разрешения* для всех дочерних объектов заданными здесь разрешениями, применимыми к дочерним объектам.

Изменения в настройках разрешений вступят в силу после нажатия на кнопку *Применить*, а затем *ОК*.

Контроль над действиями пользователей с реестром администратор может осуществлять посредством аудита. **Для этого следует выполнить следующие действия.**

1. Открыть *Редактор реестра*.
2. Выделить ветвь, ключ или подраздел, для которого нужно настроить параметры аудита.
3. В меню *Правка* выбрать команду *Разрешения*. На экране появится окно настройки параметров безопасности для выбранного раздела реестра.
4. Щелкнуть по кнопке *Дополнительно*. Откроется окно *Дополнительные параметры безопасности*.
5. Перейти на вкладку *Аудит*.
6. Двойным щелчком выбрать в списке *Элементы аудита пользователя* или группу. **Если список пуст, следует выполнить следующие действия:**

1. щелкнуть мышью по кнопке *Добавить*. На экране появится окно *Выбор*: *Пользователи* или *Группа*;
2. для выбора типа объекта щелкнуть по кнопке *Типы объектов* и выбрать требуемый вариант (в нашем случае -*Пользователи*) и щелкнуть по кнопке *ОК*;
3. произойдет возврат в окно *Выбор*: *Пользователи* или *Группа*, в котором можно ввести имена пользователей, после чего щелкнуть по кнопке *Проверить имена ОК*;
4. на экране появится окно *Элементы аудита* для выбранной ветви реестра (в нашем случае HKEYJLOCAL_MACHINE).
7. Выбрав пользователя, следует установить флажки *Успех* или *Отказ* в группе *Доступ*.

Значения параметров для различных событий аудита реестра приведены ниже в таблице. Результаты аудита можно просматривать в системном журнале *Безопасность* с помощью оснастки *Просмотр событий*.

Таблица. Параметры для различных событий аудита реестра.

Событие	Тип аудита
Запрос значения	Аудит всех попыток чтения параметров из раздела реестра
Задание значения	Аудит всех попыток задания значений параметров в разделе реестра
Создание подраздела	Аудит всех попыток создания подразделов в данном разделе реестра
Перечисление подразделов	Аудит всех попыток определения подразделов данного раздела реестра
Уведомление	Аудит событий уведомления из подраздела реестра
Создание ссылки	Аудит попыток создания символьной ссылки в указанном реестре
Удаление	Аудит попыток удаления объектов реестра
Запись DAC	Аудит всех попыток записи в избирательную таблицу управления доступом для данного раздела
Смена владельца	Аудит всех попыток смены владельца выбранного раздела
Чтение разрешений	Аудит всех попыток открытия необязательной таблицы управления доступом для данного раздела

3 Как сделать резервную копию и восстановить реестр Windows

Если вы хотите сделать какие-нибудь изменения в реестре Windows, то перед выполнением этой процедуры рекомендуется сделать резервную копию. Особенно если вы неопытный пользователь, то это нужно сделать обязательно.

На самом деле сделать бэкап не совсем просто. Дело в том, что реестр нельзя скопировать и потом восстановить как обычный файл. По крайней мере, при запущенной операционной системе этого точно сделать нельзя.

Но есть способы, как сделать резервную копию отдельных разделов реестра Windows или восстановить весь реестр, используя функцию восстановления системы.

Некоторые версии операционных систем Windows, таких как Windows 98, создают автоматически резервную копию системного реестра каждый день, при условии, что в течение дня компьютер хотя бы раз перезагружался. К сожалению, ошибки реестра, которые вы, возможно, захотите исправить с помощью резервной копии, которую создал Windows автоматически, уже может содержать эти ошибки.

Поэтому желательно время от времени создавать свои копии системного реестра с помощью программ, а не полагаться на копии операционной системы Windows. Это поможет вам восстановить работоспособность компьютера при неудачной ручной очистки реестра.

1.1 Резервное копирование и восстановление отдельных разделов реестра

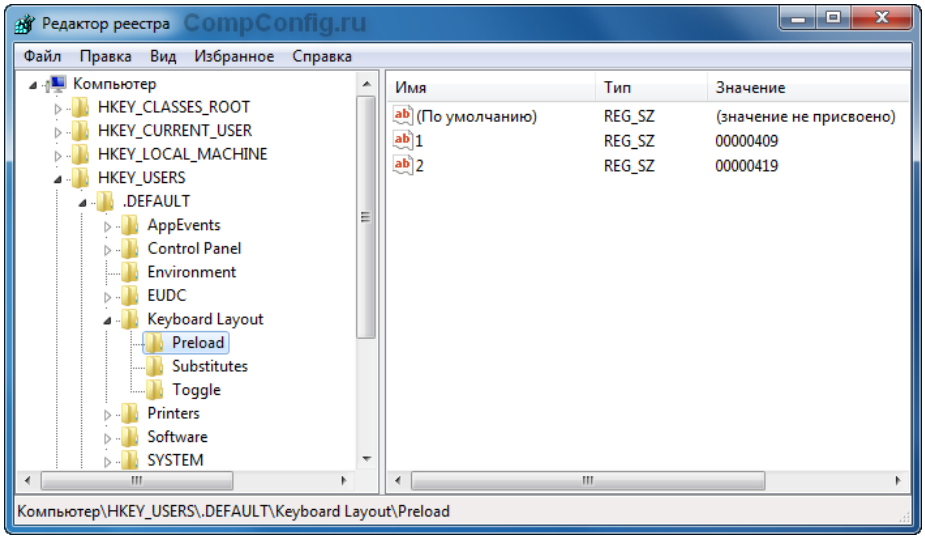
Как известно, многие программы, драйвера и сама операционная система хранят свои настройки в определенных разделах (ветвях) реестра. Поэтому если вы хотите изменить какие-то параметры, то достаточно знать, как сделать резервную копию того раздела, в котором вы хотите произвести изменения.

Этот процесс достаточно простой. Нужно только знать, в какой ветви хранятся необходимые нам настройки. Если вы хотите сохранить настройки для какого-то конкретного приложения, то необходимо найти соответствующий раздел в ветке HKEY_CURRENT_USER\Software или HKEY_LOCAL_MACHINE\Software.

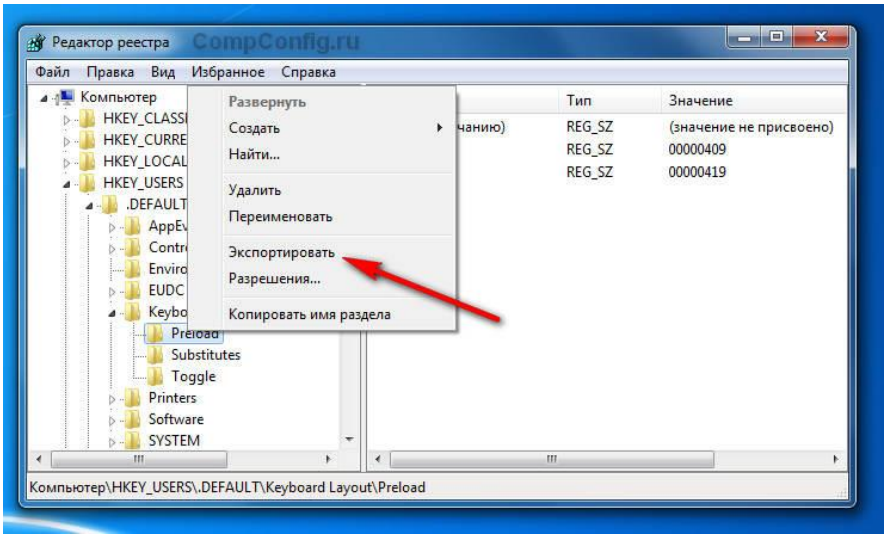
Некоторые приложения не используют системный реестр. Они могут хранить свои настройки в ini-файлах. Но для тех, что используют, вы можете легко сохранить настройки в файл и при необходимости использовать. Например, после переустановки Windows нет необходимости настраивать какую-либо программу с нуля. Достаточно установить ее и импортировать настойки в реестр из ранее сохраненного файла.

Давайте рассмотрим процесс сохранения раздела реестра на конкретном примере. Возьмем не параметры какого-либо приложения, а настройки самой ОС Windows. Допустим, вы захотели изменить язык ввода по умолчанию при загрузке системы. Параметры, отвечающие за это, находятся в следующем разделе:

HKEY_USERS\.DEFAULT\Keyboard Layout\Preload



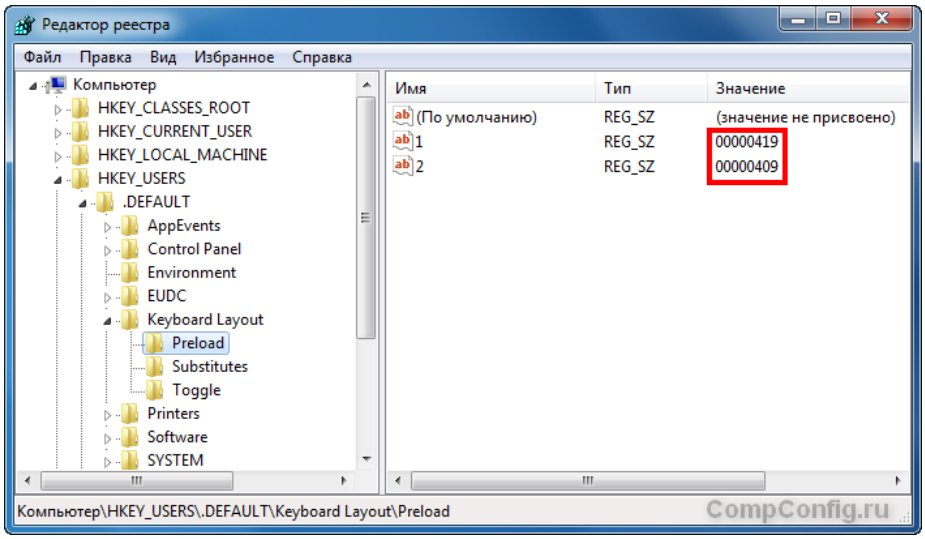
Для того, чтобы сделать резервную копию этого раздела нужно навести на него курсор и нажать правую кнопку мыши. В контекстном меню следует выбрать пункт «Экспортировать».



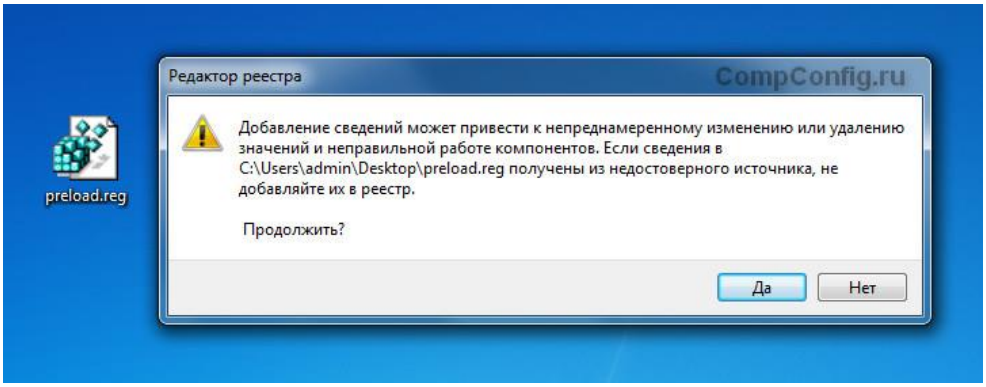
Далее откроется окно, в котором нужно указать путь для сохранения, имя файла экспорта и нажать кнопку «Сохранить». В нашем случае файл экспорта будет называться preload.reg.

Теперь, когда у вас есть резервная копия раздела реестра, в ней можно делать любые изменения.

В нашем примере, чтобы изменить язык ввода по умолчанию нужно поменять значения параметров «1» и «2» местами.



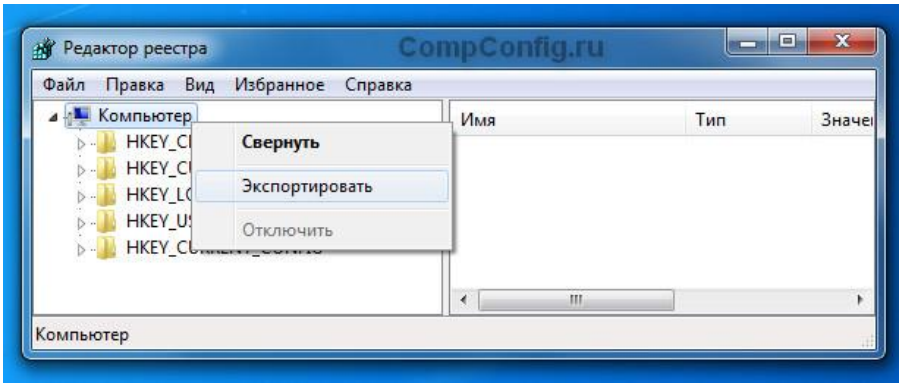
Чтобы восстановить раздел, достаточно запустить, ранее сохраненный файл. В нашем случае запускаем preload.reg.



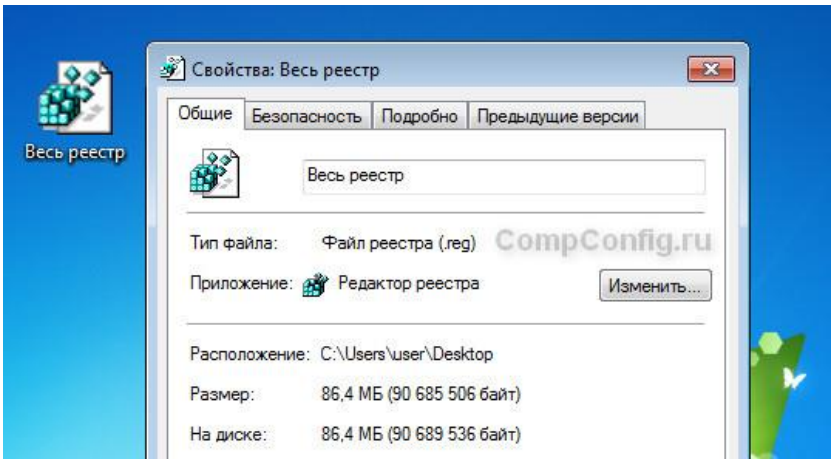
Используя эту технику, вы можете делать резервные копии ветвей реестра, содержащих настройки системы или конкретных программ.

1.2 Резервное копирование всего реестра

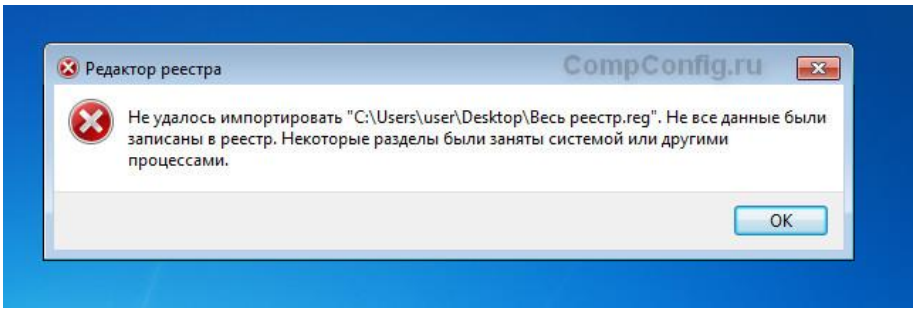
Применив способ, описанный выше, можно сделать полную резервную копию реестра.



Полученный файл будет иметь большой размер. Для только что установленной Windows 7 Максимальная x64 размер reg-файла составил 86 мегабайт.



Если потом запустить этот файл, то мы получим сообщение об ошибке. Оно появляется, потому что Windows не может восстановить все разделы и ключи реестра, т.к. многие из них используются в данный момент системой или другими процессами.



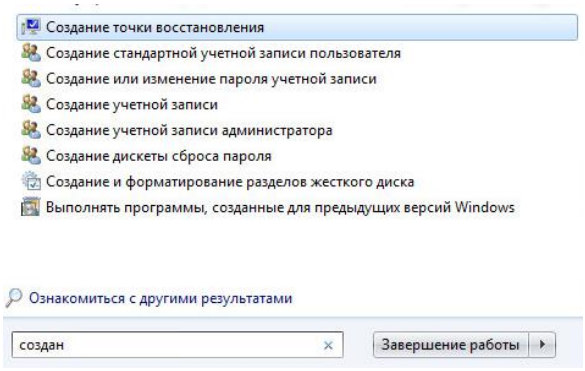
Не рекомендуется использовать этот метод резервного копирования для всего реестра. При его выполнении вы не знаете, какие именно разделы и ключи были сохранены, поэтому при последующем восстановлении нет никакой гарантии, что это вам поможет, а в худшем случае реестр можно наоборот повредить.

Чтобы не проводить эксперименты над своим компьютером, лучше использовать другой способ резервного копирования реестра Windows.

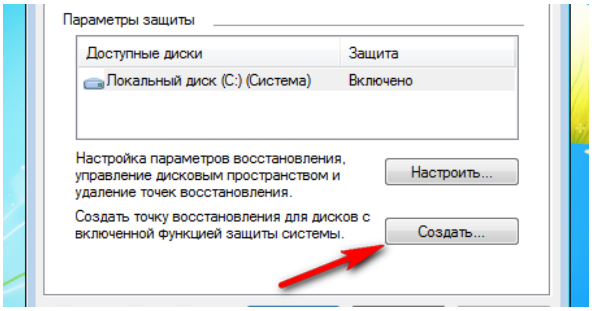
1.3 Правильное резервное копирование реестра с использованием функции восстановления системы

Если вы собираетесь вносить какие-либо серьезные изменения в реестр (установка драйверов, программного обеспечения, просто ручная правка), то для того, чтобы сделать резервную копию достаточно создать точку восстановления системы.

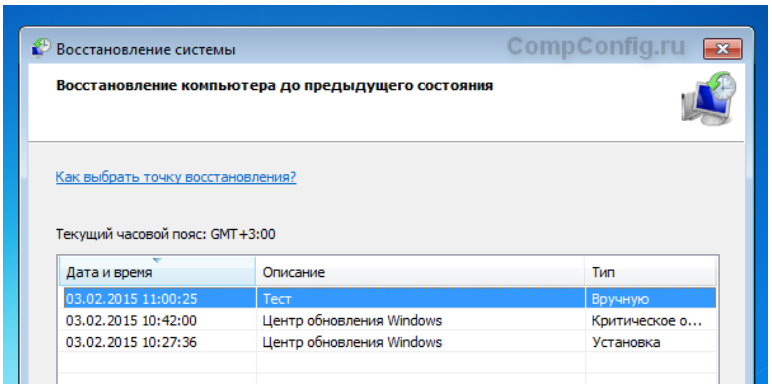
Чтобы это сделать нужно в меню «Пуск» в стоке поиска начать набирать фразу «создание точки восстановления». Потом в результатах поиска следует запустить необходимый пункт.



На вкладке «Защита системы» нужно нажать кнопку «Создать», ввести описание, и точка восстановления будет немедленно создана.



Восстановление осуществляется также легко, как и резервное копирование. Через строку поиска в меню «Пуск» найдите и запустите инструмент «Восстановление системы». После этого запустится мастер, в котором нужно выбрать нужную точку и запустить процесс восстановления.



После перезагрузки компьютера вы получите версию реестра на момент создания точки восстановления.

4 Очистка реестра

Записи приложений содержатся в разделах HKEY_CURRET_USER\Software (персональные параметры пользователя) и HKEY_LOCAL_MACHINE\SOFTWARE (параметры, общие для всех пользователей компьютера). После удаления многие приложения оставляют в реестре свои параметры, как правило, системные или пользовательские, что может привести к следующим проблемам.

1. *Неэффективное использование системных ресурсов*. Windows «верит» всему, что записано в реестре. Если реестр содержит запись о том, что для обработки файла или решения задачи в системе имеется определенное приложение, то Windows будет искать его, потребляя системные ресурсы, даже если приложения на самом деле нет.
2. *Нарушение надежности системы*. Проблемы с устойчивостью системы начинают проявляться, когда компьютер пытается использовать несуществующий ресурс. Вообще система успешно восстанавливается после одиночных и даже двойных ошибок. Но по мере накопления в реестре некорректных записей правильно реагировать на ошибки становится труднее и начинают появляться проблемы.
3. *Нарушение надежности приложений*. Приложения получают данные из ряда источников - от других приложений, драйверов устройств, операционной системы. Если одно приложение посредством Windows посылает запрос другому приложению, но оказывается, что оно удалено из системы, то запрашивающее приложение потерпит крах по причине некорректной записи в реестре, если оно не обладает хорошим механизмом обработки ошибок.

4. *Нарушение безопасности*. Безопасность компьютера может пострадать из-за «мусора» в реестре. Вирусам проще скрыть себя среди устаревших записей, а антивирусным приложениям труднее их обнаружить. Более того, устаревшие записи способствуют проникновению в систему вредоносных программ. Например, загрузка из-за некорректной записи в реестре в память ненужной библиотеки DLL делает возможными атаки на систему через эту библиотеку.

Специальные средства очистки реестра помогают в поиске устаревших данных установки приложений, однако не гарантируют обнаружения всех этих данных. Большинство средств очистки ищут некорректные параметры, но с этими параметрами часто связано множество записей. Для самостоятельного поиска таких записей можно воспользоваться рядом критериев: имя разработчика; имя приложения; имя исполняемого файла приложения; имена DLL; GUID компонентов (GloballyUniqueIdentifier - глобально уникальный идентификатор); URL приложения; уникальные данные настройки приложения.

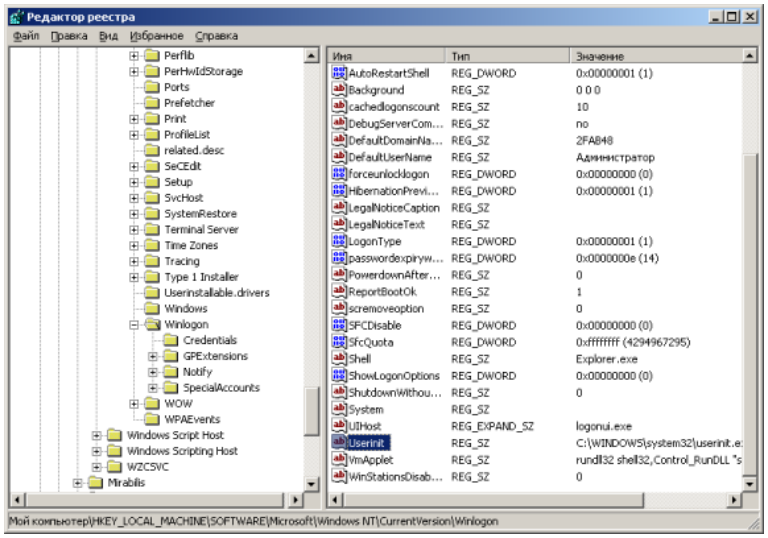
Хорошие утилиты для очистки реестра способны значительно упростить обнаружение некорректных записей и снизить вероятность удаления нужных. Перед очисткой реестра следует не забывать делать его резервную копию. Утилиты очистки резервную копию реестра создают автоматически.

Помимо regedit реестр можно править с помощью специальных программ сторонних производителей. Одна из наиболее распространенных программ называется CCleaner. Она бесплатная и имеет достаточно простой интерфейс.

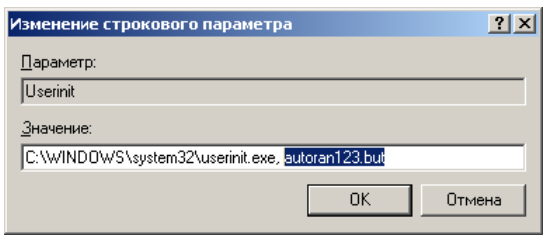
Задание 1

Проверить потенциальные места записей «троянских программ» в системном реестре операционной системы Windows.

1. Откройте Редактор реестра.
2. В ветви **HKEY_LOCAL_MACHINE** выберите **Software\ Microsoft\WindowsNT\CurrentVersion\Winlogon.**
3. В правой половине окна появится список ключей для данной ветви. Найдите ключ **Userinit** и проверьте его содержимое.



Нормальным значением данного ключа является запись **C:\WINDOWS\system32\userinit.exe**. Наличие в ключе других записей может свидетельствовать о наличии на компьютере вредоносных программ. Для удаления из ключа подозрительной записи щелкните на нем два раза левой кнопкой мыши. После редактирования нажмите ОК.



Далее перейдите в папку, где находится подозрительный файл и удалите его вручную. После этого перезагрузите систему. Если после перезагрузки в ключе удаленное значение не появилось снова – система вылечена.

Как уже говорилось выше, любимым местом вредоносных программ являются разделы реестра, отвечающие за автозапуск.

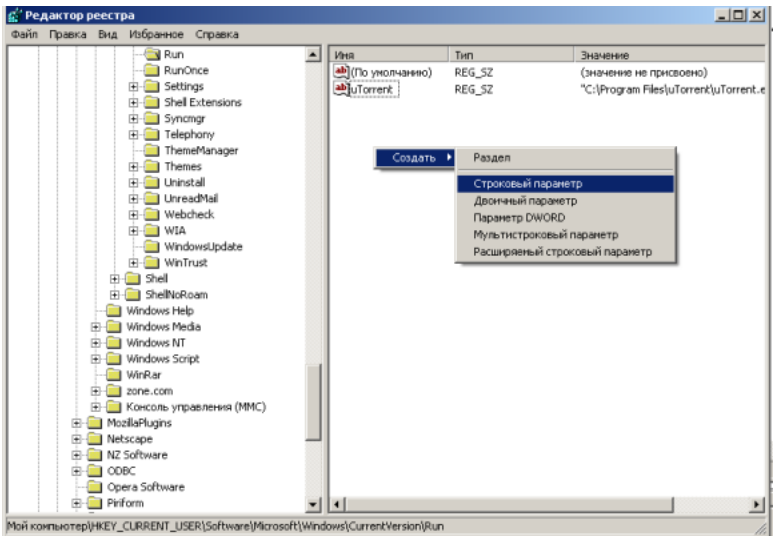
В реестре автозагрузка представлена в нескольких местах:

- **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run** - программы, которые запускаются при входе в систему. Данный раздел отвечает за запуск программ для всех пользователей системы.
- **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce** - программы, которые запускаются только один раз при входе пользователя в систему. После этого ключи программ автоматически удаляются из данного раздела реестра. Данный раздел отвечает за запуск программ для всех пользователей системы.
- **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx** - программы, которые запускаются только один раз, когда загружается система. Этот раздел используется при инсталляции программ, например для запуска настроечных модулей. После этого ключи программ автоматически удаляются из данного раздела реестра. Данный раздел отвечает за запуск программ для всех пользователей системы.
- **HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run** - программы, которые запускаются при входе текущего пользователя в систему
- **HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce** - программы, которые запускаются только один раз при входе текущего пользователя в систему. После этого ключи программ автоматически удаляются из данного раздела реестра.
- **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices**- программы, которые загружаются при старте системы до входа пользователя в Windows.
- **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce** - программы отсюда загружаются только один раз, когда загружается система.

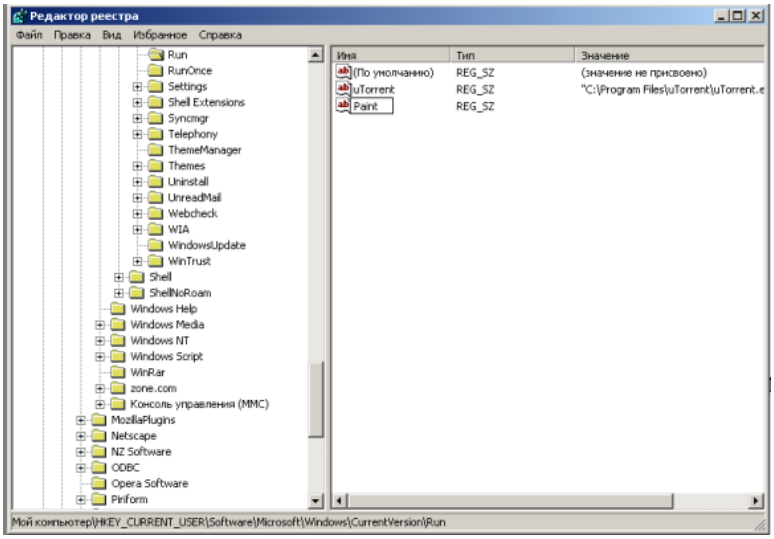
1. Откройте Редактор реестра
2. Откройте следующий раздел

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run

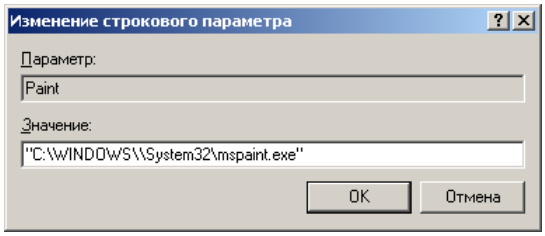
3. В правой части окна вызовите правой кнопкой мыши контекстное меню и выберите Создать->Строковый параметр



4. Задайте имя для создаваемого параметра.



5. Откройте вновь созданный параметр и задайте ему значение в виде пути к исполняемому файлу Paint - **C:\WINDOWS\\System32\mspaint.exe**



6. Нажмите ОК и перезагрузите компьютер. Paint загрузится автоматически после загрузки операционной системы.

Задание 2

1. Выполнить резервное копирование файлов системного реестра.
2. Изучить функции редактора реестра *Registry Editor*:

− создать новый ключ в разделе *Hkey_Current_config*, создать для него параметр строкового типа и задать его значение –"Мой"; какой параметр для вновь созданного ключа появляется по умолчанию?

В отчете указать иерархию ключа, названия и значения созданного параметра и параметра по умолчанию.

− удалить созданные ключ и параметр;
− найти первых два ключа с полным именем «Setup»;
− в отчете указать иерархию ключа
− проверить, имеет ли реестр ключ со значением любого его параметра 35;

отразить результата поиска в отчете;

− проверить возможность экспортировать реестр в новый файл и импортировать его из ранее сохраненного файла; какое расширение имеют файлы импорта-экспорта реестра?

3. Исследование раздела Hkey_Clauses_Root.
− Найти ссылку на подключ для файлов с расширением DOC.
− Найти подключ, на который указывает эта ссылка.
− Для найденного подключа определить следующие ключи настройки Word: вид графического значка (icon); командная строка для запуска исполняемого файла.
− Отредактировать значения параметра "По умолчанию" для этих двух ключей таким образом, чтобы изменился графический значок Word, а также изменилось приложение, которое автоматически запускается при открытии файлов с расширением Doc.
− Проверить выполненные установки, открыв любой файл Doc.

В отчете указать иерархию двух ключей, название исследуемых параметров, их новое и старое значения.

4. Исследование раздела *Hkey_Local_Machine*.
− Найти подключи конфигурации оборудования. Сколько конфигураций имеет данный компьютер?

В отчете указать иерархию ключа, название исследуемого параметра, его значения.

5. Исследование раздела Hkey_Current_config.
− Копией какого ключа является данный раздел?
− Найти ключ, отвечающий за настройки дисплея.
− Ознакомиться со списком параметров этого ключа.
− Изменить текущую разрешающую способность монитора на значение "640,480".
− Для проверки выполнения перегрузить операционную систему (ОС).

В отчете указать иерархию ключа, название исследуемого параметра, его новое и старое значения.

6. Исследование раздела *Hkey_Current_user*.
− Копией какого ключа является данный раздел?
− Найти ключ, отвечающий за настройки Рабочего стола. Ознакомиться со списком вложенных ключей. Для произвольно выбранных из списка 5 ключей исследовать, аналогом каких настроек Панели управления они являются.

В отчете указать иерархию пяти ключей и соответствующие настройки Панели управления.

− Изменить с помощью реестра ширину полосы прокрутки и строки командного меню в окнах Windows. Проверить выполненные настройки.

В отчете указать иерархию ключа, название исследуемого параметра, его новое и старое значения.

− В подразделе установленного программного обеспечения для текущего пользователя найти ключ, хранящий полное имя файла справки Word.

В отчете указать иерархию ключа.

− Для приложения Word найти ключ, хранящий информацию о каталоге автоматически сохраняемых документов. Сравнить его с каталогом, указанным в параметрах Word. (Запустить Word, вызвать Сервис-Параметры - Расположение-Автосохраненные).

Иерархию ключа и результаты сравнения отразить в отчете.

Проверить влияние изменения параметров приложения Word через меню Сервис на значение параметра автосохранения ключа в реестре, а также обратную связь. *В отчете указать иерархию ключа, название исследуемого параметра.*

- Для приложения Excel найти ключ, хранящий информацию о последних 9 загруженных файлах XLS. Запомнить названия параметров и значение одного из них. Информацию о ключе, параметре и его значении отразить в отчете.

7. Восстановить состояние системного реестра из резервных копий или из копий ОС.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №6-7:

1. Что такое системный реестр?
2. Где находится системный реестр?
3. Из каких файлов состоит реестр? Где они расположены?
4. Назначение реестра;
5. Способы редактирования реестра;
6. Структура реестра;
7. Структура основного раздела Hkey_Clauses_Root;
8. Основные разделы и их назначение;
9. Параметры ключей. Типы параметров и их значения;
10. Назовите ключи, имеющие псевдонимы;
11. Способы восстановления реестра;
12. Описать виды ключей системного реестра и их значения?
13. Дать описание основных разделов системного реестра;
14. В каких файлах хранится информация о реестре и где они находятся?
15. Поясните особенности «троянских программ».
16. Почему профилактика «троянских программ» связана с системным реестром?
17. Какие разделы и ключи являются потенциальными местами записей «троянских программ»?

Лабораторная работа №8
«Процессы в ОС Linux. Анализ производительности»

Цель работы:

1. Ознакомиться на практике с понятием процесса в операционной системе.
2. Приобрести опыт и навыки управления процессами в операционной системе Linux.

1 Основные понятия и термины

Основными активными сущностями в системе Linux являются процессы. Каждый процесс выполняет одну программу и изначально получает один поток управления. Иначе говоря, у процесса есть один счетчик команд, который отслеживает следующую исполняемую команду. Linux позволяет процессу создавать дополнительные потоки (после того, как он начинает выполнение).

Когда дело доходит до оптимизации производительности системы Linux оперативная память - один из самых важных факторов на которые нужно обратить внимание. В Linux есть множество утилит для контроля использования такого драгоценного ресурса, как физическая память. Инструменты отличаются друг от друга детализацией мониторинга (например для системы в целом, отдельного процесса или отдельного пользователя) интерфейсом (консольный интерфейс или графический) и режимом работы (интерактивный или пассивный режим)

Ниже представлен небольшой список консольных и GUI утилит для проверки количества свободной и используемой оперативной памяти для платформы Linux.

1. /proc/meminfo

Простейший способ проверить использование оперативной памяти - посмотреть /proc/meminfo. Это автоматически обновляемый файл, который является источником для таких утилит как free, ps и top. Кроме количества свободной и использованной памяти в /proc/meminfo есть все что можно узнать о памяти. Информация о памяти для конкретного процесса находится по адресу /proc/pid/statm или /proc/pid/status.

```
$ cat /proc/meminfo
```

```
$ cat /proc/meminfo
MemTotal:        8035340 kB
MemFree:         1259304 kB
Buffers:           68792 kB
Cached:          2303044 kB
SwapCached:        62796 kB
Active:          4836092 kB
Inactive:        1593468 kB
Active(anon):    3329544 kB
Inactive(anon):  1103844 kB
Active(file):    1506548 kB
Inactive(file):   489624 kB
Unevictable:        5232 kB
Mlocked:            5232 kB
SwapTotal:       8245244 kB
SwapFree:        7389536 kB
Dirty:                 0 kB
Writeback:             0 kB
AnonPages:       4030828 kB
Mapped:           309640 kB
Shmem:            375664 kB
Slab:             130608 kB
SReclaimable:      69052 kB
SUnreclaim:        61556 kB
KernelStack:        8808 kB
PageTables:        90264 kB
```

2. atop

atop - основанный на ncurses интерактивный монитор ресурсов. Он показывает в динамическом режиме такие системные параметры: CPU, память, нагрузка на сеть, ввод/вывод, ядро. Также есть подсветка цветом при перегрузке системы. Можно посмотреть топ процессов или пользователей по использованию ими ресурсов, таким образом системный администратор может понять какие процессы или пользователи перегружают систему. Из памяти можно отобразить свободную, кэш, буферизированную и виртуальную.

$ sudo atop

```
ATOP - caillous          2015/01/02  00:43:40          ------          10s elapsed
PRC |  sys    0.74s  |  user   3.96s  |  #proc    331  |  #exit      6  |
CPU |  sys       7%  |  user     38%  |  idle     753%  |  wait      2%  |
cpu |  sys       1%  |  user      7%  |  idle      91%  |  cpu002 w  0%  |
cpu |  sys       1%  |  user      7%  |  idle      91%  |  cpu004 w  1%  |
cpu |  sys       1%  |  user      6%  |  idle      92%  |  cpu000 w  0%  |
cpu |  sys       1%  |  user      7%  |  idle      92%  |  cpu006 w  0%  |
cpu |  sys       2%  |  user      3%  |  idle      95%  |  cpu003 w  0%  |
cpu |  sys       1%  |  user      3%  |  idle      96%  |  cpu007 w  0%  |
cpu |  sys       0%  |  user      2%  |  idle      97%  |  cpu005 w  0%  |
cpu |  sys       0%  |  user      2%  |  idle      97%  |  cpu001 w  0%  |
CPL |  avg1    0.64  |  avg5    0.53  |  csw     31958  |  intr     5740  |
MEM |  free    2.0G  |  cache   1.6G  |  buff  117.7M  |  slab  143.4M  |
SWP |  tot     7.9G  |  free    7.1G  |  vmcom  12.4G  |  vmlim  11.7G  |
LVM |   ubuntu-root  |  busy      1%  |  read       0  |  write     35  |
DSK |           sda  |  busy      1%  |  read       0  |  write     30  |
NET |  transport     |  tcpo      2   |  udpi       2  |  udpo       0  |
NET |  network       |  ipo       3   |  ipfrw      0  |  deliv      6  |
NET |  eth0      0%  |  pcki     10   |  pcko       3  |  so    0 Kbps  |
NET |  wlan0    ----  |  pcki     11   |  pcko       0  |  so    0 Kbps  |

  PID MINFLT MAJFLT  VSTEXT  VSIZE   RSIZE  VGROW   RGROW  MEM CMD          1/7
 4026     33      0  81904K   2.1G  797.8M     0K      0K  10% chrome
 3562      0      0    1349K   1.7G  265.9M     0K      0K   3% nautilus
 3936     72      0  81904K   1.2G  263.9M     0K   -764K   3% chrome
 4107      2      0  81904K 997.8M  255.3M     0K      0K   3% chrome
22259   1818      0      4K   1.5G  175.1M  3180K   3240K   2% shutter
12329      6      0  81904K 806.7M  117.3M     0K      0K   1% chrome
 1661   5062      0   2215K 575.4M  105.5M 19300K  17340K   1% Xorg
 4564      0      0  81904K 846.3M  101.2M     0K      0K   1% chrome
11176     63      0  81904K 771.5M  96404K  1024K     68K   1% chrome
 4066     10      0  81904K 753.9M  89476K     0K      0K   1% chrome
 4332     51      0  81904K 812.9M  88456K     0K   -228K   1% chrome
```

3. free

free - это быстрый и простой способ получить информацию о использовании памяти из /proc/meminfo. Будет показано общее количество и количество свободной физической памяти и файла подкачки, а также буфера ядра.

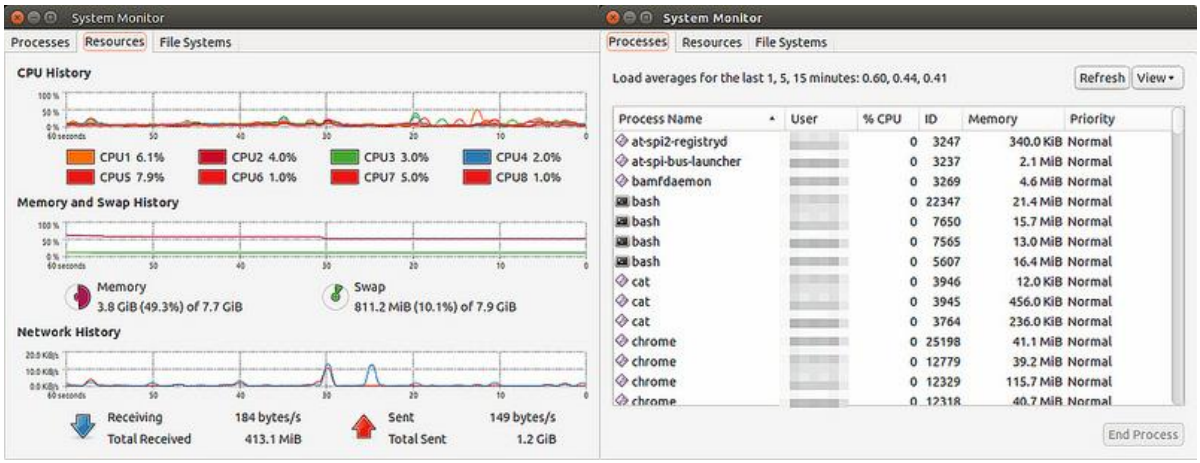
$ free -h

```
$ free -h
             total       used       free     shared    buffers     cached
Mem:          7.7G       6.3G       1.3G       379M        50M       2.2G
-/+ buffers/cache:       4.1G       3.6G
Swap:         7.9G       836M       7.0G
$
```

4. GNOME System Monitor

GNOME System Monitor - это приложение с графическим интерфейсом, которое показывает краткую статистику использования системных ресурсов - памяти, процессора, подкачки и сети. Также есть информация по использованию процессора и памяти для каждого процесса.

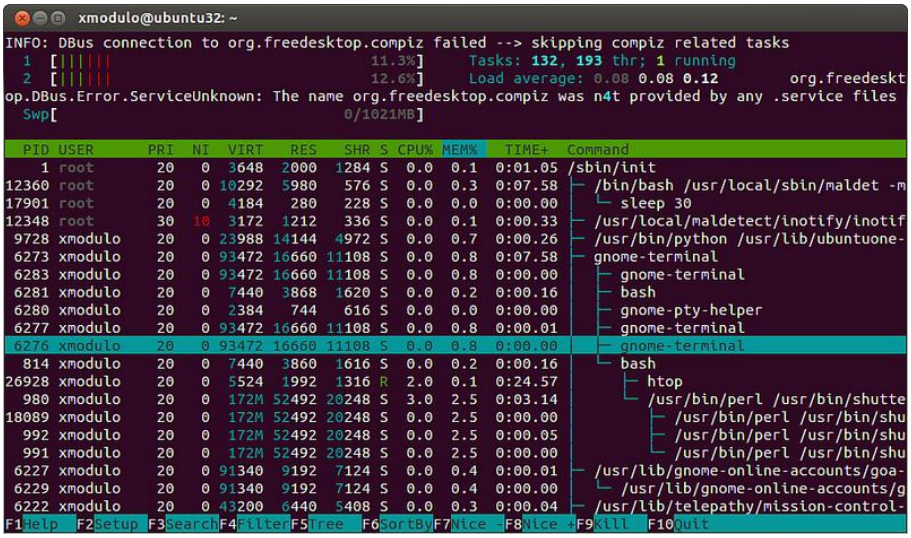
$ gnome-system-monitor



5. htop

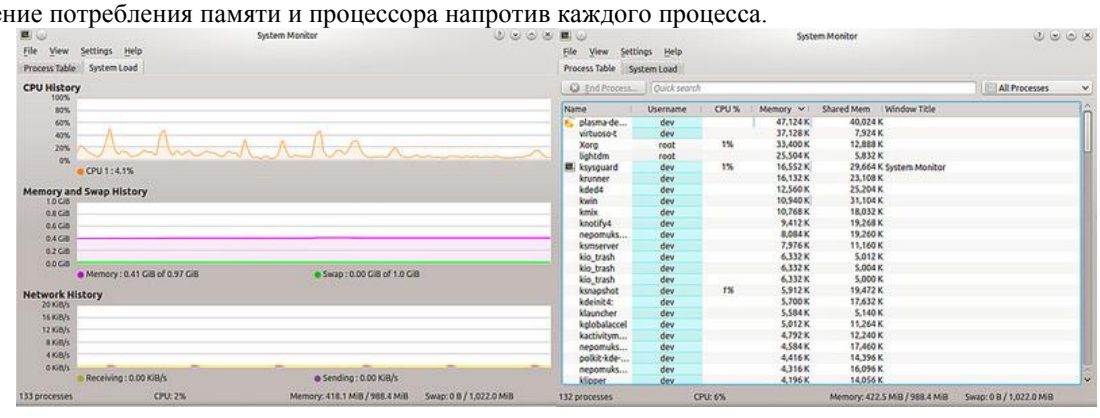
Htop - основанный на ncurses монитор процессов, который показывает использование процессора и памяти для каждого процесса по отдельности в реальном времени. Здесь можно посмотреть количество резидентной памяти, общий объем памяти для программы, размер библиотек, размер страниц, и количество памяти для всех запущенных процессов. Вы можете прокручивать список процессов по горизонтали и по вертикали.

$ htop



6. KDE System Monitor

Как и у Gnome у KDE есть свое решение для мониторинга ресурсов. KDE System Monitor очень похож по функционалу на свой аналог в Gnome. Есть просмотр статистики использования системных ресурсов, а также отображение потребления памяти и процессора напротив каждого процесса.



7. memstat

Утилита memstat полезна для определения какой исполняемый файл, процесс или библиотека использует много оперативной памяти. Утилита определяет количество использованных ресурсов по pid процесса.

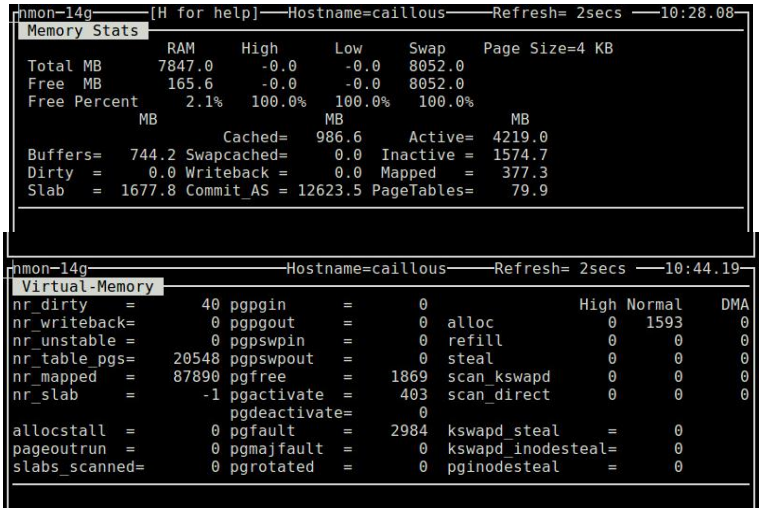
$ memstat -p pid

```
$ memstat -p 3323
   368k: PID  3323 (/usr/bin/gpg-agent)
  2116k(     64k): /usr/lib/x86_64-linux-gnu/libassuan.so.0.4.1 3323
  2124k(     68k): /usr/lib/x86_64-linux-gnu/libpth.so.20.0.27 3323
   296k(    284k): /usr/bin/gpg-agent 3323
  5288k(      0k): /usr/lib/locale/locale-archive 3323
   148k(    140k): /lib/x86_64-linux-gnu/ld-2.19.so 3323
  3844k(   1776k): /lib/x86_64-linux-gnu/libc-2.19.so 3323
  2068k(     16k): /lib/x86_64-linux-gnu/libgpg-error.so.0.10.0 3323
  2556k(    492k): /lib/x86_64-linux-gnu/libgcrypt.so.11.8.2 3323
--------
 18808k (   2840k)
$
```

8. nmon

Nmon - ncurses утилита для тестирования системы. Можно тестировать центральный процессор, память, диск, ввод\вывод, ядро, файловую систему и сетевые ресурсы в интерактивном режиме. Что касается памяти то можно посмотреть общее количество, количество свободной памяти, размер раздела подкачки, кэширование и виртуальную память. И все это в реальном времени.

$ nmon



9. ps

Команда ps может показать использование памяти для каждого процесса в реальном времени. Показывается процент используемой памяти (MEM), общее количество виртуальной памяти (VSZ) общий объем физической памяти (RSS) Вы также можете отсортировать список процессов с помощью опции --sort. Например, для сортировки по убыванию rss используйте:

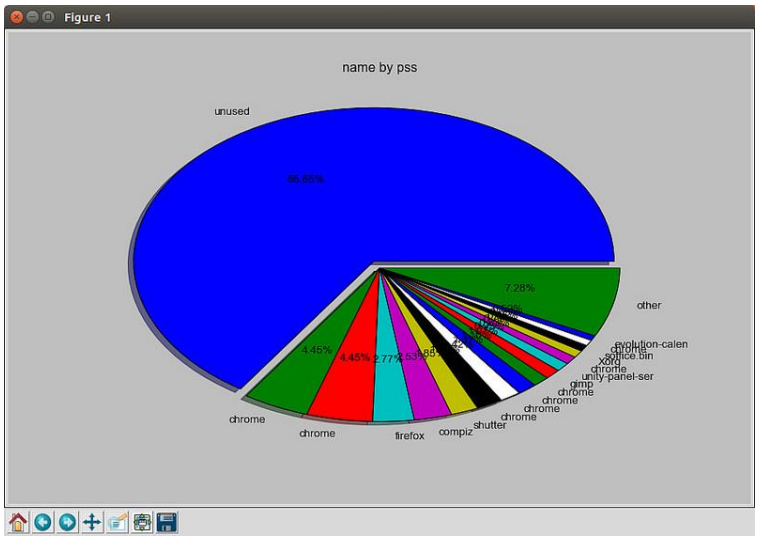
$ ps aux --sort –rss

```
$ ps aux --sort -rss
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
          4111  1.3 10.4 2086956 836880 ?      Sl   09:07   1:51 /opt/google/chrome/ch
          3945  0.3  3.4 1066832 279092 ?      Sl   09:07   0:32 /usr/lib/firefox/fire
         12215  3.3  3.2 1052920 264376 ?      Sl   11:14   0:28 /opt/google/chrome/ch
          3934  4.4  2.8 1149776 231168 ?      SLl  09:07   6:21 /opt/google/chrome/ch
          4195  1.0  2.4 918804 196100 ?       Sl   09:07   1:26 /opt/google/chrome/ch
         12567  2.8  2.4 938028 194020 ?       Sl   11:18   0:17 /opt/google/chrome/ch
          4968  0.5  1.8 831972 147372 ?       Sl   09:09   0:42 /opt/google/chrome/ch
          4173  0.6  1.7 775952 137816 ?       Sl   09:07   0:54 /opt/google/chrome/ch
          9794  0.6  1.6 1200064 134344 ?      Sl   10:27   0:22 /usr/bin/perl /usr/bi
          4859  0.1  1.6 857032 128596 ?       Sl   09:09   0:14 /opt/google/chrome/ch
          4701  0.4  1.5 843756 122516 ?       Sl   09:08   0:40 /opt/google/chrome/ch
          4491  0.5  1.5 812940 121756 ?       Sl   09:08   0:43 /opt/google/chrome/ch
          7429  0.6  1.3 828416 107356 ?       Sl   09:54   0:35 /opt/google/chrome/ch
          4647  0.0  1.2 796024 97736 ?        Sl   09:08   0:04 /opt/google/chrome/ch
          4691  0.0  1.2 782888 97612 ?        Sl   09:08   0:02 /opt/google/chrome/ch
          4411  0.4  1.1 792084 93548 ?        Sl   09:07   0:37 /opt/google/chrome/ch
          4795  0.4  1.1 793720 88532 ?        Sl   09:08   0:37 /opt/google/chrome/ch
```

10. smem

Команда smem позволяет измерить количество памяти используемое различными процессами и пользователями на основе информации взятой из /proc. Она показывает количество ресурсов в процентном соотношении. Данные могут быть экспортированы в виде диаграмм, таких как круговые графики:

$ sudo smem --pie name -c "pss"



11. top

Команда top выводит список запущенных процессов в режиме реального времени, а также различные статистические данные для каждого из них. Вы можете сортировать список процессов по использованию памяти.

```
top - 00:29:25 up 13:41,  5 users,  load average: 0.12, 0.42, 0.40
Tasks: 325 total,   1 running, 324 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1.9 us,  0.5 sy,  0.0 ni, 97.3 id,  0.3 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:   8035340 total,  5626312 used,  2409028 free,   104296 buffers
KiB Swap:  8245244 total,   826624 used,  7418620 free.  1585436 cached Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 4026 [redacted] 20   0 2240612 808796  92032 S   0.3 10.1  10:09.05 chrome
 3562 [redacted] 20   0 1764436 272280  42552 S   0.0  3.4   5:09.71 nautilus
 3936 [redacted] 20   0 1244144 268332  51192 S   1.0  3.3  25:44.46 chrome
 4107 [redacted] 20   0 1023796 261816  16856 S   1.3  3.3  13:12.13 chrome
22259 [redacted] 20   0 1581220 175796  19212 S   0.3  2.2   1:28.09 shutter
 1661 [redacted] 20   0  579720 121596 102184 S   2.3  1.5  17:34.12 Xorg
12329 [redacted] 20   0  826044 119120  21312 S   0.3  1.5   0:42.00 chrome
 4564 [redacted] 20   0  865632 103716  10844 S   0.0  1.3   1:03.93 chrome
 4066 [redacted] 20   0  773060  93192  10432 S   0.0  1.2   2:10.70 chrome
 4102 [redacted] 20   0  832364  88608  17080 S   1.0  1.1   6:17.39 chrome
 4775 [redacted] 20   0  788472  86352  15576 S   0.0  1.1   0:42.50 chrome
 4332 [redacted] 20   0  831412  85544  10992 S   1.0  1.1   6:15.19 chrome
 3234 [redacted] 20   0  554536  82340   8540 S   0.0  1.0   0:42.73 hud-service
 4271 [redacted] 20   0  795312  68200  11252 S   2.3  0.8   5:53.70 chrome
 4511 [redacted] 20   0  801912  67148  11024 S   1.0  0.8   5:48.59 chrome
 4671 [redacted] 20   0  786864  66252  12456 S   0.3  0.8   0:19.13 chrome
10825 [redacted] 20   0  752028  65348  29184 S   0.3  0.8   0:01.46 chrome
 3204 [redacted] 20   0 1293416  64868   3400 S   0.0  0.8   0:14.09 mediascann+
 4400 [redacted] 20   0  791916  62104  10852 S   0.0  0.8   0:49.11 chrome
```

12. vmstat

Утилита командной строки vmstat отображает статистические данные по использованию CPU, памяти, прерываний и ввода вывода на диск. Команда показывает не только физическую память (всего, использовано, кэшировано, буферизировано), но и статистику по виртуальной памяти (количество страниц в подкачке и т д)

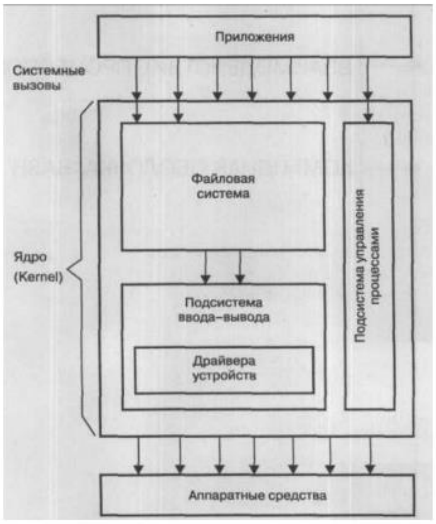
$ vmstat –s

```
$ vmstat -s
      8035340 K total memory
      6926892 K used memory
      4958512 K active memory
      1619960 K inactive memory
      1108448 K free memory
        70848 K buffer memory
      2337564 K swap cache
      8245244 K total swap
       855456 K used swap
      7389788 K free swap
       919943 non-nice user cpu ticks
        37072 nice user cpu ticks
       358405 system cpu ticks
     30461521 idle cpu ticks
       341852 IO-wait cpu ticks
        14037 IRQ cpu ticks
            0 softirq cpu ticks
            0 stolen cpu ticks
     19165190 pages paged in
      5202672 pages paged out
        52389 pages swapped in
       226781 pages swapped out
     41370113 interrupts
    138735451 CPU context switches
   1420127290 boot time
        36139 forks
```

1.1 Устройство Linux: ядро и процессы

Оперативной памяти, часть ОС Linux называется ядром (Kernel). Ядро ОС обрабатывает прерывания от устройств, выполняет запросы системных процессов и пользовательских приложений, распределяет виртуальную память, создает и уничтожает процессы, обеспечивает многозадачность посредством переключения между ними, содержит драйверы устройств, обслуживает файловую систему.

Пользовательские процессы не могут непосредственно, например, порождать другие процессы, производить чтение или запись на диск, выводить данные на экран или создавать гнездо (socket) для обмена по сети. Для выполнения этих действий они должны воспользоваться сервисами ядра. Обращения за такими услугами называются системными вызовами.



Ядро обслуживает запросы процессов. Процесс можно представить себе, как виртуальную машину, отданную в распоряжение одной задачи. Каждый процесс считает, что он на машине один и может распоряжаться всеми ее ресурсами. На самом же деле процессы надежно изолированы друг от друга, так что крушение одного не может повредить всей системе

Каждый процесс выполняется в собственной виртуальной памяти, в которую никакой другой процесс вмешаться не может. Этим и обеспечивается устойчивость всей системы.

Каждый процесс в ОС Linux характеризуется набором атрибутов, который отличает данный процесс от всех остальных процессов. К таким атрибутам относятся:

Идентификатор процесса (PID). Каждый процесс в системе имеет уникальный идентификатор. Каждый новый запущенный процесс получает номер на единицу больше предыдущего.

Идентификатор родительского процесса (PPID). Данный атрибут процесс получает во время своего запуска и используется для получения статуса родительского процесса.

Реальный и эффективный идентификаторы пользователя (UID,EUID) и группы (GID, EGID). Данные атрибуты процесса говорят о его принадлежности к конкретному пользователю и группе. Реальные идентификаторы совпадают с идентификаторами пользователя, который запустил процесс, и группы, к которой он принадлежит.

Эффективные - от чьего имени был запущен процесс. Права доступа процесса к ресурсам ОС Linux эффективными идентификаторами. Если на исполняемом файле программы установлен специальный бит SGID или SUID, то процесс данной программы будет обладать правами доступа владельца исполняемого файла.

Для управления процессом (например, kill) используются реальные идентификаторы. Все идентификаторы передаются от родительского процесса к дочернему. Для просмотра данных атрибутов можно воспользоваться командой ps, задав желаемый формат отображения колонок.

Приоритет или динамический приоритет (priority) и относительный или статический (nice) приоритет процесса.

Статический приоритет или nice-приоритет лежит в диапазоне от -20 до 19, по умолчанию используется значение 0. Значение –20 соответствует наиболее высокому приоритету, nice-приоритет не изменяется планировщиком, он наследуется от родителя или его указывает пользователь.

Динамический приоритет используется планировщиком для планирования выполнения процессов. Этот приоритет хранится в поле prio структуры task_struct процесса. Динамический приоритет вычисляется исходя из значения параметра nice для данной задачи путем вычисления надбавки или штрафа, в зависимости от интерактивности задачи.

Пользователь имеет возможность изменять только статический приоритет процесса. При этом повышать приоритет может только root.

В ОС Linux существуют две команды управления приоритетом процессов: nice и renice.

Состояние процесса. В ОС Linux каждый процесс обязательно находится в одном из перечисленных ниже состояний и может быть переведен из одного состояния в другое системой или командами пользователя.

Различают следующее состояния процессов:

- TASK_RUNNING — процесс готов к выполнению или выполняется (runnable). Обозначается символом R.
- TASK_INTERRUPTIBLE - ожидающий процесс (sleeping). Данное состояние означает, что процесс инициализировал выполнение какой-либо системной операции и ожидает ее завершения. К таким операциям относятся ввод/вывод, завершение дочернего процесса и т.д. Процессы с таким состоянием обозначаются символом S.
- TASK_STOPPED - выполнение процесса остановлено (stopping). Любой процесс можно остановить. Это может делать как система, так и пользователь. Состояние такого процесса обозначается символом T.
- TASK_ZOMBIE - завершившийся процесс (zombie). Процессы данного состояния возникают в случае, когда родительский процесс не ожидая завершения дочернего процесса, продолжает параллельно работать. Процессы с таким состоянием обозначаются символом Z.

Завершившиеся процессы больше не выполняются системой, но по-прежнему продолжают потреблять ее не вычислительные ресурсы.

- TASK_UNINTERRUPTIBLE -непрерываемый процесс (uninterruptible). Процессы в данном состоянии ожидают завершения операции ввода - вывода с прямым доступом в память. Такой процесс нельзя завершить, пока не завершится операция ввода/вывода.

Процессы с таким состоянием обозначаются символом D. Состояние аналогично TASK_INTERRUPTIBLE, за исключением того, что процесс не возобновляет выполнение при получении сигнала. Используется в случае, когда процесс должен ожидать беспрерывно или когда ожидается, что некоторое событие может возникать достаточно часто. Так как задача в этом состоянии не отвечает на сигналы, TASK_UNINTERRUPTIBLE используется менее часто, чем TASK_INTERRUPTIBLE.

1.2 Команды для управления процессами в Linux

Моментальный снимок протекающих в системе процессов – команда ps Моментальный снимок протекающих в системе процессов можно посмотреть с помощью команды ps (process status). Без аргументов она покажет список процессов, связанных с текущей консолью (или виртуальным терминалом). Список возможных ключей команды можно, как обычно, получить по команде ps --help. Вот некоторые полезные из них:

- -p < список_PID> : только процессы с указанными ID;
- -u < список_USERID > : только процессы, запущенные указанными пользователями;
- -e : все процессы в системе;
- -f : полная форма вывода;
- -H : вывод иерархии процессов в форме дерева.

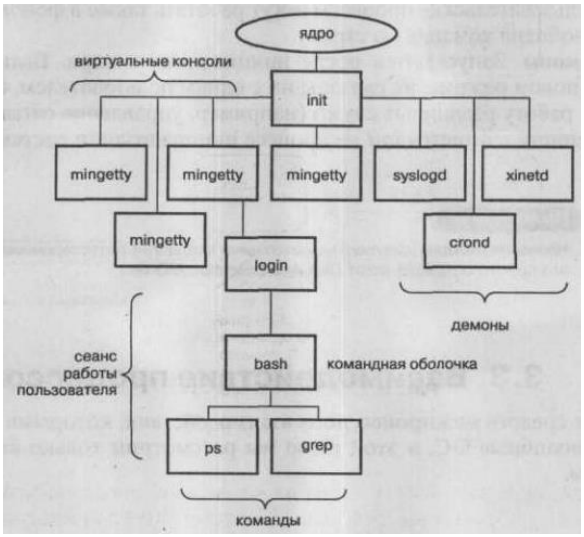
Динамика процессов – команда top Представление о динамике процессов дает команда top. Она выводит список процессов, отсортированный по количеству занятой памяти или использованного процессорного времени, и обновляет его через указанные промежутки времени (по умолчанию через каждые 3 секунды).

Последний процесс, запущенный из оболочки в фоне, можно из этой оболочки сделать активным при помощи команды **fg** ("foreground" – "передний план").

Команда **bg** (back ground), запускает в фоне последний остановленный процесс. Командой kill, как уже говорилось, можно передать процессу сигнал. Команда имеет два параметра - номер сигнала и идентификатор процесса, которому передается сигнал: **kill – номер_сигнала PID**

1.3 Иерархия процессов

В Linux реализована четкая иерархия процессов в системе. Каждый процесс в системе имеет всего одного родителя и может иметь один или более порожденных процессов.



На последней фазе загрузки ядро монтирует корневую файловую систему и формирует среду выполнения нулевого процесса, создавая пространство процесса, инициализируя нулевую точку входа в таблице процесса и делая корневой каталог текущим для процесса. Когда формирование среды выполнения процесса заканчивается, система исполняется уже в виде нулевого процесса. Нулевой процесс «ветвится», запуская fork прямо из ядра, поскольку сам процесс исполняется в режиме ядра. Код, исполняемый порожденным процессом 1, включает в себя вызов системной функции exec, запускающей на выполнение программу из файла "/etc/init".

В отличие от нулевого процесса, который является процессом системного уровня, выполняющимся в режиме ядра, процесс 1 относится к пользовательскому уровню. Обычно процесс 1 именуется процессом init, поскольку он отвечает за инициализацию новых процессов. На самом деле вы можете поместить любую программу в /sbin/init и ядро запустит её как только закончит загружаться. Задачей init'a является запуск всего остального нужным образом

1.3.1 Категории процессов

Процессы делятся на три категории:

- **Системные.** Они порождаются ядром особым образом в процессе загрузки и выполняют системные функции (например, планирование процессов или смену страниц виртуальной памяти). Выполняемая ими программа берется не из исполняемого файла, а является частью ядра.
- **Пользовательские.** Как правило, они порождаются во время сеанса работы пользователя и связаны с терминалом. Если пользовательский процесс работает в интерактивном режиме, то он захватывает терминал в монопольное владение и, пока он не завершится, пользователь не имеет доступа к командной строке на этом терминале. Пользовательские процессы могут работать также в фоновом режиме, освободив командную строку.
- **Демоны** (daemon, сокращение от Disk And Execution MONitor). Запускаются после инициализации ядра. Выполняются в фоновом режиме, не связаны ни с одним пользователем, обеспечивают работу различных служб (например, управление сетью). Главным демоном считается init — процесс инициализации системы.

Задание

1 Загрузиться не root, а пользователем.
2 Найти файл с образом ядра. Выяснить по имени файла номер версии Linux.
3 Посмотреть процессы ps –f. Прокомментировать. Для этого почитать man ps.
4 Написать с помощью редактора vi два сценария loop и loop2. Текст сценариев: Loop:
while true; do true; done
Loop2:
while true; do true; echo ‘Hello’; done
5 Запустить loop2 на переднем плане: sh loop2.
6 Остановить, послав сигнал STOP.
7 Посмотреть последовательно несколько раз ps –f. Записать сообщение, объяснить.
8 Убить процесс loop2, послав сигнал kill -9 PID. Записать сообщение. Прокомментировать.

9 Запустить в фоне процесс loop: sh loop&. Не останавливая, посмотреть несколько раз: ps – f. Записать значение, объяснить.

10 Завершить процесс loop командой kill -15 PID. Записать сообщение, прокомментировать.

11 Третий раз запустить в фоне. Не останавливая убить командой kill -9 PID.

12 Запустить еще один экземпляр оболочки: bash.

13 Запустить несколько процессов в фоне. Останавливать их и снова запускать.

14 Записать результаты просмотра командой ps –f.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8:

1. Какие программы называются файловыми менеджерами?
2. Какая информация отражается в области просмотра программы Konqueror?
3. Как создать новое окно с помощью программы Konqueror?
4. Перечислите задачи по управлению файловой системой, которые можно решать с помощью диспетчера файлов?
5. Перечислите стандартные функции KDE.
6. Что является компонентом рабочего стола KDE?
7. Назовите функции панели рабочего стола.
8. Как получить справку в диалоговом режиме?
9. Какие функции предоставляет центр управления KDE?

Лабораторная работа №9
«Функции ОС по управлению памятью. Анализ производительности»

Цель работы:

1 Практическое знакомство с управлением памятью в операционных системах Windows.

1 Функции ОС по управлению памятью

Под памятью (memory) здесь подразумевается оперативная память компьютера. В отличие от памяти жесткого диска, которую называют внешней памятью (storage), оперативной памяти для сохранения информации требуется постоянное электропитание.

Память является важнейшим ресурсом, требующим тщательного управления со стороны мультипрограммной операционной системы. Особая роль памяти объясняется тем, что процессор может выполнять инструкции протравы только в том случае, если они находятся в памяти. Память распределяется как между модулями прикладных программ, так и между модулями самой операционной системы.

В ранних ОС управление памятью сводилось просто к загрузке программы и ее данных из некоторого внешнего накопителя (перфоленты, магнитной ленты или магнитного диска) в память. С появлением мультипрограммирования перед ОС были поставлены новые задачи, связанные с распределением имеющейся памяти между несколькими одновременно выполняющимися программами.

Функциями ОС по управлению памятью в мультипрограммной системе являются:

- отслеживание свободной и занятой памяти;
- выделение памяти процессам и освобождение памяти по завершении процессов;
- вытеснение кодов и данных процессов из оперативной памяти на диск (полное или частичное), когда размеры основной памяти не достаточны для размещения в ней всех процессов, и возвращение их в оперативную память, когда в ней освобождается место;
- настройка адресов программы на конкретную область физической памяти.

Помимо первоначального выделения памяти процессам при их создании ОС должна также заниматься динамическим распределением памяти, то есть выполнять запросы приложений на выделение им дополнительной памяти во время выполнения. После того как приложение перестает нуждаться в дополнительной памяти, оно может возвратить ее системе. Выделение памяти случайной длины в случайные моменты времени из общего пула памяти приводит к фрагментации и, вследствие этого, к неэффективному ее использованию. Дефрагментация памяти тоже является функцией операционной системы.

Во время работы операционной системы ей часто приходится создавать новые служебные информационные структуры, такие как описатели процессов и потоков, различные таблицы распределения ресурсов, буферы, используемые процессами для обмена данными, синхронизирующие объекты и т. п. Все эти системные объекты требуют памяти»» В некоторых ОС заранее (во время установки) резервируется некоторый фиксированный объем памяти для системных нужд. В других же ОС используется более гибкий подход, при котором память для системных целей выделяется динамически. В таком случае разные подсистемы ОС при создании своих таблиц, объектов, структур и т. п. обращаются к подсистеме управления памятью с запросами.

Защита памяти – это еще одна важная задача операционной системы, которая состоит в том, чтобы не позволить выполняемому процессу записывать или читать данные из памяти, назначенной другому процессу. Эта функция, как правило, реализуется программными модулями ОС в тесном взаимодействии с аппаратными средствами.

Виртуализация оперативной памяти осуществляется совокупностью аппаратных и программных средств вычислительной системы (схемами процессора и операционной системой) автоматически без участия программиста и не сказывается на логике работы приложения.

Виртуализация памяти возможна на основе двух возможных подходов:

- свопинг (swapping) – образы процессов выгружаются на диск и возвращаются в оперативную память целиком;
- виртуальная память (*virtual memory*) – между оперативной памятью и диском перемещаются части образов (сегменты, страницы, блоки и т.п.) процессов.

Недостатки свопинга:

- избыточность перемещаемых данных и отсюда замедление работы системы и неэффективное использование памяти;
- невозможность загрузить процесс, виртуальное пространство которого превышает имеющуюся в наличии свободную память.

Достоинство свопинга *по* сравнению с виртуальной памятью – меньшие *затраты* времени на преобразование адресов в кодах программ, поскольку оно делается один раз при загрузке с диска в *память* (однако это преимущество может быть незначительным, т.к. выполняется при очередной загрузке только часть кода и полностью преобразовывать код, может быть, и не надо).

Виртуальная память не имеет указанных недостатков, но ее ключевой проблемой является преобразование виртуальных адресов в физические (почему это проблема, будет ясно дальше, а пока можно отметить существенные *затраты* времени на этот процесс, если не принять специальных мер).

Также управлять процессами можно и «вручную» при помощи командной строки.

Команды Windows для работы с процессами:

- at — запуск программ в заданное время
- Schtasks — настраивает выполнение команд по расписанию
- Start — запускает определенную программу или команду в отдельном окне.
- Taskkill — завершает процесс
- Tasklist — выводит информацию о работающих процессах

Для получения более подробной информации, можно использовать центр справки и поддержки или команду help (например: help at)

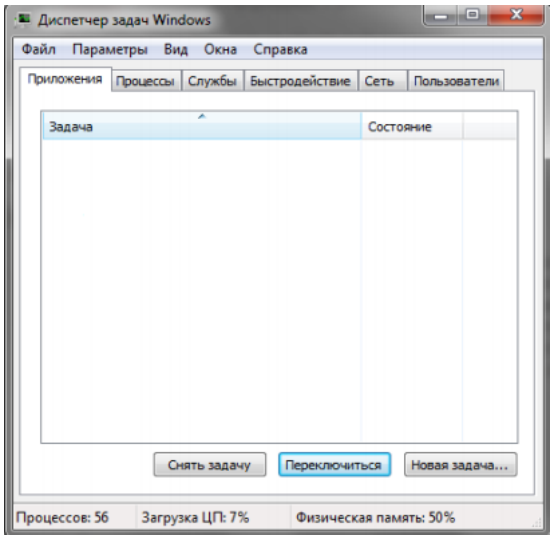
- command.com — запуск командной оболочки MS-DOS
- cmd.exe — запуск командной оболочки Windows. Ход работы:

Задание 1. Работа с Диспетчером задач Windows 7.

1. Запустите Windows 7
2. Запуск диспетчера задач можно осуществить двумя способами:

- Нажатием сочетания клавиш Ctrl+Alt+Del. При использовании данной команды не стоит пренебрегать последовательностью клавиш. Появится меню, в котором курсором следует выбрать пункт «Диспетчер задач».
- Переведите курсор на область с показаниями системной даты и времени и нажмите правый клик, будет выведено меню, в котором следует выбрать «Диспетчер задач».

3. Будет выведено окно.



4. В диспетчере задач есть 6 вкладок:
   I. Приложения
   II. Процессы
   III. Службы
   IV. Быстродействие

V. Сеть
VI. Пользователи

− Вкладка «Приложения» отображает список запущенных задач (программ) выполняющиеся в настоящий момент не в фоновом режиме, а также отображает их состояние. Также в данном окне можно снять задачу переключиться между задачами и запустить новую задачу при помощи соответствующих кнопок.

− Вкладка «Процессы» отображает список запущенных процессов, имя пользователя запустившего процесс, загрузку центрального процессора в процентном соотношении, а также объем памяти используемого для выполнения процесса. Также присутствует возможность отображать процессы всех пользователей, либо принудительного завершения процесса. Процесс — выполнение пассивных инструкций компьютерной программы на процессоре ЭВМ.

− Вкладка «Службы» показывает, какие службы запущены на компьютере. Службы

− — приложения, автоматически запускаемые системой при запуске ОС Windows и выполняющиеся вне зависимости от статуса пользователя.

− Вкладка «Быстродействие» отображает в графическом режиме загрузку процессора, а также хронологию использования физической памяти компьютера. Очень эффективным инструментом наблюдения является «Монитор ресурсов». С его помощью можно наглядно наблюдать за каждой из сторон «жизни» компьютера. Подробное изучение инструмента произвести самостоятельно, интуитивно.

− Вкладка «Сеть» отображает подключенные сетевые адаптеры, а также сетевую активность.

− Вкладка «Пользователи» отображает список подключенных пользователей.

5. После изучения диспетчера задач:

− Потренируйтесь в завершении и повторном запуске процессов.

− Разберите мониторинг загрузки и использование памяти.

− Попытайтесь запустить новые процессы при помощи диспетчера, для этого можно использовать команды: cmd, msconfig.

Задание 2. Командная строка Windows.

1. Для запуска командной строки в режиме Windows следует нажать:

(Пуск) > «Все программы» > «Стандартные» > «Командная строка»

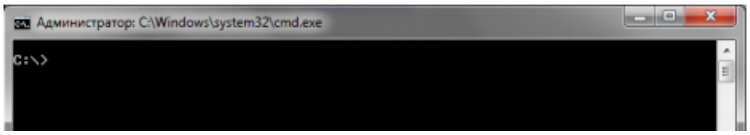
2. Поработайте выполнением основных команд работы с процессами: запуская, отслеживая и завершая процессы.

Основные команды

Schtasks — выводит выполнение команд по расписанию

Start — запускает определенную программу или команду в отдельном окне. Taskkill — завершает процесс

Tasklist — выводит информацию о работающих процессах



3. В появившемся окне наберите:

cd\ — переход в корневой каталог;

cd windows – переход в каталог Windows.

dir — просмотр содержимого каталога.

В данном каталоге мы можем работать с такими программами как «WordPad» и «Блокнот».

4. Запустим программу «Блокнот»:

C:\Windows > start notepad.exe

Отследим выполнение процесса: C:\Windows > tasklist

Затем завершите выполнение процесса: C:\Windows > taskkill /IM notepad.exe

5. Самостоятельно, интуитивно, найдите команду запуска программы WordPad.

Необходимый файл запуска найдите в папке Windows.

6. Выполнение задания включить в отчет по выполнению лабораторной работы.

Задание 3. Самостоятельное задание.

1. Отследите выполнение процесса explorer.exe при помощи диспетчера задач и командной строки.
2. Продемонстрируйте преподавателю завершение и повторный запуск процесса explorer.exe из:

- Диспетчера задач;
- Командной строки.

3. Выполнение задания включить в отчет по выполнению лабораторной работы.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №9:

1. Дайте понятие процессу в операционной системе;
2. Дайте понятие службе в операционной системе;

3. Для чего служит диспетчер задач?
4. Причислите основные команда работы с процессами при помощи командной строки.

Лабораторная работа №10
«Регистры процессоров. (Ассемблер)»

Цель работы:
1 Изучение структуры учебной ЭВМ и исполнение команд;

1 Основные понятия и термины

Регистры процессора – это внутренняя сверхоперативная память небольшого размера, которая предназначена для временного хранения служебной информации или данных.

Практически все команды так или иначе изменяют значения регистров, и всегда быстрее и удобнее обращаться к регистру, чем к памяти. Количество регистров в разных процессорах может быть от 6-8 до нескольких десятков.

Все регистры можно разделить на 2 группы:
1) *универсальные* (предназначены для хранения различных данных и взаимозаменяемы);
2) *специализированные* (предназначены для хранения только определённых данных и не взаимозаменяемы).

Все внутренние регистры процессора Intel 8086 являются 16-битными:

Всего процессор содержит 12 программно-доступных регистров, а также регистр флагов (FLAGS) и указатель команд (IP).

Регистры общего назначения (РОН) AX, BX, CX и DX используются для хранения данных и выполнения различных арифметических и логических операций. Кроме того, каждый из этих регистров поделён на 2 части по 8-бит, с которыми можно работать как с 8-битными регистрами (AH, AL, BH, BL, CH, CL, DH, DL). Младшие части регистров имеют в названии букву L (от слова *Low*), а старшие H (от слова *High*). Некоторые команды неявно используют определённый регистр, например, CX может выполнять роль счетчика цикла.

Индексные регистры предназначены для хранения индексов при работе с массивами. SI (*Source Index*) содержит индекс источника, а DI (*Destination Index*) — индекс приёмника, хотя их можно использовать и как регистры общего назначения.

Регистры-указатели BP и SP используются для работы со стеком. BP (*Base Pointer*) позволяет работать с переменными в стеке. Его также можно использовать в других целях. SP (*Stack Pointer*) указывает на вершину стека. Он используется командами, которые работают со стеком. (Про стек я подробно расскажу в отдельной части учебного курса)

Сегментные регистры CS (*Code Segment*), **DS** (*Data Segment*), **SS** (*Stack Segment*) и **ES** (*Enhanced Segment*) предназначены для обеспечения сегментной адресации. Код находится в сегменте кода, данные — в сегменте данных, стек — в сегменте стека и есть еще дополнительный сегмент данных. Реальный физический адрес получется путём сдвига содержимого сегментного регистра на 4 бита влево и прибавления к нему смещения (относительного адреса внутри сегмента).

COM-программа всегда находится в одном сегменте, который является одновременно сегментом кода, данных и стека. При запуске COM-программы сегментные регистры будут содержать одинаковые значения.

Указатель команд IP (*Instruction Pointer*) содержит адрес команды (в сегменте кода). Напрямую изменять его содержимое нельзя, но процессор делает это сам. При выполнении обычных команд значение IP увеличивается

на размер выполненной команды. Существуют также команды передачи управления, которые изменяют значение IP для осуществления переходов внутри программы.

Регистр флагов FLAGS содержит отдельные биты: флаги управления и признаки результата. Флаги управления меняют режим работы процессора:

- D (*Direction*) — флаг направления. Управляет направлением обработки строк данных: DF=0 — от младших адресов к старшим, DF=1 — от старших адресов к младшим (для специальных строковых команд).
- I (*Interrupt*) — флаг прерывания. Если значение этого бита равно 1, то прерывания разрешены, иначе — запрещены.
- T (*Trap*) — флаг трассировки. Используется отладчиком для выполнения программы по шагам.

1.1 Команды

Команды языка ассемблера – это символьная форма записи машинных команд. Команды имеют следующий синтаксис:

[<метка>:] <мнемокод> [<операнды>] [;<комментарий>]

Метка – это имя. Метка обязательно должна отделяться двоеточием, но может размещаться отдельно, в строке, предшествующей остальной части команды.

Метки нужны для ссылок на команды из других мест, например, в командах перехода. Компилятор языка ассемблера заменяет метки адресами команд.

Мнемокод – это служебное слово, указывающее операцию, которая должна быть выполнена. Язык ассемблера использует не цифровые коды операций, а мнемокоды, которые легче запоминаются. Мнемокод является обязательной частью команды.

Операнды команды, если они есть, отделяются друг от друга запятыми.

1.1.1. Операнды команд

В качестве операндов команд языка ассемблера могут использоваться:

- регистры, обращение к которым осуществляется по именам;
- непосредственные операнды – константы, записываемые непосредственно в команде;
- ячейки памяти – в команде записывается адрес нужной ячейки.

Пример 1

Начнем изучение языка ассемблера с рассмотрения простой программы, которая выводит на экран строку с текстом. В MS-DOS существует два типа исполняемых файлов, каждый из которых отличается структурой, ограничение на максимальный размер и т.д. Соответственно программы, предназначенные для получения каждого из типов файлов, будут немого различаться, все дело в инициализации основных сегментных регистров. В примере 1.1. будет приведена программа, ориентированная для получения исполняемого файла .EXE.

```
text    segment 'code'           ;(1)Начало сегмента команд
        assume  CS:text, DS:text ;(2)Сопоставление сегментного регистра и сегмента
begin:  mov     AX,text          ;(3)Адрес сегмента команд занесен в регистр AX
        mov     DS,AX            ;(4)А затем в DS
        mov     AH,09h           ;(5)AH=09h номер функции вывода на экран
        mov     DX,offset message;(6)В DX заносится адрес выводимого сообщения
        int     21h              ;(7)Вызов прерывания MS-DOS
        mov     AH,4Ch           ;(8)AH=4Ch номер функции завершения программы
        mov     AL,00h           ;(9)AL=00h, параметр функции 4Ch-код успешного завершения
        int     21h              ;(10)пользовательской программы
message db      'Ассемблер$'     ;(11)Выводимое сообщение
text    ends                     ;(12)Конец сегмента команд
end     begin                    ;(13)Конец программы, с указание точки входа
```

Как видно, в приведенном примере используются как строчные, так и прописные символы, но транслятор программы не различает строки:

mov ds,ax
MOV DS,AX
mov DS,AX

каждая строка воспринимается одинаково.

Программа содержит 13 строк - предложений языка ассемблера. Первое предложение с помощью оператора segment открывает сегмент команд нашей программы. Сегменту дается произвольное имя text. Описатель 'code' (так называемый класс сегмента) говорит о том, что это сегмент команд. В конце предложения после точки с запятой располагается комментарий. Таким образом, предложение языка ассемблера может состоять из четырех полей: имени, оператора, операндов и комментария, располагаемых в перечисленном порядке.

Любая программа должна обязательно состоять из сегментов - без сегментов программ не бывает. Обычно в программе задаются три сегмента: команд, данных и стека, но мы в нашей простой программе пока ограничились одним сегментом команд.

Во втором предложении мы с помощью оператора assume сообщаем ассемблеру (программе-транслятору), что сегментные регистры CS и DS будут указывать на один и тот же сегмент text. Сегментные регистры (а всего их в процессоре четыре) играют очень важную роль.

Когда программа загружается в память и становится известно, по каким адресам памяти она располагается, в сегментные регистры заносятся начальные адреса закрепленных за ними сегментов. В дальнейшем любые обращения к ячейкам программы осуществляются путем указания сегмента, в котором находится интересующая нас ячейка, а также номера того байта внутри сегмента, к которому мы хотим обратиться.

Пример 2

```
text    Segment 'code'                ;(1) Начало сегмента команд
        Assume CS: text, DS: data     ;(2)  CS - на сегмент команд,
                                      ;DS - на сегмент данных
begin:  Mov     AX, data              ;(3) Адрес сегмента данных загрузим
        Mov     DS, AX                ;(4) сначала в AX, затем в DS
        Push    DS                    ;(5) Загрузим в стек содержимое DS
        Pop     ES                    ;(6) Выгрузим его из стека в ES
        mov     AH, 9                 ;(7) Функция DOS вывода на экран
        mov     DX,offset message     ;(8) Адрес выводимого сообщения
        int     21h                   ;(9) Вызов DOS
        mov     AX,4C00h              ;(10) Функция DOS завершения программы
                                      ;с указанием кода завершения 00h
        int     21h                   ;(11) Вызов DOS
text    ends                          ;(12) Конец сегмента команд
data    segment                       ;(13) Начало сегмента данных
message Db 'Знание - сила$ '          ;(14) Выводимый текст
data    ends                          ;(15) Конец сегмента данных
        end     begin                 ;(16) Конец текста программы с указанием  точки
                                      ;входа точки входа
```

В предложении 5 содержимое DS сохраняется в стеке, а в следующем предложении выгружается из стека в ES. После этой операции оба сегментных регистра, и DS, и ES, будут указывать на один и тот же сегмент данных. В нашей программе эти строки не имеют практического смысла, но вообще здесь продемонстрирован удобный прием переноса содержимого одного сегментного регистра в другой. Выше уже отмечалось, что в силу особенностей архитектуры микропроцессора для сегментных регистров действуют некоторые ограничения. Так, в сегментный регистр нельзя непосредственно загрузить адрес сегмента; нельзя также перенести число из одного сегментного регистра в другой. При необходимости выполнить последнюю операцию в качестве «перевалочного пункта « часто используют стек.

Запустите под управлением отладчика программу. Посмотрите, чему равно содержимое регистров SS и SP. Вы увидите, что в SS находится тот же адрес памяти, что и в CS; отсюда можно сделать вывод, что сегменты команд и стека совпадают. Однако содержимое SP равно 0. Первая же команда PUSH уменьшит содержимое SP на 2, т.е. поместит в SP –2. Значит ли это, что стек будет расти, как ему и положено, вверх, но не внутри сегмента команд, а над ним по адресам –2, -4, -6 и т.д. относительно верхней границы сегмента команд?

Оказывается, это не так. Если взять 16 – разрядный счетчик, в котором записан 0, и послать в него два вычитающих импульса, то после первого импульса в нем окажется число FFFFh, а после второго – FFFEh. При желании мы можем рассматривать число FFFEh, как 2 – (что и имеет место при работе со знаковыми числами, о которых будет идти речь позже), однако процессор при вычислении адресов рассматривает содержимое регистров, как целые числа без знака, и число FFFEh оказывается эквивалентным не –2, а 65534. В результате первая же команда занесения данного в стек поместит это данное не над сегментом команд, а в самый его конец, в последнее слово по адресу CS:FFFEh. При дальнейшем использовании стека его указатель будет смещаться в сторону меньших адресов, проходя значения FFFCh. FFFAh и т.д.

Таким образом, если в программе отсутствует явное объявление стека, система сама создает стек по умолчанию в конце сегмента команд. Рассмотренное явление, когда при уменьшении адреса после адреса 0 у нас получится адрес FFFFh, т.е. от начала сегмента мы прыгнули сразу в его конец, носит название циклического возврата или оборачивания адреса. С этим явлением приходится сталкиваться довольно часто. Расположение стека в конце сегмента команд не приводит к каким-либо неприятностям, пока размер программы далек от граничной величины 64К. в этом случае начало сегмента команд занимают коды команд, а конец-стек.

Если, однако, размер программы приближается к 64К, то для стека остается все меньше места. При интенсивном использовании стека в программе может получиться, что по мере занесения в стек новых данных, стек дорастет до последних команд сегмента команд и начнет затирать эти команды. В то же время система не проверяет,

что происходит со стеком и никак не реагирует на затирание команд или данных. Таким образом, оценка размеров собственно программы, данных и стека является важным этапом разработки программы. Современные программы часто имеют значительный размер (даже не помещаясь в один сегмент команд), а стек иногда используется для хранения больших по объему массивов данных. Поэтому целесообразно ввести в программу отдельный сегмент стека, определив его размер, исходя из требований конкретной программы.

1.2 Команды пересылки и обмена

Одна из основных команд языка ассемблер – это команда пересылки. С её помощью можно записать в регистр значение другого регистра, константу или значение ячейки памяти, а также можно записать в ячейку памяти значение регистра или константу. Команда имеет следующий синтаксис:

MOV <операнд1>, <операнд2>

По команде MOV значение второго операнда записывается в первый операнд. Операнды должны иметь одинаковый размер. Команда не меняет флаги.

```
mov   eax, ebx          ; Пересылаем значение регистра EBX в регистр EAX
mov   eax, 0ffffh ; Записываем в регистр EAX шестнадцатеричное значение ffff
mov   x, 0              ; Записываем в переменную x значение 0
mov   eax, x            ; Переслать значение из одной ячейки памяти в другую нельзя.
mov   y, eax            ; Но можно использовать две команды MOV.
```

На самом деле процессор имеет много команд пересылки – код команды зависит от того, куда и откуда пересылаются данные. Но компилятор языка ассемблера сам выбирает нужный код в зависимости от операндов, так что, с точки зрения программиста, команда пересылки только одна.

Для перестановки двух величин используется команда обмена:

XCHG <операнд1>, <операнд2>

Каждый из операндов может быть регистром или ячейкой памяти. Однако переставить содержимое двух регистров можно, а двух ячеек памяти – нет. Операнды должны иметь одинаковый размер. Команда не меняет флаги.

1.3 Оператор указания типа

Как было сказано, операнды команды MOV должны иметь одинаковый размер. В некоторых случаях компилятор может определить размер операнда. Например, регистр EAX имеет размер 32 бита, а регистр DX – 16 бит. Размер переменной определяется по директиве, указанной в её объявлении. Если можно определить размер только одного операнда, то размер второго операнда подгоняется под размер первого, если это возможно. Если же можно определить размеры обоих операндов, то они должны совпадать.

```
x db ?
mov   x, 0              ; 0 может иметь любой размер, в данном случае берётся 1 байт
mov   eax, 0            ; 0 может иметь любой размер, в данном случае берётся 4 байта
mov   al, 1000h   ; Ошибка – попытка записать 2-байтное число в 1-байтный регистр
mov   eax, cx           ; Ошибка – размеры операндов не совпадают
```

Однако не всегда бывает возможно определить размер пересылаемой величины по операндам команды MOV. Например, если один из операндов является ячейкой памяти, адрес которой записан в регистре, то по этому адресу можно записать и 1 байт, и 2 байта, и 4 байта. Если второй операнд является регистром, то размер пересылаемых данных определяется по размеру регистра. Если же второй операнд является константой, то размер пересылаемых данных определить нельзя, и компилятор фиксирует ошибку. Для того чтобы избежать этой ошибки, надо явно указать размер пересылаемых данных. Для этого используется оператор PTR:

<тип> PTR <выражение>

В качестве типа используется BYTE, WORD или DWORD.

```
mov   [ebx], 0                  ; Ошибка, т.к. 0 может иметь любой размер
mov   byte ptr [ebx], 0   ; Пересылаем 1 байт
mov   dword ptr [ebx], 0  ; Пересылаем 4 байта
```

Задание

1. Изучите представленную теорию;
2. Реализуйте приведенные примеры;
3. Ответьте на контрольные вопросы;
4. Реализацию примеров и ответы на вопросы оформите в виде отчета.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №10:

1. Что такое регистр?
2. Предназначение регистра?
3. Какие регистры процессора вы знаете?
4. Расскажите о группах регистров;
5. Что такое ассемблер?
6. Как составляется команда ассемблера?
7. Что может использоваться в качестве операндов?
8. Для чего предназначены регистры mov, int, pop, push?
9. Расскажите о регистре Flags.

Лабораторная работа №11
«Режим микрокоманд (Ассемблер)»

Цель работы:

1 Приобрести умение разрабатывать микропрограммы для выполнения простейших арифметических операций.

1 Арифметические операции

1.1 Команды сложения и вычитания

Команды *сложения* и *вычитания* реализуют хорошо всем известные арифметические операции. Единственное, что нужно учитывать при использовании этих команд – особенности сложения и вычитания, связанные с представлением чисел в памяти компьютера.

***ADD* <операнд$_1$>, <операнд$_2$>**
***SUB* <операнд$_1$>, <операнд$_2$>**

Команда *ADD* складывает операнды и записывает их сумму на место первого операнда. Команда *SUB* вычитает из первого операнда второй и записывает полученную разность на место первого операнда. Операнды должны иметь одинаковый размер. Если первый операнд – регистр, то второй может быть также регистром, ячейкой памяти и непосредственным операндом. Если первый операнд – ячейка памяти, то второй операнд может быть регистром или непосредственным операндом. Возможно сложение и вычитание как знаковых, так и беззнаковых чисел любого размера. Команды меняют флаги AF, CF, OF, PF, SF и ZF.

```
a dd 45d
b dd -32d
c dd ?
mov   eax, a
add   eax, b
mov   c, eax              ; c = a + b
```

Команды *инкремента* и *декремента* увеличивают и уменьшают на 1 свой операнд.

***INC* <операнд>**
***DEC* <операнд>**

Операндом может быть регистр или ячейка памяти любого размера. Команды меняют флаги AF, OF, PF, SF и ZF. Команды инкремента и декремента выгодны тем, что они занимают меньше места, чем соответствующие команды сложения и вычитания.

```
inc   eax
```

К арифметическим операциям можно также отнести команду *изменения знака*:

***NEG* <операнд>**

Операндом может быть регистр или ячейка памяти любого размера. Команда *NEG* рассматривает свой операнд как число со знаком и меняет знак операнда на противоположный. Команда меняет флаги AF, CF, OF, PF, SF и ZF.

```
mov   ax, 1
neg   ax                  ; AX = -1 = ffffh
mov   bl, -128
neg   bl                  ; BL = -128, OF = 1
```

1.2 Команды умножения

Сложение и вычитание знаковых и беззнаковых чисел производятся по одним и тем же алгоритмам. Поэтому нет отдельных команд сложения и вычитания для знаковых и беззнаковых чисел. А вот умножение и деление знаковых и беззнаковых чисел производятся по разным алгоритмам, поэтому существуют по две команды умножения и деления.

Для *беззнакового умножения* используется команда *MUL*:

***MUL* <операнд>**

Операнд, указываемый в команде, – это один из сомножителей. Он может быть регистром или ячейкой памяти, но не может быть непосредственным операндом.

Местонахождение второго сомножителя и результата фиксировано, и в команде явно не указывается. Если операнд команды *MUL* имеет размер 1 байт, то второй сомножитель берётся из регистра AL, а результат помещается в регистр AX. Если операнд команды *MUL* имеет размер 2 байта, то второй сомножитель берётся из регистра AX, а результат помещается в регистровую пару DX:AX. Если операнд команды *MUL* имеет размер 4 байта, то второй сомножитель берётся из регистра EAX, а результат помещается в регистровую пару EDX:EAX.

Команда меняет флаги CF и OF. Если произведение имеет такой же размер, что и сомножители, то оба флага сбрасываются в 0. Если же размер произведения удваивается относительно размера сомножителей, то оба флага устанавливаются в 1.

```
x dw 256
```

```
mov   ax, 105
mul   x                         ; AX = AX * x, AX = 26880, CF = OF = 0
mov   eax, 500000
mov   ebx, 100000
mul   ebx               ; EDX:EAX = EAX * EBX, EDX:EAX = 50000000000, CF = OF = 1
```

Для *знакового умножения* используется команда *IMUL*:

***IMUL* <операнд>**
***IMUL* <операнд>, <непосредственный операнд>**
***IMUL* <операнд$_1$>, <операнд$_2$>, <непосредственный операнд>**
***IMUL* <операнд$_1$>, <операнд$_2$>**

Команда знакового умножения имеет несколько вариантов.

Первый соответствует команде *MUL* – один из сомножителей указывается в команде, второй должен находиться в регистре EAX/AX/AL, а результат помещается в регистры EDX:EAX/DX:AX/AX.

Второй вариант команды *IMUL* позволяет указать регистр, который будет содержать один из сомножителей. В этот же регистр будет помещён результат. Второй сомножитель указывается непосредственно в команде.

Третий вариант команды *IMUL* позволяет указать и результат, и оба сомножителя. Однако результат может быть помещён только в регистр, а второй сомножитель может быть только непосредственным операндом. Первый сомножитель может быть регистром или ячейкой памяти.

Четвёртый вариант команды *IMUL* позволяет указать оба сомножителя. Первый должен быть регистром, а второй – регистром или ячейкой памяти. Результат помещается в регистр, являющийся первым операндом.

Команда *IMUL* устанавливает флаги так же, как и команда *MUL*. Однако расширение результата в регистр EDX/DX происходит только при использовании первого варианта команды *IMUL*. В остальных случаях часть произведения, не помещающаяся в регистр-результат, теряется, даже если в качестве результата указан регистр EAX/AX. При умножении двух 1-байтовых чисел, произведение которых больше байта, но меньше слова, в регистре-результате получается корректное произведение.

```
mov   eax, 5
mov   ebx, -7
imul  ebx                       ; EAX = fffffdd, EDX = ffffffff, CF = 0
mov   ebx, 3
imul  ebx, 6                    ; EBX = EBX * 6
mov   ebx, 500000
imul  eax, ebx, 100000          ; EAX = EBX * 100000, старшая часть результата теряется
x dd 40
mov   eax, 55
imul  eax, x                    ; EAX = EAX * x
```

1.3 Команды деления

Деление, как и умножение, реализуется двумя командами, предназначенными для знаковых и беззнаковых чисел:

***DIV* <операнд>** **; Беззнаковое деление**
***IDIV* <операнд>** **; Знаковое деление**

В командах указывается только один операнд – делитель, который может быть регистром или ячейкой памяти, но не может быть непосредственным операндом. Местоположение делимого и результата для команд деления фиксировано.

Если делитель имеет размер 1 байт, то делимое берётся из регистра AX. Если делитель имеет размер 2 байта, то делимое берётся из регистровой пары DX:AX. Если же делитель имеет размер 4 байта, то делимое берётся из регистровой пары EDX:EAX.

Поскольку процессор работает с целыми числами, то в результате деления получается сразу два числа – частное и остаток. Эти два числа также помещаются в определённые регистры. Если делитель имеет размер 1 байт, то частное помещается в регистр AL, а остаток – в регистр AH. Если делитель имеет размер 2 байта, то частное помещается в регистр AX, а остаток – в регистр DX. Если же делитель имеет размер 4 байта, то частное помещается в регистр EAX, а остаток – в регистр EDX.

```
mov   ax, 127
mov   bl, 5
div   bl                        ; AL = 19h = 25,  AH = 02h = 2
mov   ax, 127
mov   bl, -5
idiv  bl                        ; AL = e7h = -25, AH = 02h = 2
mov   ax, -127
mov   bl, 5
idiv  bl                        ; AL = e7h = -25, AH = feh = -2
mov   ax, -127
mov   bl, -5
```

```
idiv  bl                    ; AL = 19h = 25,  AH = feh = -2
; x = a * b + c
mov   eax, a
imul  b
add   eax, c                ; Операнды команды сложения вычисляются слева направо
mov   x, eax
; x = a + b * c
mov   eax, b
imul  c
add   eax, a                ; Операнды команды сложения вычисляются справа налево
mov   x, eax
```

Задание

1. Изучите представленную теорию; Реализуйте приведенные примеры;
2. Ответьте на контрольные вопросы;
3. Реализацию примеров и ответы на вопросы оформите в виде отчета.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №11:

1. Что такое регистр процессора?
2. Какие регистры микропроцессорной памяти используются для адресации данных, команд программы, стековой памяти?
3. Назовите основные компоненты языка ассемблер, приведите структуру команд.
4. Приведите структуру ассемблерной программы и дайте краткую характеристику основных структурных фрагментов этой программы.
5. Каково назначение отладчика программ? Назовите основные его возможности.

Лабораторная работа №12
«Средства поддержки сегментации памяти. Виртуальное адресное пространство»

Цель работы:

1 Изучить понятие сегментации памяти;
2 Научиться работать с файлом подкачки.

1 Сегментация памяти

Недостатки системы, используется одно адресное пространство.

Один участок может полностью заполниться, но при этом останутся свободные участки. Можно конечно перемещать участки, но это очень сложно.

Эти проблемы можно решить, если дать каждому участку независимое адресное пространство, называемое **сегментом**. **Сегмент** - это логический объект.

В системах с сегментацией памяти каждое слово в адресном пространстве пользователя определяется виртуальным адресом, состоящим из двух частей: старшие разряды адреса, рассматриваются как номер сегмента, а младшие - как номер слова внутри сегмента. Наряду с сегментацией может также использоваться страничная организация памяти. В этом случае виртуальный адрес слова состоит из трех частей: старшие разряды 33 адреса определяют номер сегмента, средние - номер страницы внутри сегмента, а младшие - номер слова внутри страницы.

Как и в случае страничной организации, необходимо обеспечить преобразование виртуального адреса в реальный физический адрес основной памяти. С этой целью для каждого пользователя операционная система должна сформировать таблицу сегментов. Каждый элемент таблицы сегментов содержит описатель (дескриптор) сегмента (поля базы, границы и индикаторов режима доступа). При отсутствии страничной организации поле базы определяет адрес начала сегмента в основной памяти, а граница - длину сегмента. При наличии страничной организации поле базы определяет адрес начала таблицы страниц данного сегмента, а граница - число страниц в сегменте. Поле индикаторов режима доступа представляет собой некоторую комбинацию признаков блокировки чтения, записи и выполнения.

Таблицы сегментов различных пользователей операционная система хранит в основной памяти. Для определения расположения таблицы сегментов выполняющейся программы используется специальный регистр защиты, который загружается операционной системой перед началом ее выполнения. Этот регистр содержит дескриптор таблицы сегментов (базу и границу), причем база содержит адрес начала таблицы сегментов выполняющейся программы, а граница - длину этой таблицы сегментов. Разряды номера сегмента виртуального адреса используются в качестве индекса для поиска в таблице сегментов. Таким образом, наличие базово-граничных пар в дескрипторе таблицы сегментов и элементах таблицы сегментов предотвращает возможность обращения программы пользователя к таблицам сегментов и страниц, с которыми она не связана. Наличие в элементах таблицы сегментов индикаторов режима доступа позволяет осуществить необходимый режим доступа к сегменту со стороны данной программы. Для повышения эффективности схемы используется ассоциативная кэш-память.

Отметим, что в описанной схеме сегментации таблица сегментов с индикаторами доступа предоставляет всем программам, являющимся частями некоторой задачи, одинаковые возможности доступа, т. е. она определяет единственную область (домен) защиты. Однако для создания защищенных подсистем в рамках одной задачи для того, чтобы изменять возможности доступа, когда точка выполнения переходит через различные программы, управляющие ее решением, необходимо связать с каждой задачей множество доменов защиты. Реализация защищенных подсистем требует разработки некоторых специальных аппаратных средств. Рассмотрение таких систем, которые включают в себя кольцевые схемы защиты, а также различного рода мандатные схемы защиты, выходит за рамки данного обзора.

Реализация сегментации существенно отличается от страничной организации памяти: страницы имеют фиксированный размер, а сегменты — нет.

2 Виртуальное адресное пространство

2.1 Файл подкачки

2.1.1 Для чего нужен файл подкачки

Файл подкачки Windows 7, или иначе своп-файл (swap-file), является виртуальной памятью, располагающейся на одном из жёстких дисков, и представляет собой «продолжение» физической оперативной памяти (ОЗУ). Если при работе какого-либо приложения ему не хватает объёма установленного ОЗУ, то Windows 7 использует своп-файл для хранения данных приложения, то есть производит запись в него и чтение из него данных, которые не поместились в ОЗУ. Этот процесс записи и чтения носит название свопинга. В Windows 7 этот файл имеет строго определённое имя pagefile.sys, которое нельзя изменить.

При своей установке Windows 7 самостоятельно определяет необходимый размер своп-файла и размещает его на системном разделе жёсткого диска. Часто бывает так, что подобное поведение системы относительно размеров и размещения этого файла не даёт максимального быстродействия компьютера. Поэтому пользователю приходится самому настраивать параметры pagefile.sys и оптимизировать его работу. Постараемся осветить наиболее важные моменты этого процесса.

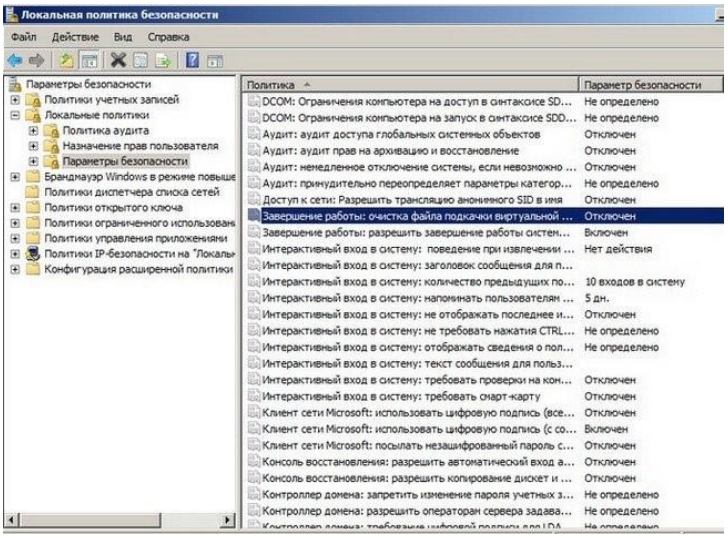
2.1.2 Оптимальные размеры файла подкачки

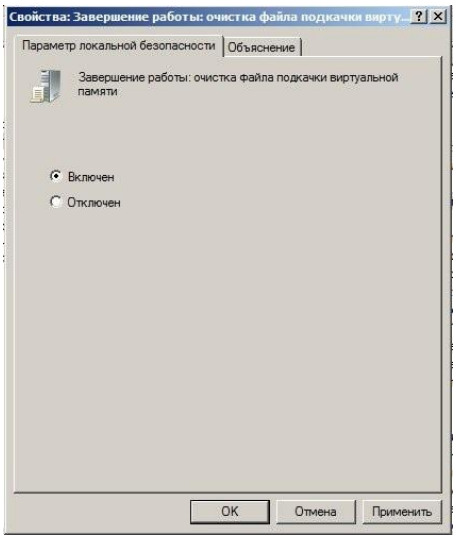
Считается, что для получения хорошей производительности требуется задать минимальный объем файла подкачки, равный объёму ОЗУ, а максимальный – вдвое больше. То есть, если компьютер имеет ОЗУ размером 2 Гб, то для своп-файла следует задать минимальный и максимальный размеры, равные 2 и 4 Гб, соответственно. Задание двух этих параметров с различными значениями приводит к тому, что фактический размер этого объекта дисковой памяти изменяется динамически, значит, он будет подвергаться фрагментации и снижать быстродействие. Поэтому многие пользователи задают одинаковые значения. В этом случае pagefile.sys становится статическим (не фрагментируемым), что снижает нагрузку на систему и повышает её быстродействие. Но и в случае динамического своп-файла есть способ устранения снижения производительности, если включить очистку файла при завершении работы операционной системы.

- 512 Mb — 2248-2248 Mb
- 1024 Mb — 1712-1712 Mb
- 2048 Mb — 1224-1224 Mb
- 4024 Mb — 768—768 Mb
- Гб (или выше) — без файла подкачки (т.е 0 Mb, т.е отключен).

2.1.3 Очистка своп-файла при завершении работы

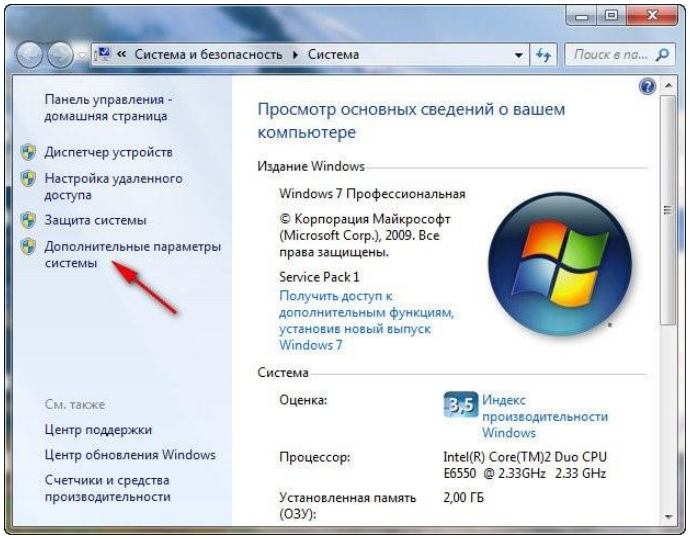
Для включения очистки файла pagefile.sys при завершении работы операционной системы необходимо в режиме командной строки выполнить команду secpol.msc («Пуск – Выполнить»). В открывшемся окне следует найти элемент «завершение работы: очистка файла подкачки…». Двойным щелчком мыши по нему устанавливаем параметр безопасности в значение «Включён» и нажимаем кнопку «Применить». Эти действия показаны двумя следующими рисунками.



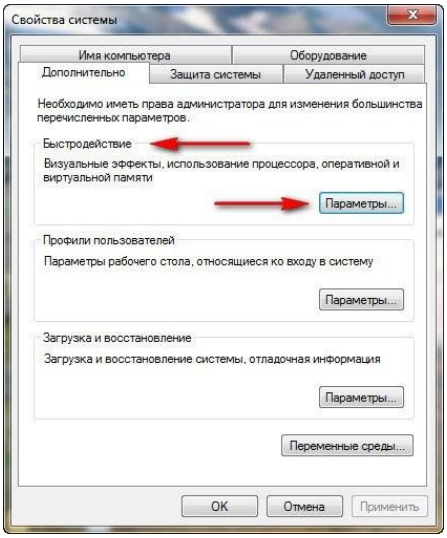


2.1.4 Выбор места расположения своп-файла

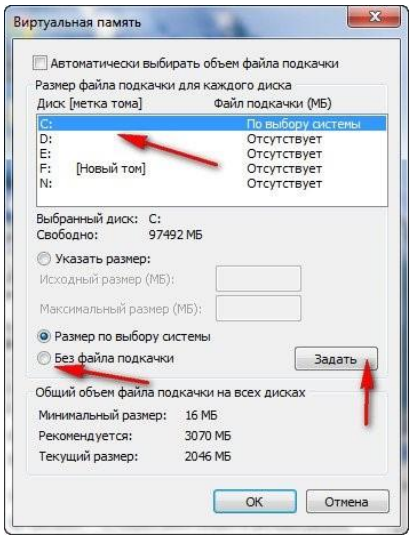
Windows 7 допускает располагать своп-файл одновременно на нескольких дисках, но этот вариант производительность системы не увеличивает. Наиболее оптимальным способом по отношению к производительности является один файл подкачки, расположенный на любом разделе жёсткого диска, кроме системного раздела. Для изменения места расположения pagefile.sys требуется вначале его удалить, а затем создать в нужном месте. Для этого щёлкаем «Пуск», а затем «Мой компьютер – Свойства» (правая кнопка мыши). В появившемся окне щёлкаем по «Дополнительные параметры системы».



Откроется окно, в котором нужно выбрать вкладку «Дополнительно».

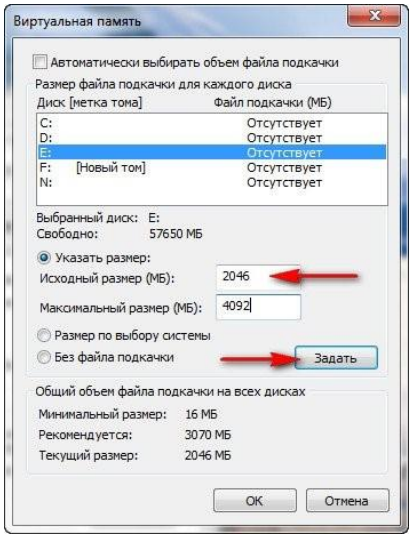


Здесь в разделе «Быстродействие» нажимаем кнопку «Параметры» и в открывшемся окне снова выбираем вкладку «Дополнительно». Обращаемся к разделу «Виртуальная память» и нажимаем кнопку «Изменить». Открывается окно, позволяющее задавать местоположение и размеры своп-файлов на любых разделах жёстких дисков. Это же окно позволяет как отключить файл подкачки, так и включить его.



Как уже говорилось, по умолчанию Windows 7 размещает pagefile.sys в системном разделе диска, что видно по активированному пункту «Размер по выбору системы». Для переноса файла подкачки на другой раздел, его следует выбрать, активировать пункт «Без файла подкачки» и нажать кнопку «Задать». Этими действиями своп-файл удаляется из своего прежнего местоположения. Иногда после этого появляется сообщение с предупреждением об отключении этого файла или задании слишком малого его объёма, что может привести к возникновению системной ошибки. В этом сообщении следует просто нажать «Да».

Теперь переходим на создание файла в нужном месте. Для этого: Выбираем требуемый раздел диска. Активируем пункт «Указать размер». Задаём размеры файла (минимальный максимальный). Нажимаем кнопку «Задать». Все эти действия показаны на следующем рисунке.



В приведённом рисунке пользователь создал динамический своп-файл, у которого максимальный размер вдвое превышает минимальный. При необходимости создать статический pagefile.sys, эти параметры должны быть одинаковыми.

Следует отметить, что своп-файл может быть изменён в своих размерах без изменения его местоположения. Для этого все приведённые выше действия нужно выполнить без изменения раздела диска. Разумеется, излишне говорить, что во всех окнах с кнопкой «ОК» её следует нажимать по окончании всех требуемых действий. И ещё: при изменении объёма в сторону уменьшения изменения вступают в действие мгновенно, в противном случае может понадобиться перезагрузка, о чём пользователь получит сообщение, в котором нужно нажать «ОК».

2.1.5. Отключение и включение файла подкачки

Многие пользователи часто прибегают к отключению своп-файла. Это оправдано в тех случаях, когда в компьютере установлено ОЗУ достаточно большого размера. И действительно, зачем тратить время на свопинг и уменьшать быстродействие системы, если можно просто добавить одну или несколько планок ОЗУ. Стоимость ОЗУ в данное время не так уж велика, зато увеличение объёма установленного ОЗУ и отключение файла подкачки может дать существенное увеличение производительности, особенно при не очень мощном компьютере. Отключить своп-файл легко – для этого достаточно удалить его, задав чекбокс «Без файла подкачки», как показано на предпоследнем рисунке.

Включить своп-файл в работу так же просто, как и отключить – достаточно только снять галку «без файла подкачки» и установить «Размер по выбору системы» или задать свои его значения.

Полезные советы.

- Попытаемся дать несколько рекомендаций от опытных пользователей по настройке и оптимизации файла подкачки. Вот они:
- Излишне говорить, что идеальным вариантом будет sys, расположенный на отдельном разделе жёсткого диска.
- Материальное положение пользователя не сильно пошатнётся, если он установит дополнительные планки ОЗУ и вообще откажется от виртуальной памяти. Это даст увеличение скорости работы Windows Для неё вполне хватит 6 ГБ оперативной памяти.
- Windows 7 производит постоянное увеличение размера своп-файла, что приводит к фрагментации жёсткого диска и лишним накладным расходам. Поэтому рекомендуется задать свой размер с одинаковыми значениями для минимального и максимального размеров.
- Не следует задавать размер этого файла менее 1 Гб, иначе возможна фрагментация жёсткого диска.
- Отдельной рекомендации требует случай использования в качестве системного SSD-диска. Это твердотельный накопитель, не имеющий механических вращающихся элементов. По сути – это большая флешка с очень высокой скоростью чтения-записи, обычно на порядок выше скорости жёстких дисков. Но при всех её очень хороших скоростных параметрах она имеет ограниченное число циклов запись-чтение. Поэтому очень важно обеспечить минимальное количество перезаписей на неё, а для этого нужно или вообще отключить своп-файл, или сделать его статическим.

Задание

1. Посмотрите какой swap-file на вашем ПК;
2. При необходимости поменяйте размер swap-file;
3. Ответьте на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №12:

1. Для чего используется сегментация памяти?
2. Для чего используется файл подкачки?
3. Как изменить размер файла подкачки?
4. Что такое своп-файл? Для чего отключают файл-подкачки?

Лабораторная работа №13
«Файловые операции»

Цель работы:

1. Познакомиться с файлами, файловыми системами и основными файловыми операциями.

1 Файлы и файловая система. Операции с файлами

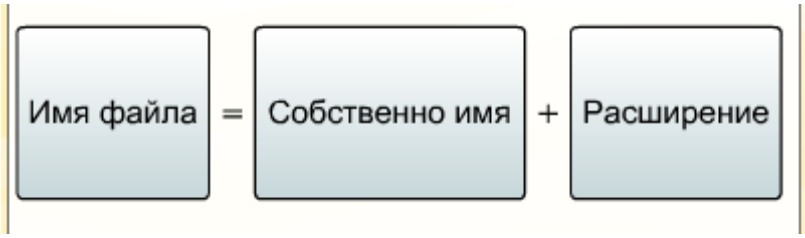
1.1 Файлы

Любая информация – текстовые данные, программы, фотографии и т.п. – может сохраняться во внешней памяти компьютера **только** в виде файлов. В информатике используется два определения файла.

Определение 1 – *файл* – логическая единица хранения информации

Определение 2 – *файл* — это упорядоченная совокупность данных, хранимая на диске и занимающая именованную область внешней памяти. Файл в процессе обработки рассматривается как единое целое, которое имеет собственное *имя*

Имя файла состоит из двух частей, отделенных друг от друга точкой. Первая часть – это собственно имя файла, вторая часть – расширение. По расширению, как правило, можно определить тип файла: например doc – это текстовый файл, а exe – исполняемый файл. Современные операционные системы позволяют настроить соответствие («ассоциацию») расширения файла с программой, обрабатывающей файлы такого типа.



В различных операционных системах приняты разные требования к именам файлов.

В операционной системе MS-DOS непосредственно имя файла должно было содержать не более 8 разрешенных символов, к которым относились латинские буквы, цифры и знак нижнего подчеркивания. Расширение файлов в этой операционной системе состояло из 3 латинских букв. Например, в системе MS-DOS можно было работать с файлом test.txt, но имя файла корова.html являлось недопустимым.

В операционной системе Windows (начиная с версии 95) имя файла должно иметь не более 255 символов, причем в имени файла могут использоваться и русские буквы. Кроме того, в имени файла может использоваться любое количество символов точки; при этом расширение имени может иметь любую длину и располагается после последней по счету точки.

Приведем примеры типов файлов и наиболее распространенных расширений для этих типов:

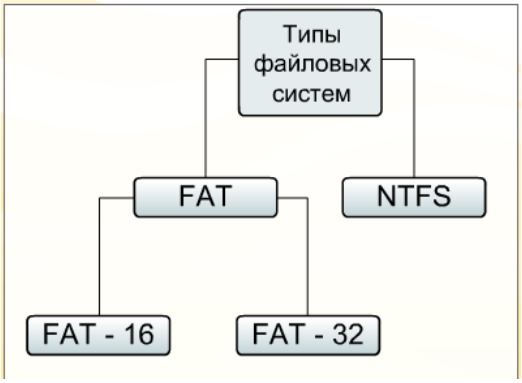
Тип файла	Расширение
Программы	exe, com
Текстовые файлы	doc, txt, docx
Графические файлы	bmp, jpeg, jpg, gif
Звуковые файлы	wav, mp3
Видеофайлы	avi, mov
Коды программ на языках программирования	pas, bas

1.2 Файловая система

Пользователь работает с файлом как с единой логической единицей данных только благодаря *файловой системе*, в действительности же файл может быть разбросан «кусочками» по всему внешнему носителю, причем его разбиение никак не будет связано с логической структурой файла.

Определение. *Файловая система* – функциональная часть ОС, которая отвечает за хранение данных на внешних носителях и обмен данными между внешними носителями.

Операционная система сама осуществляет все необходимые операции по чтению и записи файлов, скрывая от пользователя достаточно сложный механизм размещения файлов во внешней памяти, что, конечно же, удобно для пользователя. Устройство файловой системы напрямую зависит от операционной системы, установленной на компьютере. Однако одна и та же операционная система может использовать несколько разных файловых систем. Так, современные операционные системы Windows могут работать с файловыми системами FAT и NTFS.

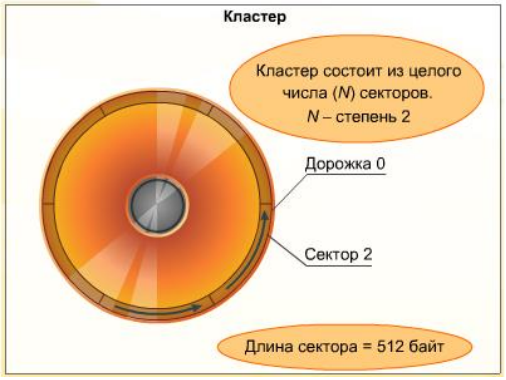


В любой файловой системе, кроме информации, содержащейся в файле, необходимо хранить некоторую вспомогательную информацию, такую как дата создания файла, размер файла и некоторые другие *атрибуты* файла. Такая информация хранится в специальной служебной области, которая по аналогии с всем известным библиотечным термином была названа **каталогом.** Каталог также содержит имена файлов и указание на начало каждого файла на диске (номер кластера).

Следует помнить, что по именам к файлам обращается пользователь, а ОС обращается к файлам только по номерам их первых кластеров.

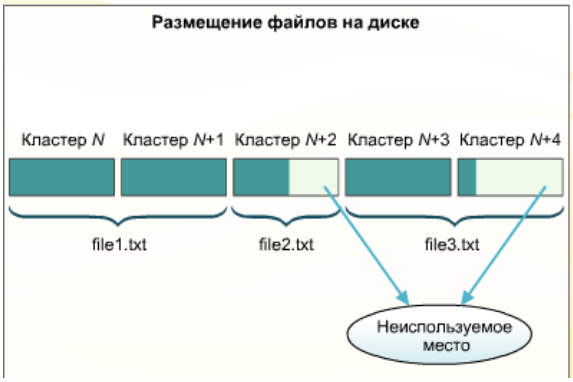
Определение. *Кластер* – это минимальный адресуемый блок дисковой памяти для записи/чтения данных на дисковом накопителе.

Жесткий диск состоит из дорожек, где каждая дорожка делится на секторы. Кластер может занимать один или несколько таких секторов. «Координатами» файла для ОС являются номера занятых им кластеров.



Все файлы на диске вне зависимости от своего объема имеют размер, кратный размеру кластера. Любой маленький файл не может занимать размер меньше кластера.

Из-за такой организации файловой системы на диске теряется часть места (много маленьких файлов занимают на диске заметно больше места по сравнению с их суммарной длиной).



Можно продолжить «библиотечную» аналогию, тогда область хранения файлов будет соответствовать книгохранилищу, а каталог – картотеке библиотеки. Кроме того, точно так же, как книга состоит из страниц, на диске файл состоит из кластеров.

Разберемся теперь, как устроены файловые системы FAT и NTFS.

1.2.1 Файловая система FAT (File Allocation Table – «таблица размещения файлов»)

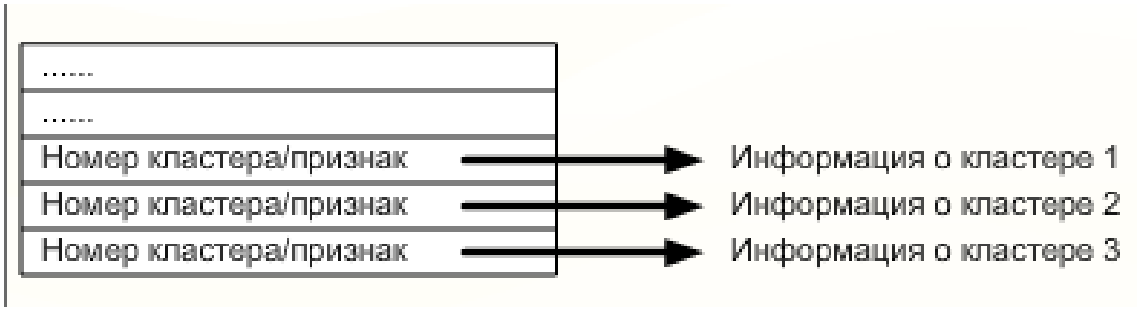
Файловая система FAT устроена следующим образом:

БД «Корневой каталог» – это база данных, в которой хранится информация о файлах и папках, находящихся на диске. Упрощенно структуру БД «Корневой каталог» можно представить в виде таблицы:

8 байт	3 байта	1 байт	2 байта	2 байта	2 байта	4 байта
Имя файла	Расширение	Атрибут	Время создания	Дата	Номер начального кластера	Размер файла

Общая длина записи о каждом файле в корневом каталоге составляет 32 байта и содержит, кроме указанной выше, также другую информацию, необходимую операционной системе для работы с файлами. Кроме того, для файлов с «длинными» именами (в ОС Windows 95 и более новых версий) дополнительные поля используются для хранения имен и расширений общей длиной 255 байт, а в основной записи корневого каталога записывается «усеченное» имя в стандарте «8.3» (8 символов имени + 3 символа расширения) и ссылка на запись исходного «длинного» имени файла.

БД «Элементы FAT» содержит информацию о кластерах диска. Упрощенно структуру этой базы данных можно представить следующим образом:



Для каждого кластера, имеющегося на диске (кластеры нумеруются последовательно начиная с нуля, причем «служебные» кластеры 0 и 1 отмечаются особо и для записи файлов не используются), в таблице FAT отводится одна запись («ячейка»), которая указывает возможное текущее состояние данного кластера:

- кластер свободен – значение 0;
- bad-кластер(поврежденный кластер), то есть испорченный кластер, который по каким-то причинам использовать уже нельзя;
- кластер занят каким-либо файлом - в качестве признака записывается номер следующего кластера (сектора), занятого этим файлом;
- последний кластер файла;
- «зарезервированный» кластер.

Значения ячеек FAT для bad-кластеров, последних кластеров файлов и «зарезервированных» кластеров зависят от конкретной версии FAT.

До какого-то момента времени использование файловых таблиц FAT было очень удобным. Под номер кластера отводилось ровно 2 байта, то есть 16 бит. По этой причине такие файловые системы получили название FAT16. Однако с увеличением объема дисков производители программного обеспечения столкнулись со следующей проблемой. Так как под номер кластера отводится 2 байта, то, следовательно, всего можно адресовать 65536 различных кластеров. В результате оказалось, что файловые системы FAT16 не могут работать с дисками объемом более 2 Гбайт. Подсчитаем минимальный размер кластера для дисков разных объемов:

1,2 Гбайт = 1,2 Гбайт/65536 = 19,2 Кбайт/кластер ≈32 Кбайт
2,1 Гбайт = 2,1 Гбайт/65536 = 33,6 Кбайт/кластер ≈ 64 Кбайт.

Таким образом, на диске емкостью 2 Гбайт и более кластеры имеют размер 64 Кбайт, или 65536 байт. Но самое большое значение, которое может быть представлено в 16 разрядах, составляет 65535. В Microsoft обнаружили, что при разработке многих существующих в то время программ их авторы исходили из предположения,

что значение числа байт в кластере умещается в 16 разрядах. Поэтому после появления дисков объемом больше 2 Гбайт, появилась файловая система FAT32, в которой под номер кластера было отведено уже 4 байта. При этом адресуемое число кластеров стало гораздо большим, а минимальный размер кластера мог составлять уже 512 байт. Правда, в таком случае размер самой файловой системы становится очень большим, что снижает удобство ее использования.

Недостатки файловой системы FAT16:

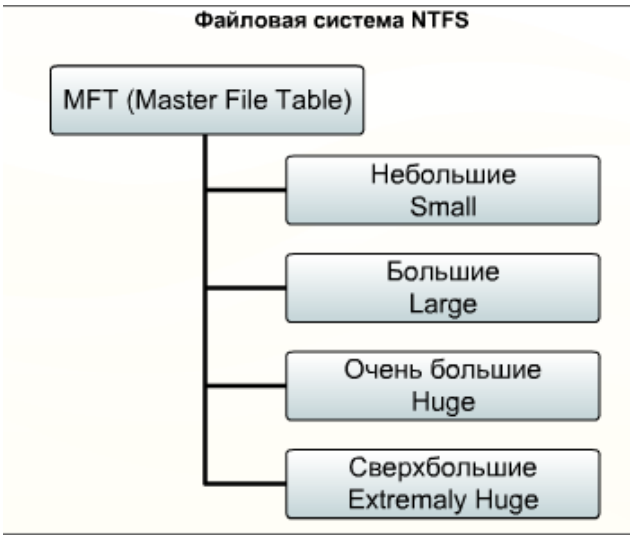
- ограничения на объем диска и размер файла;
- ограничения длины имени файла;
- фрагментация файлов, приводящая к снижению быстродействия и износу оборудования;
- потери памяти диска, вызванные большими размерами кластера.

В файловой системе FAT32 были сняты ограничения на длину имени и размер корневого каталога, но остальные ограничения, присущие FAT, остаются в силе.

1.2.2 Файловая система NTFS (New Technology File System – «файловая система по новой технологии»)

Структура NTFS

В файловой системе NTFS все файлы подразделяются по размеру на следующие категории:



Для каждого файла в MFT выделается отдельная запись размером в 2 Кбайт следующего вида:

H (заголовок)	**SI** (атрибуты стандартной информации)	**FN** (имя файла)	**Data** (поток данных)	**SD** (дескриптор безопасности)

Небольшие файлы целиком помещаются в MFT в поле «Data». Для остальных файлов в поле «Data» указывается номер кластера, с которого начинается непрерывный фрагмент, и количество кластеров в этом фрагменте.

Файловая система NTFS поддерживает почти любые размеры кластеров от 512 байт до 64 Кбайт, однако неким стандартом считается кластер размером 4 Кбайт.

Файловая система NTFS разбивает диск следующим образом:

- первые 12% диска отводятся под так называемую MFT-зону – пространство, в котором находится MFT; запись каких-либо данных в MFT-зону невозможна – это делается, чтобы этот самый главный служебный файл (MFT) не фрагментировался при его увеличении;
- остальные 88% диска представляют собой обычное пространство для хранения файлов.

Длинные и короткие имена файлов

В файловой системе NTFS для имен файлов введены следующие правила:

- имена файлов могут иметь длину до 255 символов;
- имена файлов записываются в 16-разрядной кодировке Unicode;
- система NTFS автоматически генерирует имя файла, поддерживаемое в MS-DOS («короткое» имя) и записывает его в поле «SI» в MFT.

Примеры:

1) «Длинное имя файла.русское» → «ДЛИННО~1.РУС»;
2) «Длинное письмо.руслан» → «ДЛИННО~2.РУС».

«Короткое» имя при этом генерируется по следующим правилам:

- удаляются все пробелы, запрещённые символы и точки, кроме одной;
- имя усекается до 6 символов, а в конце его добавляется символ «~» (тильда) и порядковый номер;

− расширение имени файла усекается до 3 символов.

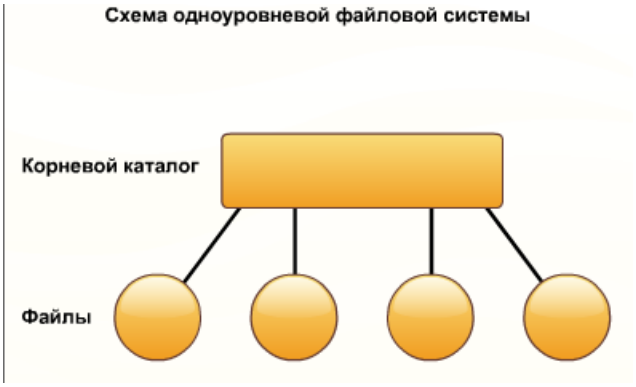
Например:

1) «Длинное имя файла. Русское» → «ДЛИННО~1.РУС»
2) «Длинное письмо. Руслан» → «ДЛИННО~2.РУС» .

1.3 Типы файловых систем

1.3.1 Одноуровневые файловые системы

Давным-давно, когда внешними носителями на персональных компьютерах были лишь гибкие 8- и 5-дюймовые дискеты в бумажных конвертах, количество хранимых файлов на них было относительно невелико, и вся служебная информация могла храниться в едином каталоге. Такая файловая система называлась *одноуровневой* и выглядела следующим образом:



Такая файловая система с одним каталогом использовалась, например, операционными системами CP/M и RT-11 (небольшая однопользовательская операционная система реального времени, разработанная фирмой DEC для 16-битных компьютеров серии PDP-11). Одноуровневую файловую систему можно представить следующей таблицей:

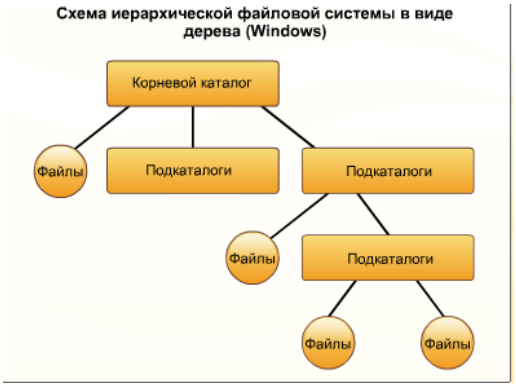
Имя файла	Номер начального сектора
Файл1	10
Файл2	39
...	...
Файл100	879

1.3.2 Иерархические файловые системы

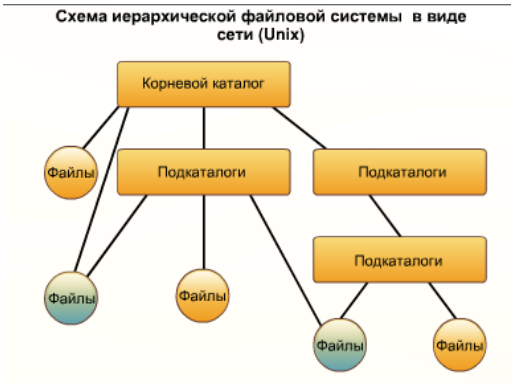
Появление жестких дисков объемом в несколько мегабайт кардинально изменило ситуацию. Теперь на таком диске можно было хранить сотни и даже тысячи файлов, список которых уже не умещался на одном экране, и поиск нужного файла стал весьма проблематичным. В связи с этим организация хранения файлов во внешней памяти изменилась, и файловая система приобрела иерархическую структуру.

Основной каталог в иерархической файловой системе остался, но теперь он стал называться **корневым каталогом**. В нем могут храниться не только файлы, но и другие каталоги более низкого уровня, называемые подкаталогами, или поддиректориями. В операционной системе Windows подкаталоги принято называть **папками,** хотя, по существу, термин «папка» совпадает с классическим понятием каталога.

Представить иерархическую файловую систему для операционных систем MS-DOS и Windows можно с помощью **файлового дерева.**



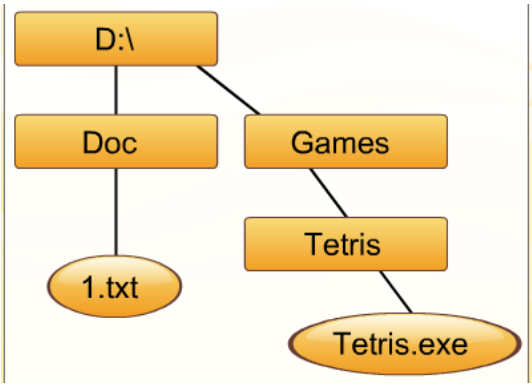
Для операционных систем семейства Unix (например, для ОС Linux) файловую систему принято называть сетью, а представить ее можно в виде графа.



Принципиальное различие организации файловых систем для Windows и для Unix состоит в том, что если для файловых систем Windows можно установить однозначное соответствие между каталогом и хранимыми в нем файлами, то для файловых систем Unix это вовсе не обязательно. В файловых системах ОС Unix некоторые файлы могут одновременно принадлежать нескольким каталогам и даже по-разному называться в разных каталогах (такие файлы отмечены на рисунке другим цветом).

Пример 1. Рассмотрим иерархическую файловую систему на конкретном примере. Каждый диск имеет логическое имя, причем гибкие диски обычно имеют имена A: и B:. Остальные буквы латинского алфавита отводятся для жестких, оптических и флеш-дисков.

Пусть в корневом каталоге диска D: имеются два каталога первого уровня: Doc и Games, а в каталоге Games имеется каталог второго уровня - Tetris. В каталоге Doc хранится файл с именем 1.txt, а в каталоге Tetris – файл с именем tetris.exe.



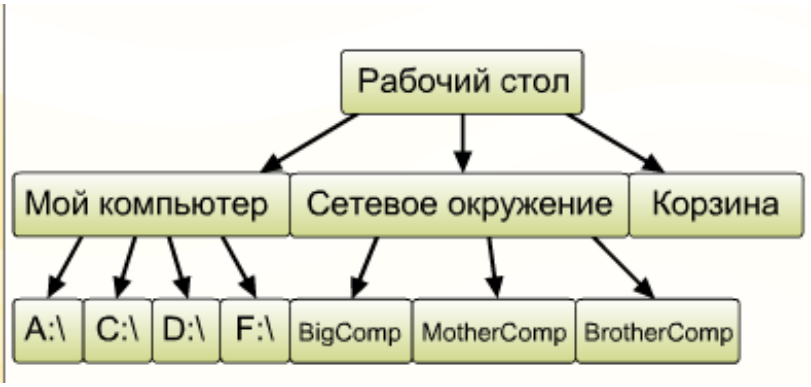
Чтобы найти нужный файл в указанной иерархической структуре, необходимо указать *путь к этому файлу*. Для этого нужно указать логическое имя диска, на котором хранится файл, а потом через знак «\» указать последовательно все подкаталоги, вложенные друг в друга, где последним будет тот подкаталог, в котором расположен файл. Например, в показанной выше иерархии путь к файлу 1.txt записывается как D:\Doc\, а путь к файлу tetris.exe – как D:\Games\Tetris\.

Определение. Если указаны и путь к файлу, и его имя, то такая конструкция называется *полным именем файла*.

Так, полное имя файла tetris.exe – это D:\Games\Tetris\tetris.exe.

1.4 Возможности графического интерфейса для работы с файловой системой

Операционная система Windows предоставляет пользователю возможность работать с файловой системой с помощью графического интерфейса, представляя ее в виде стилизованных папок и документов. Для MS-DOS в вершине дерева располагается корневой каталог диска, для Windows же корневым каталогом можно считать папку «Рабочий стол», в которой располагаются подкаталоги «Мой компьютер», «Корзина» и «Сетевое окружение». Эти подкаталоги носят стандартные названия и служат для заранее определенных целей.



Папка **«Мой компьютер»** в качестве подкаталогов содержит все диски, имеющиеся в данном компьютере, и все сетевые диски, которые подключил к данному компьютеру пользователь.

Папка **«Сетевое окружение»** содержит папки всех компьютеров, которые подключены в этот момент к локальной сети и к которым можно обратиться с данного компьютера.

Папка **«Корзина»** временно хранит все удаленные на данном компьютере папки и файлы. При необходимости хранящиеся в «Корзине» файлы и папки можно восстановить. Чтобы безвозвратно удалить содержимое «Корзины», необходимо выполнить операцию удаления из папки «Корзина». Следует отметить, что файл можно удалить безвозвратно и сразу, минуя «Корзину». Для этого необходимо выделить файл и нажать комбинацию клавиш Shift + Del. Кроме того, нужно не забывать, что при недостатке места на диске «Корзина» самоочищается.

1.5 Специфика организации каталогов для разных видов устройств

Интересно, что от вида устройства может зависеть и способ организации каталогов.

Так, на дискетах, для которых характерно частое обновление информации, каждый подкаталог хранится автономно и фактически является специфическим видом файла.

На оптических дисках CD-ROM и DVD-ROM ситуация другая. Так как все каталоги там обычно создаются единовременно во время сессии записи, а длина всех записываемых файлов известна заранее, то для ускорения доступа к файлам вся информация каталога группируется в одном месте, а каждый файл размещается на диске непрерывно и однозначно определяется его началом и длиной.

Совсем по-другому ведется организация файловой системы на флеш-дисках. При сохранении информации на таком диске принимаются специальные меры по оптимизации размещения данных так, чтобы они размещались наиболее равномерно и нагрузка на различные блоки флеш-диска тоже осуществлялась равномерно.

Организация файловой системы сильно зависит и от архитектуры ЭВМ. Например, файловые системы на компьютерах Apple и на более привычных нам IBM-совместимых компьютерах различны.

Как уже говорилось ранее, для пользователя файл является единым целым, однако операционная система делит его на отдельные блоки – кластеры, которые состоят из нескольких секторов. При записи большого файла на диск, на котором ранее было записано много маленьких файлов, а затем часть этих файлов была удалена, ОС записывает новый файл отдельными «кусками» на имеющиеся свободные участки, а при чтении такого файла – снова «соединяет» все такие его фрагменты. Подобная ситуация (замедляющая работу с файлом, так как на доступ к каждому отдельному его «кусочку» требуется больше времени, чем на чтение слитной последовательности кластеров) называется *фрагментацией*. Для ее устранения периодически требуется выполнять операцию *дефрагментации* диска, которая заключается в переразмещении (переписывании) файлов в виде сплошных последовательностей кластеров (насколько это возможно). Впрочем, в современных ОС реализованы алгоритмы записи файлов, существенно уменьшающие фрагментацию (например, ОС сначала «старается» записывать файлы целиком в остающейся свободной области в конце диска и только при ее отсутствии начинает записывать файлы фрагментарно).

1.6 Операции с файлами

В процессе работы на компьютере с файлами чаще всего приходится выполнять следующие операции:

- копирование;
- перемещение;
- удаление;
- переименование;
- создание.

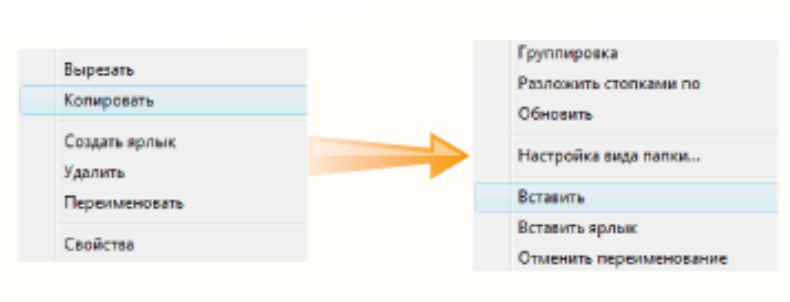
Выполнять операции с файлами можно как непосредственно при помощи графического интерфейса операционной системы, так и с помощью специализированных приложений – различных *файловых менеджеров*: стандартного для Windows приложения «Проводник», программ Norton Commander, Far, Windows Commander и многих других.

Копирование (копия файла помещается в другой каталог) – при выполнении этой операции физически создаются новый файл и новая запись в файловой системе.

Чтобы скопировать файл в «Проводнике» Windows, достаточно щелкнуть на имени файла правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Копировать». После этого необходимо перейти в нужный каталог, щелкнуть на свободном месте его окна правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Вставить».

Аналогичного результата можно достичь нажатием комбинаций клавиш **Ctrl+C** после выделения файла и **Ctrl+V** – после перехода в нужный каталог.

Эти же операции применимы и для копирования целых подкаталогов со всем их содержимым.

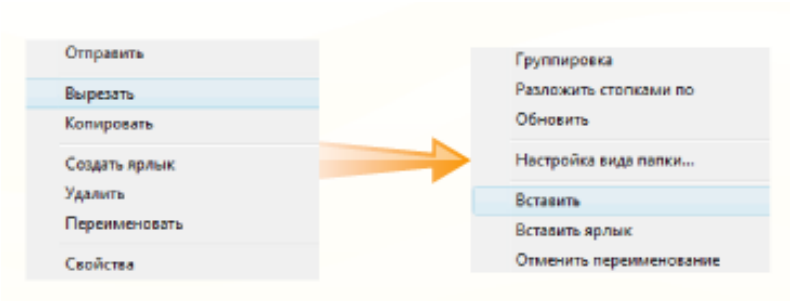


Перемещение (файл перемещается в другой каталог) – при выполнении этой операции файл физически остается на диске на прежнем месте, но меняется его «адрес» в файловой системе.

Чтобы переместить файл, в «Проводнике» Windows достаточно щелкнуть на имени файла правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Вырезать». После этого необходимо перейти в нужный каталог, щелкнуть на свободном месте его окна правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Вставить».

Аналогичного результата можно достичь нажатием комбинаций клавиш **Ctrl+X** после выделения файла и **Ctrl+V** – после перехода в нужный каталог.

Эти же операции применимы и для перемещения целых подкаталогов со всем их содержимым.



Удаление –физически файл остается на диске, но из файловой системы удаляется запись о нем.

Чтобы удалить файл, в «Проводнике» Windows достаточно щелкнуть на имени файла правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Удалить». Вслед за этим появляется диалоговое окно с просьбой подтвердить удаление. Это делается, чтобы предотвратить случайное удаление нужного файла. При таком удалении файл будет перемещен в «Корзину».

Аналогичного результата можно достичь нажатием клавиши **Del** для заранее выделенного файла.

Следует отметить, что нажатие комбинации клавиш **Shift + Del** позволяет безвозвратно удалить файл минуя «Корзину». Подтверждение удаления файла, конечно же, запрашивается и в этом случае.

Аналогично можно удалять и целые подкаталоги со всем их содержимым

Переименование – Переименование – изменяется только имя файла в файловой системе, а сам этот файл остается неизменным.

Чтобы переименовать файл, в «Проводнике» Windows достаточно щелкнуть на имени файла правой кнопкой мыши и выбрать в появившемся контекстном меню пункт «Переименовать». После этого имя файла становится доступным для редактирования, и пользователь может ввести новое имя файла. Обычно предлагается ввести именно имя файла, а расширение файла не должно меняться, так как это может привести к невозможности прочтения файла целевой программой. При наличии соответствующих установок свойств папки ничто не запрещает пользователю вручную изменить и расширение файла, но операционная система при этом запросит дополнительное подтверждение того, что это изменение делается пользователем осознанно.

Аналогичного результата можно достичь нажатием клавиши **F2** для предварительно выделенного файла.

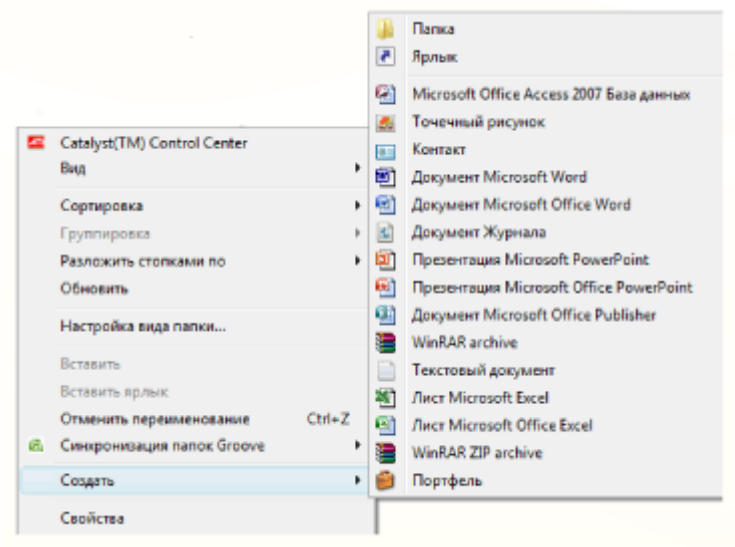
Аналогичным образом осуществляется и переименование каталогов.

Создание

Чаще всего файлы создаются программно: автоматически или при сохранении пользователем введенной в ту или иную программу информации (например, при сохранении на диске текста, набранного в редакторе Word). При создании файла ОС ищет на диске свободное место, в котором можно начать размещать файл, а после его записи в каталог записывается адрес начала файла (номер его первого кластера), имя, дата создания и другая служебная информация.

Однако операционная система Windows позволяет создавать новые файлы и непосредственно в выбранном каталоге. Для этого необходимо на свободном месте в окне каталога щелкнуть правой кнопкой мыши и выбрать в контекстном меню пункт «Создать», а затем - нужный тип файла. После выбора типа файла система создаст пустой файл с нужным расширением и стандартным именем, который остается только переименовать и заполнить нужной информацией (открыв его для этого в соответствующей целевой программе)

Аналогично можно создавать и новые вложенные каталоги. Для этого в том же самом контекстном меню нужно выбрать расположенный в самом верху пункт «Папка».



1.7 Команды ОС для работы с файлами

Современные операционные системы предоставляют пользователю удобный графический интерфейс для работы с файлами. Но интересно узнать, какие команды ОС «спрятаны» от пользователя, что происходит «внутри» ОС, когда мы делаем нужные движения мышью.

Оказывается, что операционная система использует для работы с файлами не более 20 команд, а суть всех операций над файлами не зависит от операционной системы. Например, в операционной системе MS-DOS все операции над файлами на уровне файловой системы выполнялись так же, как и в системе Windows. Однако для пользователя все это выглядело несколько иначе.

Для выполнения операций с файлами в MS-DOS пользователь должен был набрать нужную команду в командной строке. С интерфейсом командной строки MS-DOS можно работать и в операционной системе Windows. Для этого в меню **Пуск** нужно выбрать пункт «Выполнить» и набрать в появившейся строке команду «cmd». После подтверждения ее ввода (нажатием клавиши Enter) появится черное окно, в котором можно набирать команды MS-DOS.

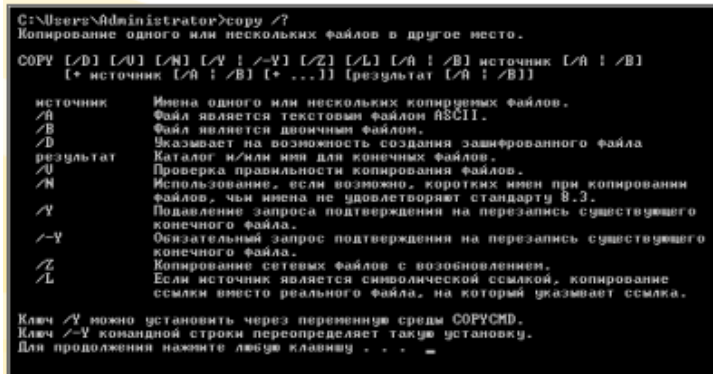
В ответ на приглашение системы (в виде символа «>» в конце выведенной строки с указанием пути к текущему каталогу) можно вводить следующие команды для работы с файлами и каталогами:

Название команды	Действие	Пример
Команды для работы с файлами		
copy	Копирование	copy d:\1.txt c:\2.txt
del, erase	Удаление	del 2.doc
move	Перемещение	move d:\array.pas progs
rename	Переименование	rename 1.doc 5.doc
Команды работы с каталогами		
dir	Вывод содержимого текущего каталога	dir *.bak – выведет информацию обо всех файлах с расширением bak, имеющихся в текущем каталоге
mkdir (или md)	Создание каталога	md games – создаст директорию games в текущем каталоге
chdir (или cd)	Смена текущего каталога	cd\ – переход в корневой каталог cd.. – переход на один каталог выше (обязательно надо ввести две точки подряд)
rmdir	Удаление каталога	rd d:\video – удаление каталога d:\video
Команды работы с дисками		
format	Форматирование диска	format D: /Q – быстрое форматирование диска D:
defrag	Дефрагментация диска	defrag D: -w -f
path	Вывод или задание пути поиска исполняемых програм	path или path d:\progs

Заметим, что многие команды MS-DOS имеют сокращенное написание, например, chdir → cd, mkdir → md.

Для всех команд обработки файлов следует отметить: если после команды указывается только имя файла без записи полного пути к нему, то данная команда будет искать и обрабатывать файл в текущем каталоге.

Существует множество команд, выполняемых из командной строки MS-DOS, причем у каждой команды свой формат и свои параметры. Но в запоминании форматов и параметров каждой команды нет необходимости, так как существует удобная система подсказок по всем командам MS-DOS. Чтобы получить подсказку по какой-либо команде, достаточно ввести после этой команды *ключ /?* или набрать help <имя команды>. Команда help выведет список всех команд с их кратким описанием. Например, если мы хотим получить информацию о команде copy, то необходимо набрать в командной строке: *copy /?*.



1.8 Маски имен файлов

Часто при поиске файлов мы не можем точно указать, какой именно файл мы ищем. Например, нам может понадобиться найти документ, созданный в текстовом редакторе Word, но мы не помним точно, как он называется. Просматривать же все папки и отсеивать ненужные нам документы слишком долго.

В подобной ситуации мы можем воспользоваться системой поиска так, чтобы она выдала нам список всех документов с расширением .doc. Оказывается, что при поиске документа можно пользоваться *масками имен*, или *подстановочными символами*. Символ «?» означает один любой символ, а символ «*» – любое количество или отсутствие любых символов. Тогда для поиска всех документов с расширением .doc можно указать в строке для поиска следующую комбинацию: *.doc

Этими же подстановочными символами можно пользоваться и при вводе команд в командной строке. Например, команда del *.exe означает удаление всех файлов с расширением .exe из текущего каталога, а команда del c:\games*.* удалит вообще все файлы из каталога Games. А в примере, показанном на рисунке ниже, в каталоге «Рабочий стол» осуществляется поиск всех файлов, имена которых состоят из четырех символов.

2 Основные цели использования файла

- *Долговременное и надежное хранение информации*. Долговременность достигается за счет использования запоминающих устройств, не зависящих от питания, а высокая надежность определяется средствами защиты доступа к файлам и общей организацией программного кода ОС, при которой сбои аппаратуры чаще всего не разрушают информацию, хранящуюся в файлах.
- *Совместное использование информации*. Файлы обеспечивают естественный и легкий способ разделения информации между приложениями и пользователями за счет наличия понятного человеку символьного имени и постоянства хранимой информации и расположения файла. Пользователь должен иметь удобные средства работы с файлами, включая каталоги-справочники, объединяющие файлы в группы, средства поиска файлов по признакам, набор команд для создания, модификации и удаления файлов. Файл может быть создан одним пользователем, а затем использоваться совсем другим пользователем, при этом создатель файла или администратор могут определить права доступа к нему других пользователей. Эти цели реализуются в ОС файловой системой.

Файловая система (ФС)— это часть операционной системы, включающая:

- совокупность всех файлов на диске;
- наборы структур данных, используемых для управления файлами, такие, например, как каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске;
- комплекс системных программных средств, реализующих различные операции над файлами, такие как создание, уничтожение, чтение, запись, именование и поиск файлов.

Таким образом, файловая система играет роль промежуточного слоя, экранирующего все сложности физической организации долговременного хранилища данных, и создающего для программ более простую логическую модель этого хранилища, а также предоставляя им набор удобных в использовании команд для манипулирования файлами.

Широко известны следующие файловые системы:

1. *файловая система операционной системы MS-DOS, в основу которой положена таблица размещения файлов - FAT (File Allocation Table).*

Таблица содержит сведения о расположении всех файлов (каждый файл делится на *кластеры* в соответствии с наличием свободного места на диске, кластеры одного файла не обязательно расположены рядом). Файловая система MS-DOS имеет значительные ограничения и недостатки, например, под *имя* файла отводится 12 байт, работа с жестким диском большого объема приводит к значительной фрагментации файлов;

Основные функции в такой ФС нацелены на решение следующих задач:

- именование файлов;
- программный интерфейс для приложений;
- отображения логической модели файловой системы на физическую организацию хранилища данных;
- устойчивость файловой системы к сбоям питания, ошибкам аппаратных и программных средств.

2. *файловая система операционной системы OS/2, называемая HPFS (High-Performance File System - быстродействующая файловая система).*

Обеспечивает возможность иметь имя файла до 254 символов. Файлы, записанные на диск, имеют минимальную фрагментацию. Может работать с файлами, записанными в MS DOS;

К перечисленным выше задачам добавляется новая задача совместного доступа к файлу из нескольких процессов. Файл в этом случае является разделяемым ресурсом, а значит, файловая система должна решать весь комплекс проблем, связанных с такими ресурсами. В частности, в ФС должны быть предусмотрены средства блокировки файла и его частей, предотвращения гонок, исключение тупиков, согласование копий и т. п.

В многопользовательских системах появляется еще одна задача: защита файлов одного пользователя от несанкционированного доступа другого пользователя.

3. *файловая система операционной системы Windows 95*

Имеет уровневую структуру, что позволяет поддерживать одновременно несколько файловых систем. Старая файловая система MS-DOS поддерживается непосредственно, а файловые системы, разработанные не

фирмой *Microsoft*, поддерживаются с помощью специальных *модулей*. Имеется возможность использовать длинные (до 254 символов) имена файлов.

4. *файловые системы операционной системы Unix*

Они обеспечивают унифицированный способ доступа к файловым системам ввода-вывода.

Права доступа к файлу практически определяют права доступа к системе (владелец файла – пользователь, который его создал).

3 Почему изучение MS-DOS остается актуальным

Понятия, введенные для работы с файлами, организация их хранения и соглашения о файлах, используемые в MS-DOS, рациональны, естественны и остались практически неизменными. Они сохранили роль базового понятийного аппарата.

Ведущие сетевые ОС типа Unix, в отличие от Windows, исходно не имеют графического пользовательского интерфейса (GUI, Graphic User Interface). Более того, для системных сетевых разработок он не актуален. Принципы же построения команд и работы с системой команд одинаковы для MS-DOS и Unix. Освоив первые, создадим базу для освоения вторых.

Огромную роль играет концепция файловых оболочек (файловых менеджеров), наиболее известной из которых является Norton Commander для MS-DOS (1986 г.). Можно сказать, что ее появление соизмеримо с появлением самой MS-DOS и Windows. Идея и реализация оказались настолько жизнеспособными, что остаются не менее актуальными и поныне.

Такого рода оболочки существуют и разрабатываются и для современных ОС (NCW, Windows Commander, Far Manager и др.). Профессиональные программисты однозначно отдают им предпочтение перед проводником Windows. В итоге файловые оболочки (наследие MS-DOS) и специфические инструменты Windows сосуществуют, органично дополняя друг друга.

В профессиональной программистской деятельности часто возникает необходимость в запуске небольших информационных программ, объективно не требующих графического интерфейса (даже файлового менеджера). К таким программам относятся, например, ipconfig.exe (определение сетевого ip-адреса машины), arp.exe (назначение сетевых карт и адресов), at.exe (назначение задания для выполнения), ping.exe (проверка прохождения пакетов в сети) и т. п. В этом случае работа с простой командной строкой оказывается более рациональной.

3.1 Работа с файлами и каталогами. Основные команды

Смена текущего дисковода: набрать имя дисковода, который должен стать текущим, и затем двоеточие, например:

A:- переход на дисковод A;
B:- переход на дисковод B;
C:- переход на дисковод C.

После ввода команды нажать клавишу Enter.

Изменение текущего каталога: команда CD (Change Directory). Формат команды:

cd [дисковод:] путь

Если дисковод указан, то текущий каталог изменяется на этом дисководе, иначе – на текущем дисководе.

Просмотр каталога: команда DIR. Формат команды:

DIR [дисковод] [путь \] [имя файла][/P] [/W]

Можно использовать групповое имя файла (с маской *). Если имя файла не задано, то выводится все оглавление каталога.

Если не заданы дисковод или путь, подразумевается текущий дисковод и текущий каталог.

Для каждого файла команда DIR сообщает его имя, расширение имени, размер файла в байтах, дату и время создания или последнего обновления файла.

Подкаталоги отмечаются пометкой

В конце выдачи сообщается о размере свободного пространства на диске.

Параметр **/P** задает поэкранный вывод оглавления.

Создание каталога: команда md (Make Directory). Формат команды:

md [дисковод:] путь

Уничтожение каталога (пустого!): команда rd (Remove Directory).Формат команды:

rd [дисковод:] путь

Удаление файлов: команда del (delete). Формат команды:

del [дисковод **:**] [путь \] имя файла

В команде может быть указано групповое имя файла.

Если Вы захотите удалить все файлы из каталога командой del *.*, то операционная система выдаст запрос:

Are you sure (Y/N)? (Вы уверены?).

Для удаления файлов надо нажать «Y» и Enter, для отмены команды - нажать «N» и Enter.

Переименование файлов: команда ren(Rename). Формат команды:

ren [дисковод:][путь\]имя файла имя файла.

Первое имя файла в команде задает имя (имена) переименовываемых файлов, второе- новое имя (имена) файлов.

Переименовываются все файлы из заданного каталога, подходящие под шаблон, заданный в первом имени файла в команде. Если символы * и ? имеются во втором имени файла в команде, то символы имен файлов на соответствующих позициях не изменяются.

Вывод файла на экран: команда type. Формат команды:

type [дисковод:][путь \] имя файла

Вывод на экран можно приостановить нажатием комбинации Ctrl-S. Повторное нажатие Ctrl-S возобновляет вывод на экран. Закончить вывод можно, нажав Ctrl-C или Ctrl-Break.

Копирование файлов: команда COPY. Формат команды:

COPY [дисковод:][путь\]имя файла [дисковод :] [путь\]имя-файла
или
COPY [дисковод:][путь\]имя файла [дисковод:][путь].

Из каталога, указанного в первом параметре команды, копируются файлы, заданные именем в первом параметре команды. Дисковод и путь во втором параметре команды указывают каталог, в который копируются файлы.

Если во втором параметре имя файла отсутствует, то имена файлов при копировании не меняются. Если во втором параметре команды задано имя файла, то оно указывает новое имя копируемого файла.

В команде COPY вместо имен файлов можно использовать обозначение устройств, например: CON – консоль (клавиатура для ввода, монитор для вывода), PRN – принтер (только как выходной файл).

При вводе с клавиатуры конец файла задается комбинацией Ctrl-Z.

Задание

1 Изучить теорию, ответить на контрольные вопросы;

2 Продумать возможную структуру некоторого личного каталога (для определенности это может быть, например, часть каталога для хранения электронных материалов по курсу; виды материалов: лекции – отдельный файл по каждой теме; справочные материалы; лабораторные работы – отдельный файл по каждой работе).

3 Изобразить структуру своего каталога. Тщательно продумать имена каталогов и файлов, проставить их на схеме.

4 Описать сеанс по созданию дерева каталогов согласно схеме.

5 Создать в одном или нескольких каталогах произвольные текстовые файлы.

6 Скопировать в соответствующие каталоги какие-либо файлы по собственному усмотрению.

7 Просмотреть каталоги и файлы в них.

8 Внести, если необходимо, коррективы (например, удалить лишние файлы).

9 Получить справку по какой-либо из команд и записать ее в файл.

10 Просмотреть файл одного каталога из другого каталога.

11 Создать пакетный файл из 2-3 команд и запустить его.

12 Приветствуются любые осмысленные действия, направленные на освоение работы в DOS.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №13:

1. Что называется, файлом и папкой?
2. По каким правилам составляются имена файлов и папок?
3. Как записать путь к файлу?
4. Какие вы знаете файловые системы?
5. Что такое маски имен?
6. Почему MS-DOS остается актуальным?
7. Назовите основные команды для работы с файлами в MS-DOS.
8. Доступны ли файлы одного каталога из другого каталога?
9. Как просмотреть содержимое каталога? Файла?
10. Какова структура командной строки?
11. Как получить справку о команде?
12. Что такое символы подстановки в имени файла и как они используются? Привести пример.

Лабораторная работа №14
«Обмен данными с файлом. Блокировки файлов. Контроль доступа к файлам»

Цель работы:

1. Изучить теорию обмена данными с файлом;
2. Освоить технологию блокировки файлов;
3. Изучить современные программные средства для ограничения доступа к файлам.

1 Обмен данными с файлом

Для обмена данными с файлом (предварительно открытым) используются процедуры read и write. В том случае, когда необходимо явным образом указать, с какого байта файла необходимо читать или записывать данные используются дополнительные системные вызовы.

На основе системных вызовов ввода – вывода строятся более мощные библиотечные функции ввода – вывода, составляющие прикладной интерфейс ОС

2 Блокировки файлов

Блокировки файлов и отдельных записей в файлах являются средством синхронизации между работающими в кооперации процессами, пытающимися использовать один и тот же файл одновременно.

Процессы могут иметь соответствующие права доступа к файлу, но одновременное использование этих прав (в особенности права записи) может привести к некорректным результатам. Примером такой ситуации является одновременное редактирование одного и того же документа несколькими пользователями. Если доступ к файлу не управляется блокировками, то каждый пользователь, который имеет право записи в файл, работает со своей копией данных файла. Результат такого редактирования непредсказуем — он зависит от того, в какой последовательности записывали изменения в файл применяемые пользователями приложения-редакторы.

Многопользовательские операционные системы обычно поддерживают специальный системный вызов, позволяющий программисту установить и проверить блокировки на файл и его отдельные области. В UNIX такой системный вызов называется **fcntl**. В его аргументах указывается дескриптор файла, для которого нужно установить или проверить блокировки, тип операции (блокирование или проверка, блокирование доступа для чтения или для записи), а также область блокирования — смещение от начала файла и размер в байтах.

При проверке наличия блокировок, установленных другими процессами, вызов fcntl немедленно возвращает управление с сообщением результата. При установке блокировки можно задать два режима работы системного вызова: с переходом процесса в состояние ожидания в том случае, если блокировку установить невозможно (синхронный системный вызов), и с немедленным возвратом в такой ситуации с сообщением отрицательного результата (асинхронный вызов).

Запрошенная блокировка записи не может быть установлена в том случае, если другой процесс уже установил свою блокировку записи на тот же файл. То есть блокировка записи является исключительной. Блокировки чтения не являются исключительными и могут устанавливаться на файл в том случае, если их области действия не перекрываются. Если на какую-то область файла установлена блокировка чтения, то на эту область нельзя установить блокировку записи.

В UNIX существуют два режима действия блокировок — консультативный (advisory) и обязательный (mandatory). Основным рекомендуемым для использования режимом является консультативный. При нем операционная система не занимается блокированием операций с файлом, а только устанавливает признаки блокирования областей в структурах file, поддерживающих операции с файлами. Кооперирующиеся процессы обязательно должны проверять наличие блокировок на файл, чтобы синхронизировать свою работу. Если же блокировки установлены, но процесс не проверяет их, то операционная система не запрещает доступ процесса к файлу, когда процесс делает системные вызовы read или write.

В обязательном режиме запрет на выполнение операции с заблокированным файлом поддерживает операционная система, поэтому процесс в любом случае не получит доступа к такому файлу. Однако при работе в этом режиме операционная система тратит много усилий и времени на его поддержание, поэтому обычно он не рекомендуется.

3 Контроль доступа к файлам

Файлы – один из видов разделяемых ресурсов, доступ к которым ОС должна контролировать. Существуют и другие виды ресурсов, с которыми пользователи работают в режиме совместного использования: принтеры, модемы, графопостроители и т.п. Во всех этих случаях пользователи или процессы пытаются выполнить с разделяемым ресурсом определенные операции, а ОС должны решить, имеют ли пользователи на это право. Пользователи являются субъектами доступа, а разделяемые ресурсы – объектами. Пользователь осуществляет доступ к объектам не непосредственно, а с помощью прикладных процессов, которые запускаются от его имени.

Для каждого типа объекта существует набор операций, которые можно с ним выполнять. Система контроля доступа ОС должна предоставлять средства для задания прав пользователей по отношению к объектам дифференцированно по операциям.

В качестве субъектов доступа могут выступать как отдельные пользователи, так и группы пользователей. Объединение пользователей с одинаковыми правами в группу и задания прав доступа в целом для группы является одним из основных приемов администрирования в больших системах.

У каждого объекта доступа существует владелец. Владелец объекта имеет право выполнить с ним любые допустимые для данного объекта операции. Во многих ОС существует особый пользователь – администратор «superuser», который имеет все права по отношению к объектам системы, не обязательно являясь их владельцем. Эти права (полномочия) необходимы администратору для управления политикой доступа.

4 Программы для ограничения доступа к файлам и настройкам ОС

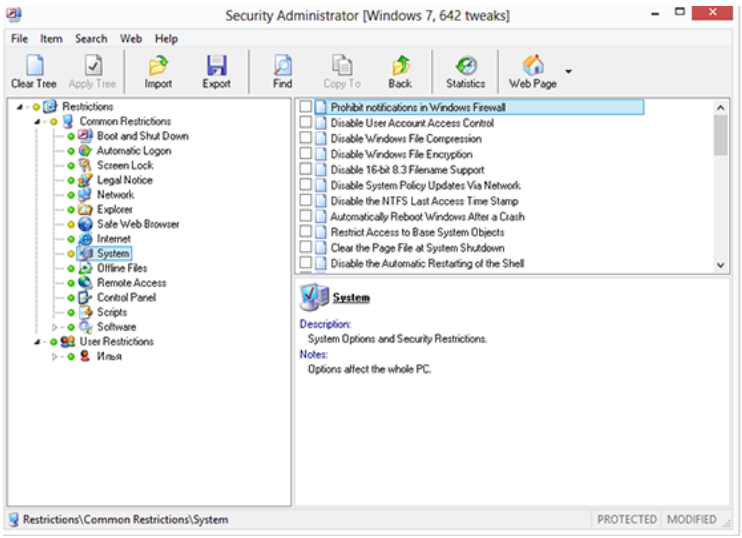
Несмотря на то, что в составе Windows есть инструменты для ограничения доступа, на практике оказывается, что они не слишком удобны, причем в самых заурядных ситуациях. Достаточно упомянуть такие простые примеры, как установка пароля на директорию или запрет на открытие Панели управления.

Можно отметить, что в Windows 8, по сравнению с предшествующей ей Windows 7, был усовершенствован родительский контроль. Сейчас с ним можно ознакомиться в разделе «Семейная безопасность» Панели управления. Фильтр имеет следующие возможности:

- Фильтрация посещения веб-сайтов
- Ограничения по времени
- Ограничения на Магазин Windows и игры
- Ограничения на приложения
- Просмотр статистики активности пользователей

Из перечисленного ясно, что даже эти функции помогут администратору компьютера решить многие частные моменты. Поэтому далее пойдет речь о небольших программах, которые позволяют ограничить доступ к информации и системным разделам в дополнение к стандартным инструментам управления ОС Windows.

4.1 Security Administrator



Программа Security Administrator напоминает типичный системный твикер, но только с акцентом на системную безопасность. Каждая из опций отвечает за определенное ограничение, поэтому общее древо настроек так и называется — «Restrictions». Оно делится на 2 раздела: общие (Common Restrictions) и пользовательские ограничения (User Restrictions).

В первой секции — параметры и подразделы, которые касаются всех пользователей системы. К ним относятся загрузка и вход в систему, сеть, Проводник, собственно Интернет, система, Панель управления и другие. Условно их можно разделить на ограничения онлайн- и офлайн-доступа, но разработчики особо сложной разбивки по вкладкам не рассмотрели. По сути, достаточно и того, что каждый «твик» имеет описание: какое влияние на безопасность оказывает та или иная опция.

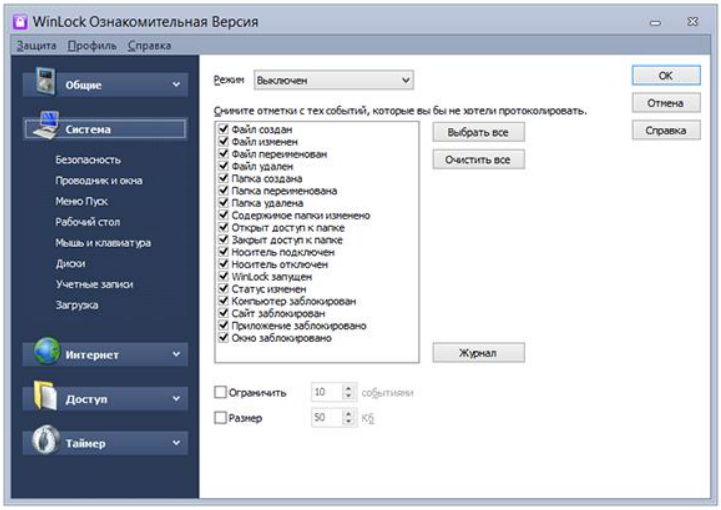
Во втором разделе, User Restrictions, можно настроить доступ для каждого пользователя Windows индивидуально. В список ограничений входят разделы Панели управления, элементы интерфейса, кнопки, горячие клавиши, съемные носители и др.

Предусмотрен экспорт настроек в отдельный файл, чтобы его можно было применить, например, на других системных конфигурациях. В программу встроен агент для слежения за активностью пользователей. Файлы журнала помогут администратору проследить потенциально опасные действия пользователя и принять соответствующие решения. Доступ к Security Administrator можно защитить паролем — в далее рассматриваемых программах это опция также имеется де факто.

Из недостатков — небольшой список программ, для которых можно применить ограничения: Media Player, MS Office и т. п. Популярных и потенциально опасных приложений намного больше. Усложняет работу отсутствие актуальной для Windows 8 версии и локализации — это как раз тот вариант, когда сложно обходиться без начальных знаний английского.

Таким образом, программа предназначена как для ограничения доступа, так и для гибкой настройки параметров безопасности ОС.

4.2 WinLock



В WinLock нет распределения настроек на общие и пользовательские, вместо этого имеются разделы «Общие», «Система», «Интернет». В сумме, возможностей меньше, чем предлагает Security Administrator, но такая логика делает работу с программой более удобной.

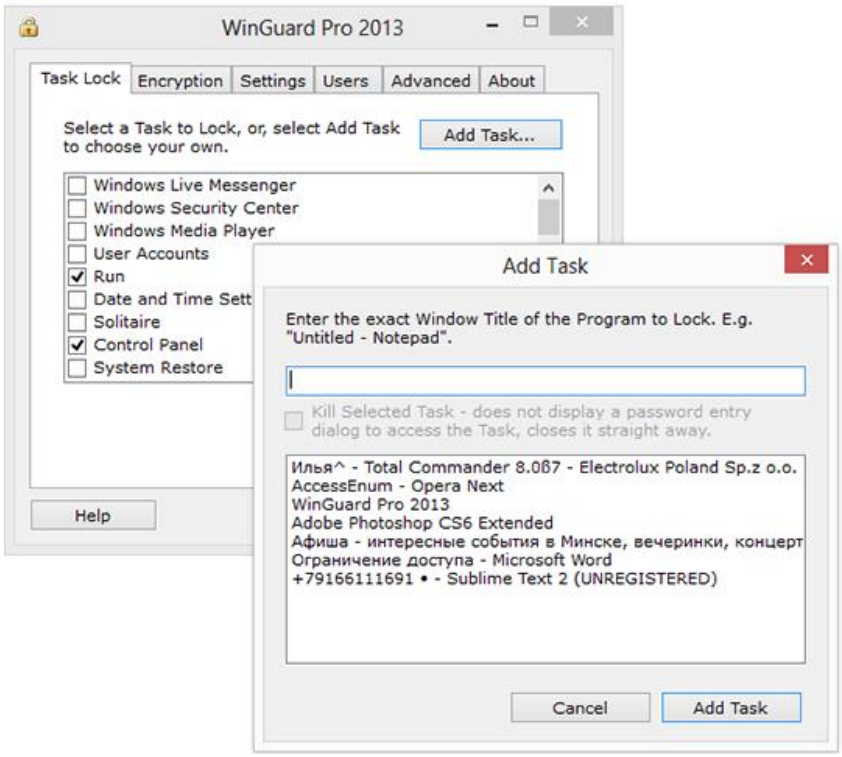
К системным настройкам относятся ограничения элементов Рабочего стола, Проводника, меню «Пуск» и им подобных. Также можно установить запрет на определеные горячие клавиши и всевозможные меню. Если интересуют только эти аспекты ограничений — ниже см. программу Deskman.

Ограничения функций Интернета представлены весьма поверхностно. Они заменяют один из компонентов Семейной безопасности: блокировка доступа к сайтам. Безусловно, любой файрволл по этой части будет оптимальным решением. Отсутствие же возможности хотя бы задать маску для веб-узлов делает данный раздел WinLock маловажным для опытного пользователя.

Кроме названных разделов, следует упомянуть «Доступ», где доступно управление приложенями. Любую программу легко занести в черный список по названию либо ручным добавлением.

В разделы «Файлы» и «Папки» можно поместить данные, которые необходимо скрыть от других пользователей. Пожалуй, не хватает парольной защиты доступа (для этого нужно обратиться к помощи других программ, см. ниже).

4.3 WinGuard



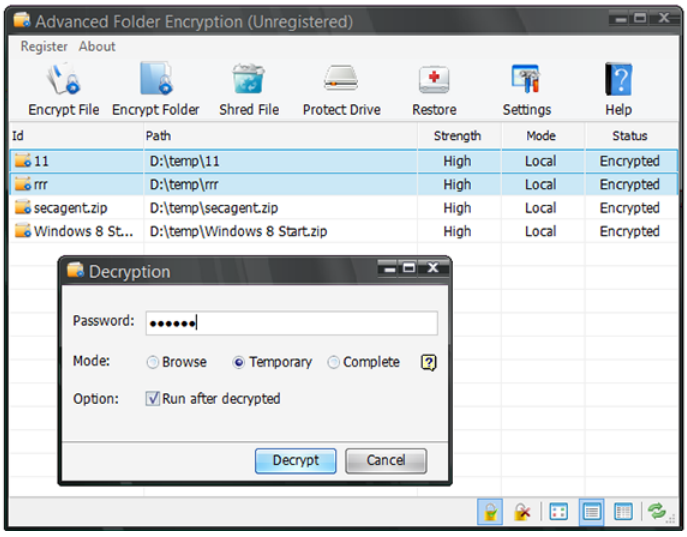
WinGuard можно использовать для блокировки приложений и разделов Windows, а также для шифрования данных. Программа распространяется в двух редакциях — бесплатной и Advanced. Функциональные отличия между ними невелики — несколько опций в одноименной вкладке «Advanced». Среди них отключение Internet Explorer, Проводника, процесса установки, записи файлов на USB.

Контроль над запуском приложений осуществляется во вкладке «Task Lock». Если нужной программы нет в списке, ее можно добавить самостоятельно, указав название в заголовке либо выбрав из списка открытых в данный момент приложений (аналогично WinLock). Если же сравнивать функцию блокировки с Security Administrator, в случае с WinGuard можно отключить ограничения для администраторского аккаунта. Однако настроить черный список приложений для каждого пользователя нельзя.

Шифрование доступно посредством раздела Encryption. Реализован пользовательский интерфейс неудобно: нельзя составить список для обработки, нет контекстного меню. Все, что нужно сделать — это указать директорию, которая будет являться и исходной, и конечной. Все содержащиеся файлы будут зашифрованы в 128-битный AES (Advanced Encryption Standard). Аналогичным образом производится расшифровка.

Таким образом, функциональность достаточно бедная, даже если взять во внимание платную версию.

4.4 Advanced Folder Encryption



Еще одна программа для AES-шифрования данных, и все же отличие от WinGuard весьма заметно.

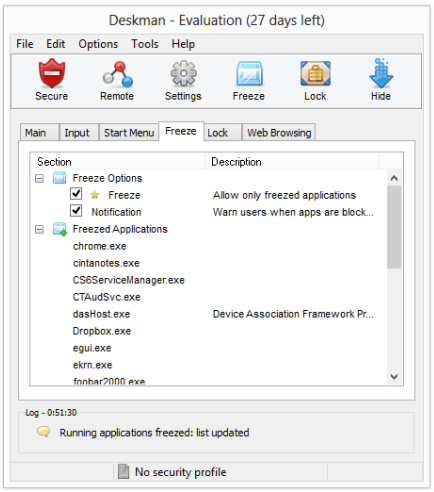
Во-первых, выбор файлов для шифрования происходит быстрее. Не нужно выбирать каждую папку по-отдельности, достаточно составить список директорий и файлов. При добавлении Advanced Folder Encryption требует выставить пароль для шифрования.

Вы, в программе нет возможности указать метод защиты, вместо этого позволяется выбрать метод Norman, High или Highest.

Второй удобный момент — шифрование через контекстное меню и расшифровка файлов одним кликом. Нужно понимать, что без установленной Advanced Folder Encryption данные не удастся просмотреть даже зная пароль. В этом отличие от архиваторов, которыми можно запаковать файлы в зашифрованный и всюду доступный exe-архив.

При выборе большого количества файлов для шифрования, как было замечено, не срабатывает кнопка отмены. Поэтому нужно быть осторожным, чтобы не получить результат в виде битого файла.

4.5 Deskman



Программа для ограничения доступа к элементам интерфейса и системным разделам. Пожалуй, тут ее уместно сравнить с Security Administrator за той разницей, что Deskman более сконцентрирован на Рабочем столе. Системные опции также присутствуют, но это, скорее, то, что не подошло в другие разделы: отключение кнопок перезагрузки, Панели управления и другие смешанные опции.

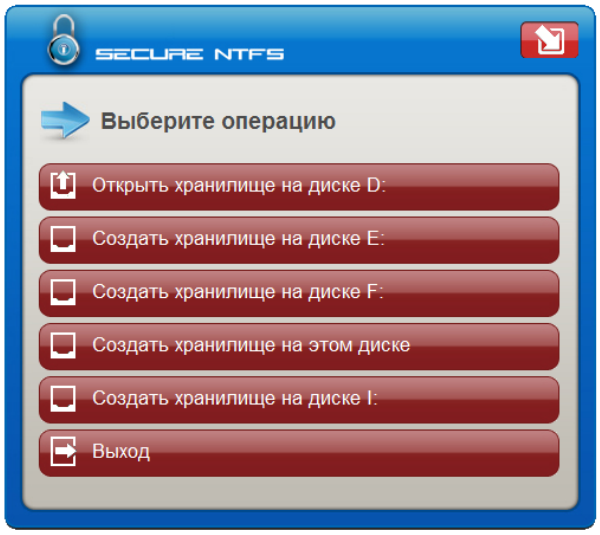
В разделе Input — отключение горячих клавиш, функциональных кнопок и функций мыши. Помимо имеющегося списка, можно определить сочетания клавиш самостоятельно.

Любопытна опция Freeze, которая доступна на панели инструментов. По ее нажатию формируется «белый список» из приложений, запущенных в данный момент. Соответственно, не входящие во whitelist программы недоступны до того момента, пока функция Freeze не будет отключена.

Еще одна возможность, связанная с уже онлайн, — защищенный веб-серфинг. Суть «защищенного» метода заключается в том, что доступны будут только те страницы, которые содержат в заголовке определенные ключевые слова. Эту функцию можно назвать разве что экспериментальной. Вдобавок, акцент делается на Internet Explorer, который, безусловно, является стандартным браузером, но явно неединственным.

Следует отметить удобное управление программой. Для применения всех установленных ограничений достаточно нажать кнопку «Secure» на панели, либо босс-клавишу для снятия ограничений. Второй момент — поддерживается удаленный доступ к программе посредством веб-интерфейса. После предварительной настройки он доступен по адресу http://localhost:2288/deskman в виде панели управления. Это позволяет следить за пользовательской активностью (просмотр журналов), запускать программы, перезагружать компьютер/выходить из системы — как на одной, так и на нескольких машинах.

4.6 Password for Drive (Secure NTFS)



Программа работает только с файловой системой NTFS и использует ее возможности для хранения информации в скрытой области.

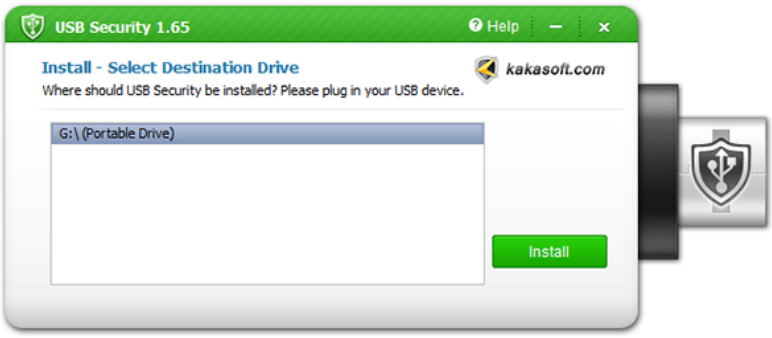
Для создания хранилища необходимо запустить Password for Drive с администраторскими правами и выбрать диск для создания хранилища. После этого файлы можно копировать в защищенную область с помощью

виртуального диска. Для доступа к данным с другого компьютера не требуются администраторские права.

В качестве хранилища можно использовать также съемный носитель. Для этого предварительно нужно отформатировать диск, например, штатными инструментами Windows, в NTFS и установить Password for Drive в редакции portable.

Программа не отличается интуитивным и удобным интерфейсом, фактически управление осуществляется минимальным набором кнопок — «Открыть»/»Удалить хранилище» и «Активировать диск». В демонстрационном режиме возможно лишь протестировать функциональность программы, так как количество открытий хранилища ограничено сотней.

4.7 USB Security



Программа предназначена для установки парольной данных на съемные носители.

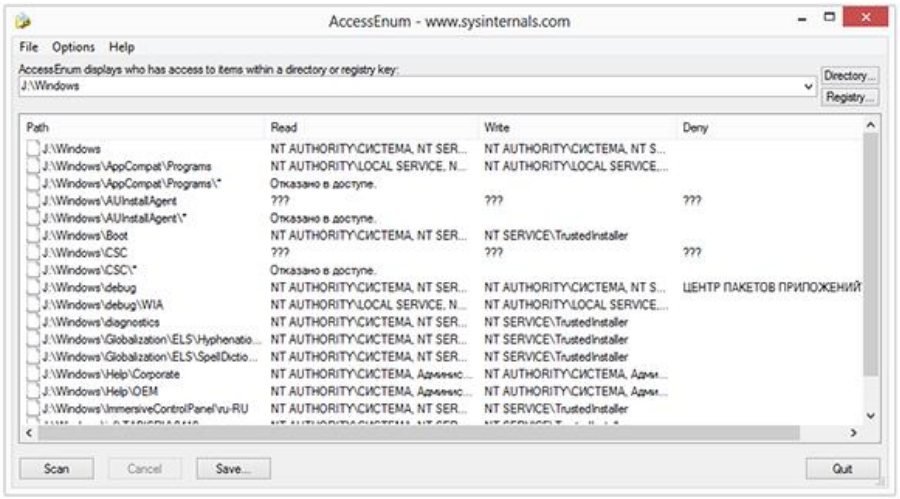
В отличие от Secure NTFS, диалог настройки значительно более интуитивен, благодаря мастеру настройки. Так, для применения защиты необходимо подсоединить к компьютеру устройство, выбрать его в списке и следовать мастеру установки. После данной процедуры пользователь получает в свое распоряжение диск, защищенный паролем. Для разблокировки достаточно запустить exe-файл в корне диска и ввести пароль.

Зашифрованный диск при открытии доступен как виртуальный диск, с которым можно производить те же операции, что и с исходным. Не стоит забывать, что на компьютерах, где запрещен запуск сторонних программ (не находящихся в «белом» списке), доступ к содержимому будет закрыт.

Также на сайте разработчиков можно скачать другие программы для защиты данных, в том числе:

- Shared Folder Protector — защита файлов внутри сети;
- Folder Protector — защита файлов на съемных носителях.

4.8 AccessEnum



Небольшая утилита, которая позволяет контролировать доступ пользователей к Реестру и файлам, находить уязвимости в выданных правах. Другими словами, программа будет полезна в том случае, если права доступа выставляются средствами Windows.

Удобство утилиты состоит в том, что ОС попросту не предоставляет средств для просмотра прав доступа к директориям в виде детального списка. Кроме файлов, также можно проверить доступ к веткам реестра.

Для проверки прав доступа необходимо указать директорию или раздел Реестра для сканирования и начать процесс сканирования. Результаты отображаются в виде колонок «Read»/»Write»/»Deny», соответствующих адресам. В свойства каждого из элементов списка можно зайти через контекстное меню.

Программа работает под всеми ОС семейства Windows NT.

Резюме

Утилиты, рассмотренные в обзоре, можно использовать в дополнение к базовому инструментарию Windows, комплексно и в довесок друг к другу. Их нельзя отнести в категорию «родительский контроль»: некоторые функции в чем-то схожи, но по большей части не совпадают.

Security Administrator и **WinLock** — утилиты-твикеры настроек безопасности, которые также можно использовать для отключения элементов Рабочего стола, системных разделов и др. В WinLock, кроме того, встроен интернет-фильтр и есть возможность блокировки файлов и папок.

В функции **Deskman** входит ограничение доступа к интерфейсу (превосходящее по гибкости **Security Administrator** и **WinLock**), управление запуском приложений, интернет-фильтр.

WinGuard сочетает в себе функции шифровальщика и ограничителя доступа к программам и устройствам. **Advanced Folder Encryption** имеет смысл рассматривать как замену WinGuard по части шифрования.

Password for Drive и **USB Security** ограничивают доступ к данным на съемных носителях.

AccessEnum — бесплатная удобная оболочка для проверки прав доступа пользователей к файлам и Реестру.

Задание

13 Изучить теорию и предлагаемые программные средства;

14 Ограничить доступ к файлам с помощью предлагаемого ПО

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №14:

1. Как осуществляется контроль доступа над файлами?
2. Расскажите о синхронизации процессов.
3. Назовите программы для ограничения доступа к файлам и настройкам ОС;

Лабораторная работа №15
«Реализация отказоустойчивости на основе резервирования. Механизм и организация контроля доступа»

Цель работы:

1. Научиться создавать резервную копию операционной системы;
2. Освоить механизмы контроля доступом;
3. Научиться разграничивать права пользователей.

1 Реализация отказоустойчивости на основе резервирования

Отказоустойчивость – свойство архитектуры КС, позволяющее пользователю или функциональной программе продолжать работу и тогда, когда в аппаратных или программных средствах возникают отказы.

По способу реализации отказоустойчивость подразделяется на активную и пассивную.

Активная отказоустойчивость базируется на процессах обнаружения отказа, локализации отказа и реконфигурации системы. Отказы обнаруживается при помощи средств контроля, локализируется с помощью средств диагностирования и устраняются автоматической реконфигурацией системы.

Пассивная отказоустойчивость заключается в свойстве системы не потерять свои функциональные свойства в случае отказа отдельных элементов системы. Пассивная отказоустойчивость связана с увеличением количества аппаратуры в несколько раз. Пассивная отказоустойчивость применяется в случае особо ответственных КС, когда не допустимы даже кратковременные перерывы в обработке КС, а также для обеспечения отказоустойчивости его важнейших блоков или устройств.

Применение активной отказоустойчивости характеризуется более экономными расходом аппаратных средств, чем применение пассивной отказоустойчивости. Однако оно связано с некоторыми потерями времени при восстановлении работы системы после отказа, а также потерями некоторой части данных. Активная отказоустойчивость реализуема только в многопроцессорных системах (с общей памятью, общей шиной, матричной, кольцевой или другой структурой). В то же время применение пассивной отказоустойчивости гарантирует практически безостановочную работу КС и сохранение всей информации. Эти обстоятельства и определяют области применения активной и пассивной отказоустойчивости.

Введение отказоустойчивости является одним из методов повышения надежности КС. Вопрос о построении и применении отказоустойчивых систем возникает тогда, когда другие пути повышения надежности не могут обеспечить требуемого уровня надежности по техническим причинам, или тогда, когда они отказываются экономически не оправданными.

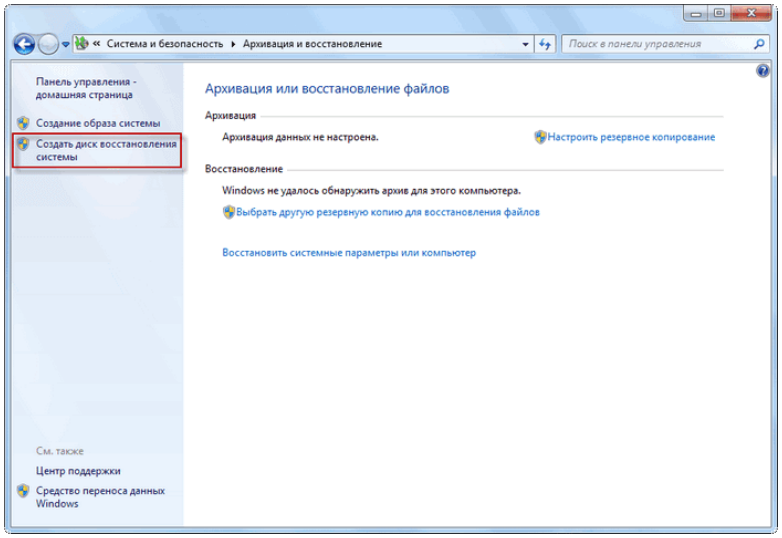
1.1 Создание резервной копии системы

1.1.1 Создание диска восстановления системы

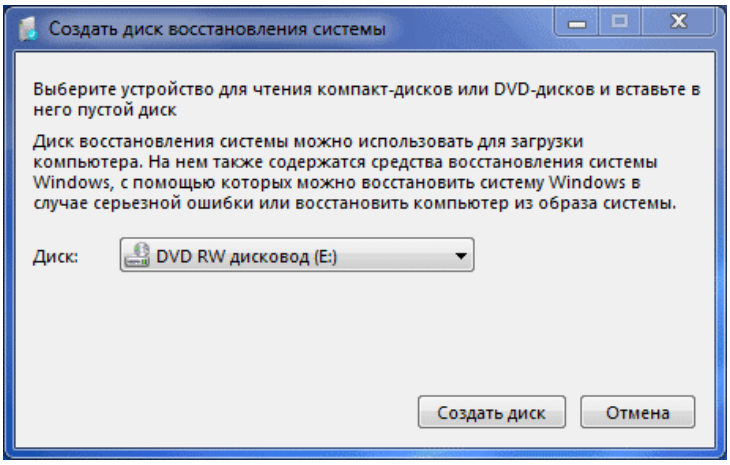
Для тех пользователей, которые имеют предустановленные операционные системы на своем компьютере, создать загрузочный диск восстановления системы просто необходимо. Производители компьютеров, сейчас часто

не комплектуют их установочным диском Windows. В этом случае, аварийный диск восстановления системы, поможет загрузить компьютер для его восстановления, если загрузиться другим способом не представляется возможным.

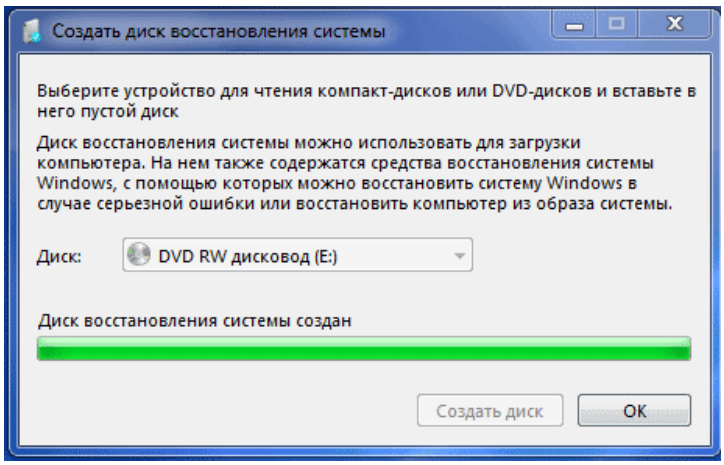
В окне «Архивация и восстановление» необходимо нажать на пункт «Создать диск восстановления системы», для создания аварийного диска восстановления системы. В этом случае, у вас будет аварийный компакт диск для восстановления системы, в случае возникновения серьезных неполадок с операционной системой.



После нажатия на пункт «Создать диск восстановления системы», открывается окно «Создать диск восстановления». Для создания диска восстановления системы необходимо вставить в устройство для чтения оптических дисков пустой CD или DVD диск, а затем нажать на кнопку «Создать диск».



Далее происходит процесс создания диска восстановления системы. После завершения создания аварийного диска восстановления, нужно нажать на кнопку «ОК». Аварийный диск восстановления Windows 7 занимает объем около 150 МБ.



Теперь у вас будет возможность получить доступ к вариантам восстановления системы, используя загрузочный диск восстановления, если невозможно будет загрузить компьютер другими способами.

Для загрузки компьютера с аварийного или установочного диска, вам необходимо будет выбрать в BIOS приоритет загрузки с устройства для чтения дисков CD/DVD, а в случае использования загрузочной флешки с диска USB, к которому подключена такая загрузочная флешка.

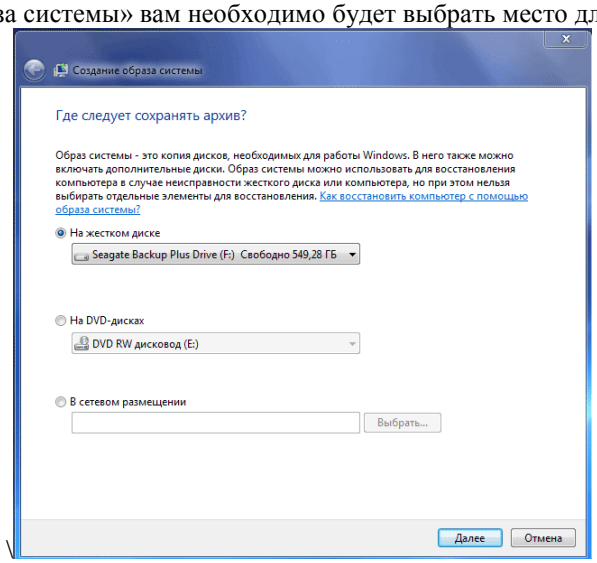
Если при создании диска восстановления вам предложат вставить установочный диск с операционной системой Windows 7, то это значит, что необходимые файлы для создания диска восстановления системы не были обнаружены. В этом случае вам необходимо будет вставить в оптический привод компьютера установочный диск DVD с операционной системой Windows 7.

С помощью установочного диска Windows 7 или аварийного диска для восстановления системы, вы сможете загрузиться на своем компьютере, и будете иметь доступ ко всем параметрам восстановления операционной системы.

1.1.2 Создание образа системы

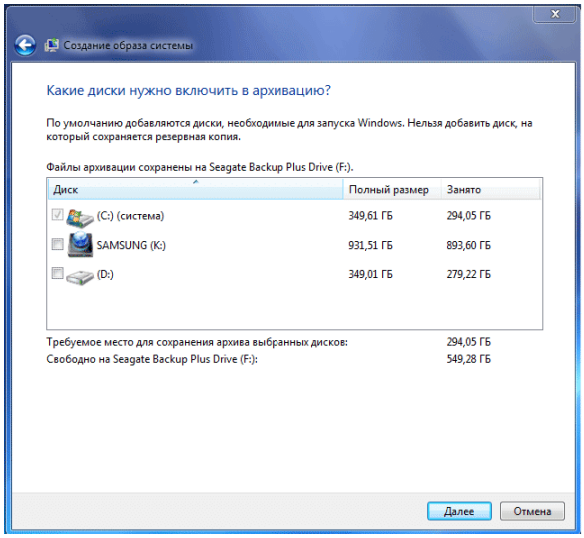
Если в окне «Архивация и восстановление» выбрать пункт «Создание образа системы», то тогда будет создан образ системы для ее восстановления, который включает в себя копии дисков, необходимых для работы операционной системы Windows. В образ системы можно включать дополнительные диски и использовать его для восстановления, в случае неполадок. При этом нельзя использовать отдельные элементы для восстановления.

В окне «Создание образа системы» вам необходимо будет выбрать место для хранения резервной копии.



После этого следует нажать на кнопку «Далее».

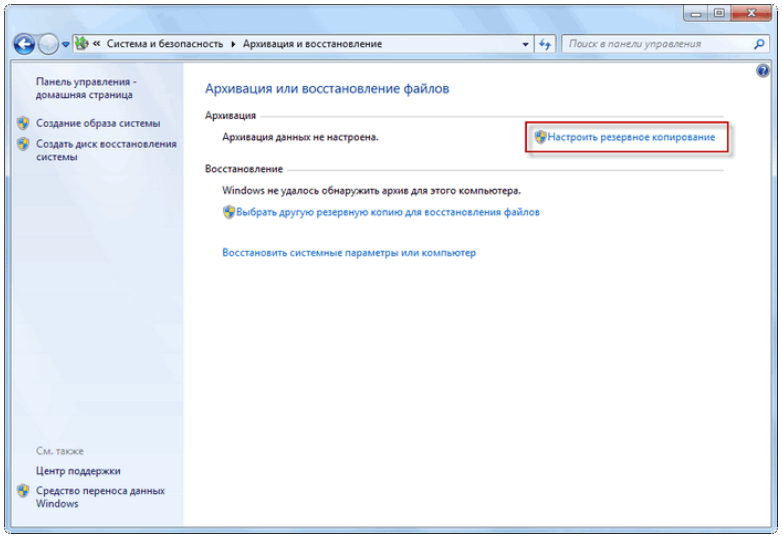
В окне «Создание образа системы» вам нужно будет выбрать диски, которые нужно включить в архивацию. При этом, нельзя будет добавить тот диск, на котором будет сохранена резервная копия. Затем нажимаете на кнопку «Далее».



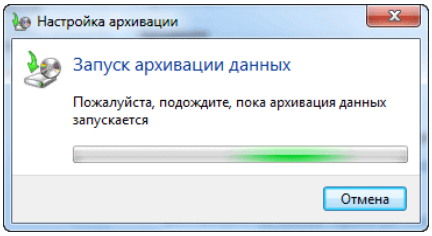
В новом окне «Создание образа системы» вам необходимо будет подтвердить параметры архивации и расположение резервной копии. После этого, нажимаете на кнопку «Архивировать».

1.1.3 Архивация в Windows 7

Теперь перейдем к настройкам архивации и резервного копирования. В окне «Архивация и восстановление файлов» нужно нажать на пункт «Настроить резервное копирование».



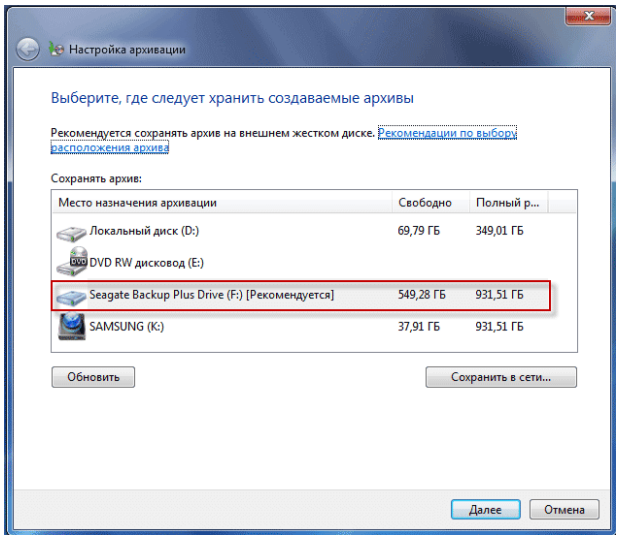
Далее открывается окно «Настройка архивации». Вам придется подождать некоторое время, пока производится запуск архивации данных. Вы увидите в окне надпись «Запуск архивации данных», при этом сама архивация еще не происходит.



Затем открывается окно «Настройка архивации». В этом окне необходимо выбрать место для хранения резервной копии архива.

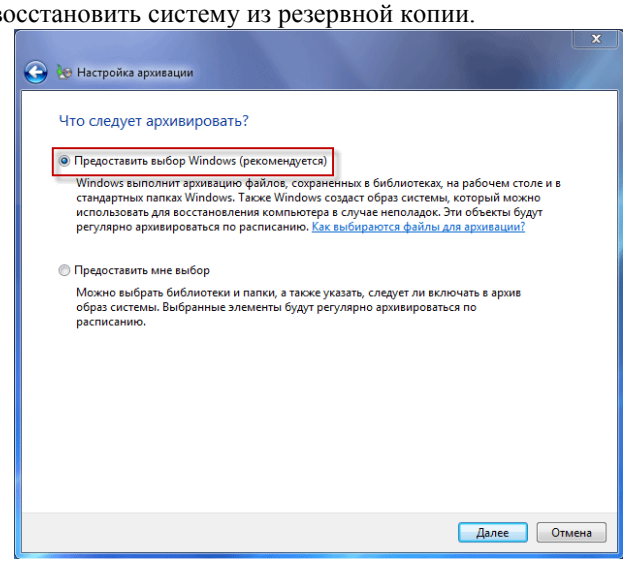
Лучшим вариантом для хранения создаваемой резервной копии системы будет внешний жесткий диск. Потому что, если вы сохраните резервные копии на другом разделе жесткого диска вашего компьютера, то в том случае, если выйдет из строя жесткий диск, то тогда будут навсегда потеряны операционная система, ваши данные и созданные резервные копии для восстановления. Данные, при этом, физически находятся на одном жестком диске, они только расположены на разных логических дисках.

На этом изображении видно, что система сама подсказала мне место для хранения — внешний жесткий диск, с достаточным местом для создания резервной копии.



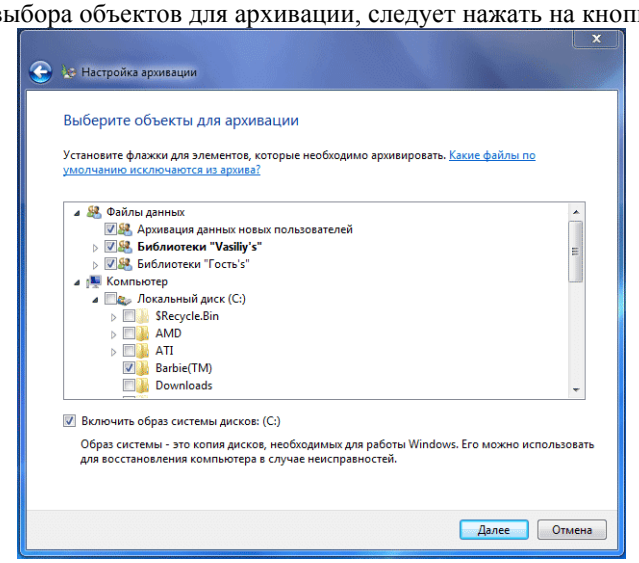
В окне «Настройки архивации» вам нужно будет выбрать, что следует архивировать.

В случае предоставления выбора Windows, операционная система выполнит архивацию файлов, сохраненных в стандартных папках, в библиотеках, на Рабочем столе, а также создаст образ системы, который позволит восстановить компьютер в случае неполадок. Все ваши данные и настройки будут архивированы, и вы сможете при необходимости восстановить систему из резервной копии.

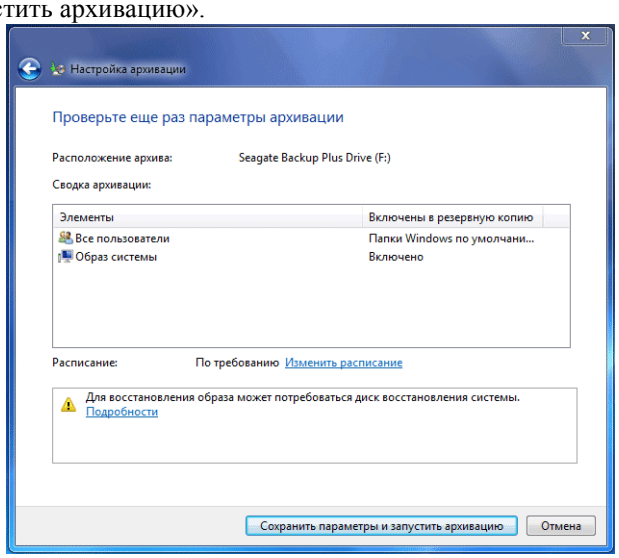


В этом случае можно будет выбрать конкретные диски, выбрать отдельные папки, находящиеся на выбранном диске. Обязательно поставьте флажок напротив пункта «Включить образ системы дисков: (C:)», для того, чтобы у вас была возможность в случае необходимости восстановить систему из созданной резервной копии.

Выбранные вами данные будут архивироваться по расписанию, если вы не будете запускать архивацию вручную. После завершения выбора объектов для архивации, следует нажать на кнопку «Далее».

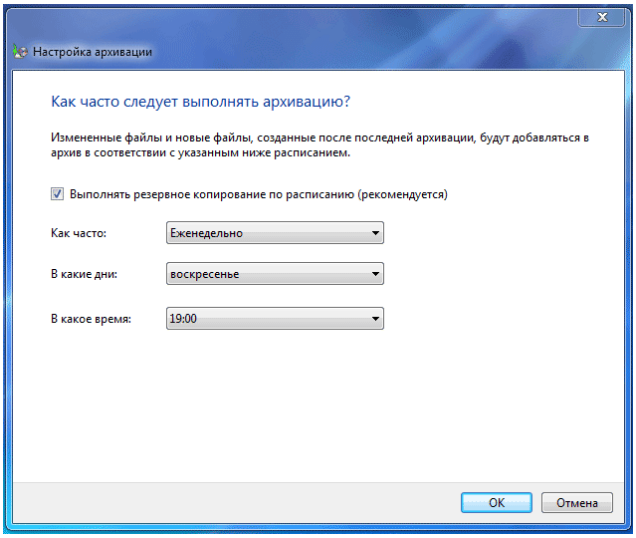


В окне настройка архивации вам еще раз нужно проверить настройки архивации, а затем нажать на кнопку «Сохранить параметры и запустить архивацию».

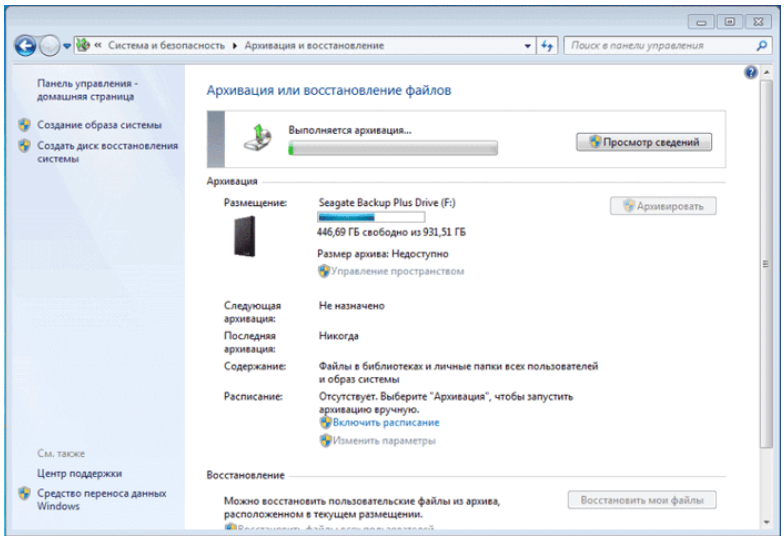


Перед запуском начала архивации, вы можете нажать на ссылку «Изменить расписание», для выбора своего расписания или выполнять резервное копирование по требованию.

Если вы будете выполнять архивацию не по расписанию, а вручную, когда это вам необходимо, то тогда вам нужно будет снять флажок напротив пункта «Выполнять архивацию по расписанию (рекомендуется)», а затем нажать на кнопку «ОК».



Далее вам необходимо будет запустить начало архивации. Время создания резервной копии будет зависеть от объема дисков и файлов, которые будут архивированы, а также от мощности вашего компьютера. Резервное копирование, запускаемое повторно будет происходить быстрее, потому что будут перезаписываться только измененные файлы после предыдущего резервного копирования.



Следует помнить, что в случае настройки резервного копирования по расписанию, внешний жесткий диск должен быть в это время подключен к вашему компьютеру. Если вы будете делать резервное копирование без определенного расписания, то оптимальным вариантом будет проведение архивации примерно раз месяц. В этом случае у вас будут сохраняться настройки системы, которые вы сделали относительно недавно.

Важные данные, которые вы часто изменяете, архивировать потребуется чаще, чем раз в месяц, чтобы всегда иметь актуальную резервную версию для восстановления.

Как отключить архивацию в Windows 7

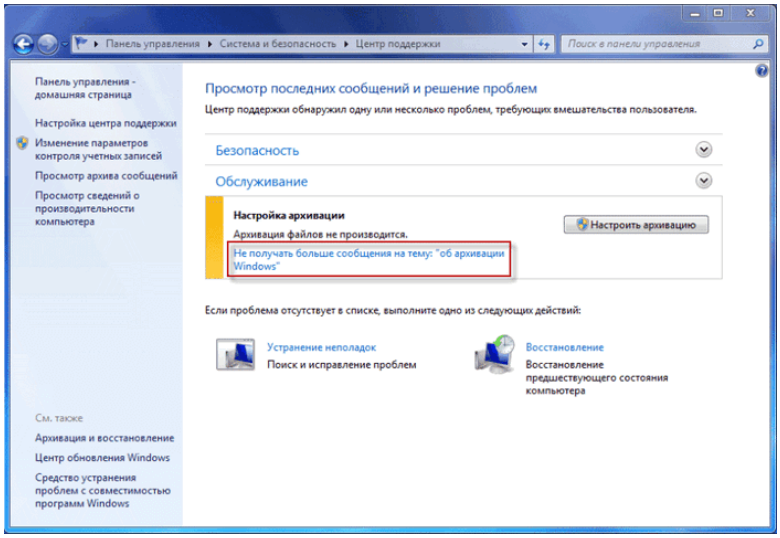
Иногда, бывает необходимо отключить архивацию, в том случае, если вы настроили создание резервных копий по расписанию, а на том диске, где вы сохраняете резервные копии, заканчивается свободное место. В этом случае, вам нужно будет отключить создание резервных копий по расписанию.

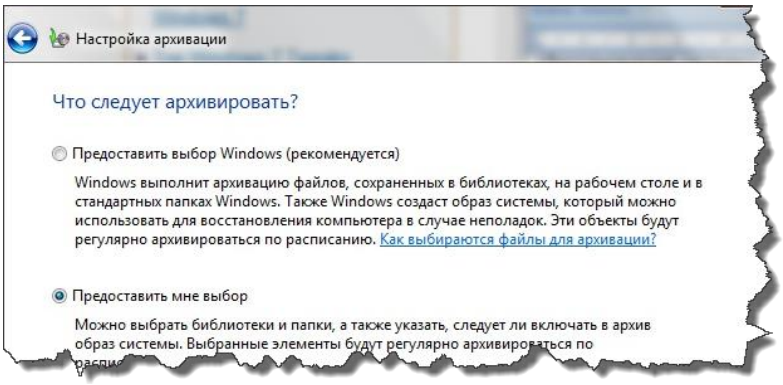
Для этого, необходимо будет войти в меню «Пуск» => «Панель управления» => «Администрирование» => «Службы». В окне «Службы» нужно найти пункт «Служба модуля архивации на уровне блоков (Служба WBENGINE используется для выполнения операций архивации и восстановления)».

Для отключения автоматической архивации необходимо изменить тип запуска службы с «Автоматически» на «Вручную». Для этого, нужно кликнуть правой кнопкой мыши по пункту «Автоматически и выбрать в контекстном меню пункт «Свойства».

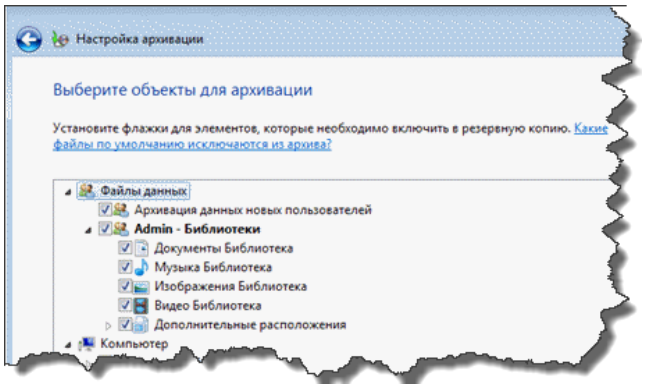
В открывшемся окне «Свойства: Служба модуля архивации на уровне блоков», во вкладке «Общие», в пункте «Тип запуска» выбрать «Вручную» и нажать на кнопку «ОК». Далее потребуется перезагрузить компьютер. Теперь запускать резервное копирование вы можете по своему усмотрению вручную.

Если вас раздражают время от времени появляющиеся сообщения о необходимости настройки архивации из Панели уведомлений (трея), то тогда такие сообщения можно будет отключить. Для этого нужно войти в меню «Пуск» => «Панель управления» => «Центр поддержки». В окне «Центр поддержки», в поле «Обслуживание», в пункте «Настройка архивации» необходимо нажать на ссылку «Не получать больше сообщений на тему: «об архивации Windows»».

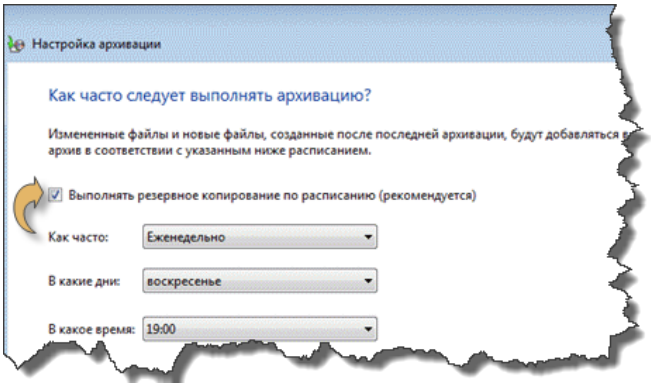




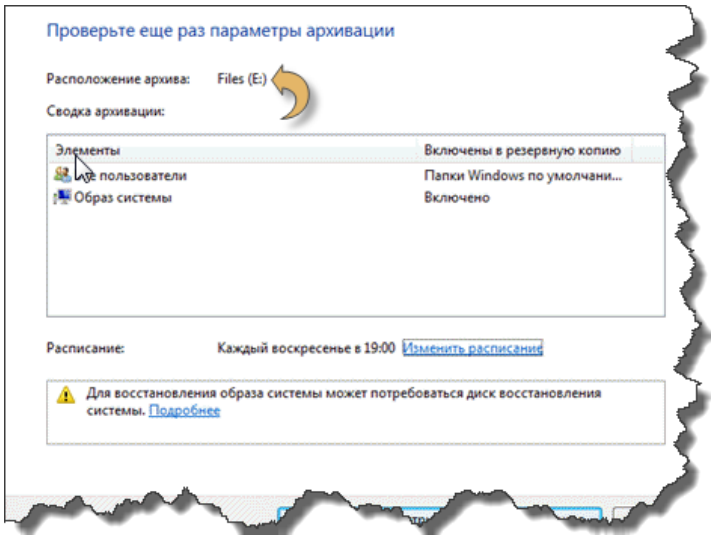
В выборе системы резервная копия виндовс 7 будет включать архивацию всего диска. Вы можете самостоятельно покопаться в недрах своего компьютера и выбрать только то, что хотите иметь в резервной копии.



Но если вы считаете, что обои рабочего стола сохранить гораздо важнее системных файлов, доверьте создание архива операционной системе Windows и не вносите в настройки никаких изменений. Архив можно сделать единожды, а можно задать расписание архивации.



Проверьте все установки, и если вас все устраивает, жмите «Сохранить параметры и запустить архивацию». После окончания операции на выбранном носителе вы найдёте резервную копию windows 7.



Установленные вами параметры расписания сохранятся в планировщике заданий. И при следующем запуске архивации ваше присутствие не потребуется. Windows 7 при сохранении архива включает в него дату архивации. В дальнейшем вам не составит труда удалить старые архивы, освободив тем самым пространство под новые копии.

Заключение

С помощью встроенного средства операционной системы — архивация Windows 7, вы сможете делать резервное копирование Windows, всего содержимого вашего жесткого диска или делать копии отдельных дисков, файлов и папок.

В случае серьезного сбоя вашего компьютера, вы сможете восстановить систему и все ваши данные из резервной копии. Архивация позволяет вам делать резервное копирование не только вручную, но и по выбранному вами расписанию.

2 Механизм и организация контроля доступа

Каждый пользователь и каждая группа пользователей имеют символическое имя и уникальный числовой идентификатор. Все идентификационные данные в т. ч. имена и идентификаторы пользователя и групп, параметры пользователя, сведения о его вхождении в группы, хранятся в специальном файле или в БД, входящем(ей) в состав ОС.

Механизм контроля доступа порождает процесс-оболочку, который поддерживает диалог с пользователем и запускает для него другие процессы.

При получении от пользователя имени и пароля процесс-оболочка находит по ним числовые идентификаторы и эти идентификаторы связываются с каждым процессом, запущенным оболочкой для данного пользователя, т.е. процесс выступает от имени данного пользователя.

Любой порожденный процесс наследует ID-ры пользователя и групп от процесса-родителя. Определить права доступа к ресурсу – это значит определить для каждого пользователя набор операций, который ему разрешено применить к данному ресурсу.

В разных ОС для одних и тех же типов ресурсов может быть определен свой список дифференцируемых операций доступа.

Практически во всех ОС матрица прав доступа хранится по частям, т.е. для каждого файла или каталога создается список управления доступом Access Control List (ACL) в котором описываются права на выполнение операций пользователей и групп по отношению к этому файлу или каталогу.

Список управления доступом (ACL) является частью характеристик файла или каталога и хранится в собственной области диска.

Обобщенно формат списка управления доступом можно представить в виде набора идентификаторов пользователей и групп пользователей, в котором для каждого идентификатора указывается набор разрешенных операций над объектом.

ACL состоит из элементного управления доступом – Access Control Element (ALE). При этом каждый элемент соответствует одному идентификатору. Список ACL с добавленным к нему ID-ром владельца называют характеристиками безопасности.

2.1 Создание учетной записи и разграничение прав пользователя

Операционная система Windows 7 имеет три типа учетных записей и все они предоставляют пользователям разные права по управлению операционной системой.

Отступление: Если нужно узнать, как создать учётную запись в Windows 8! Зачем нужна учётная запись Майкрософт? В чём разница между локальной учётной записью Windows 8 и учётной записью Майкрософт. Читайте нашу новую статью!

Моя статья была бы неполной, если бы я не сказал, зачем вообще создавать учётную запись. Во первых, *создать учётную запись Windows 7* очень просто, она сможет уберечь вашу систему от заражения вредоносной программой. К примеру, у нас есть статья о том, как учётная запись может помочь вам в борьбе с вирусом или баннером вымогателем, можете прочитать "Как избавиться от баннера". Ну а если вашим компьютером пользуются несколько человек, то создание каждому пользователю своей учётной записи, соответственно уровню подготовки пользования компьютером, намного продлит жизнь операционной системе в целом.

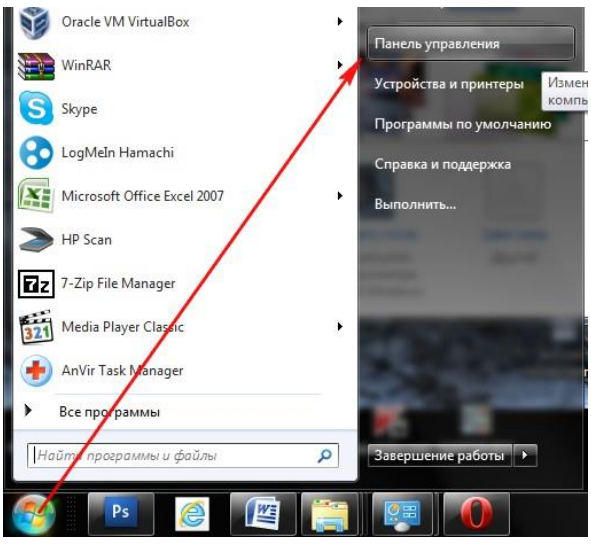
Какие бывают типы учётных записей

1. **Обычный доступ** – это очень хороший тип учётной записи для человека, который только-только начал постигать компьютерный мир, то есть для начинающего. Работая под учётной записью пользователя с обычным доступом, вы сможете устанавливать некоторые программы, а вот антивирусную программу установить не сможете, для этого вам понадобится знать пароль администратора компьютера или войти в операционную систему под учётной записью администратора.
2. **Администратор** – опытный пилот, берущий на себя полное управление современным самолётом под названием Windows и понимающий всю ответственность за это. Пользователь с правами администратора может совершать все фигуры высшего пилотажа, включая сложнейшую «Кобру». Он может изменять любые настройки операционной системы, включая редактирование критически важных значений реестра и редактирование настроек влияющих на всех пользователей в системе. Так же он несёт ответственность за других участников полёта (других пользователей ПК).
3. **Гость** – изначально встроенная учетная запись, которую не нужно создавать. Применяют для пьяных гостей, желающих оттянуться на вашем компьютере во время гулянки. А если серьёзно, то она специально придумана для временного доступа к компьютеру и сильно ограничена в правах. Работая под этой учётной записью, вы никогда ничего не испортите в вашей операционной системе, но она сильно понизит ваш уровень безопасности, поэтому советую вам её включать только в случае необходимости.

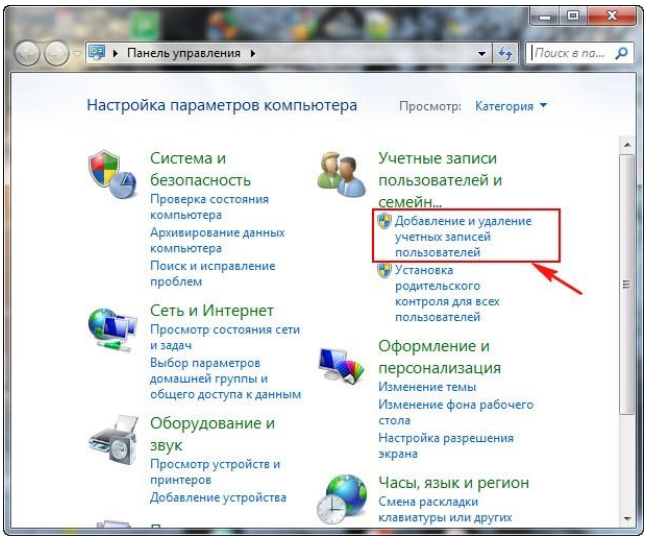
2.1.1 Создание учётной записи Windows 7

Создать учётную запись Windows 7 очень просто, но сделать это нужно из другой учётной записи, имеющей права администратора. Практически одинаково создаются учётные записи пользователей с Обычным доступом и доступом Администратора.

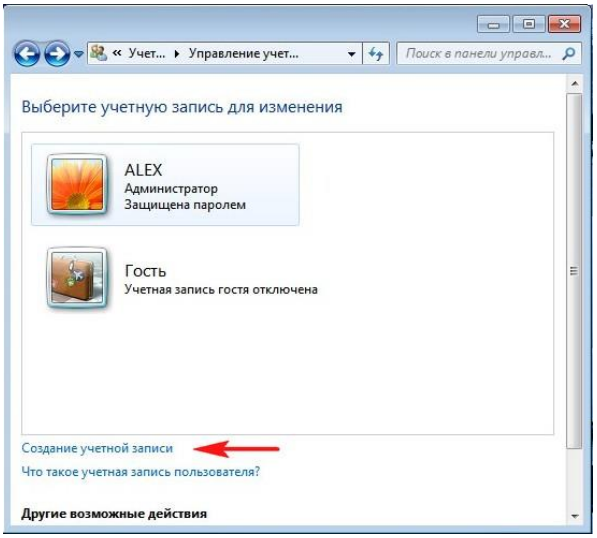
Для примера создадим учётную запись с Обычным доступом. Откроем меню Пуск->Панель управления->



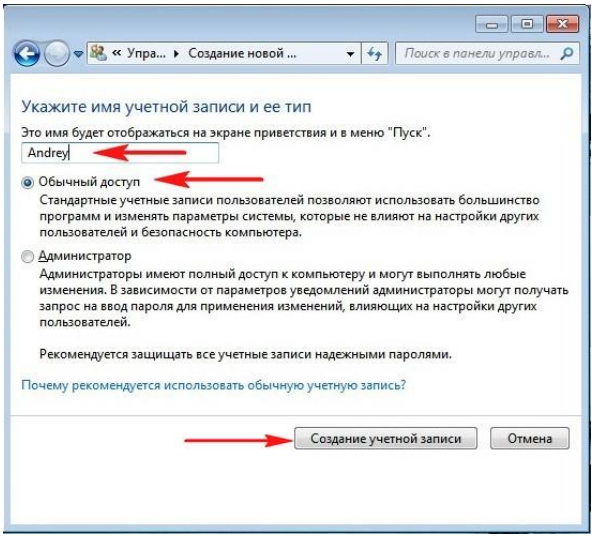
Добавление и удаление учётных записей пользователей



->Создание учётной записи.

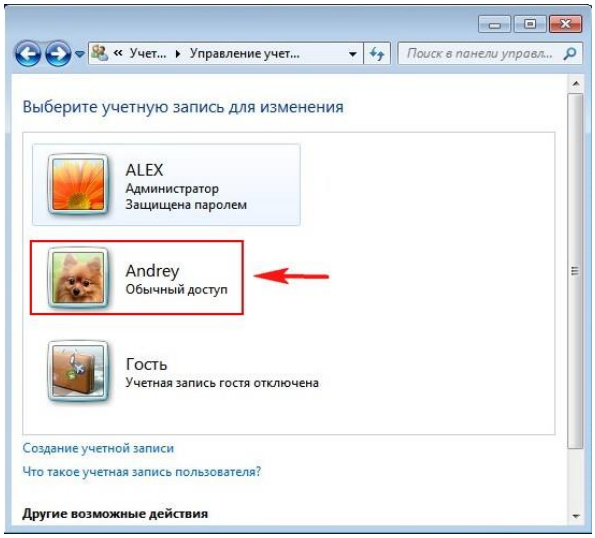


Отмечаем пункт **Обычный доступ**, далее вводим имя нашей учётной записи, к примеру Andrey и жмём на кнопку «Создание учётной записи».

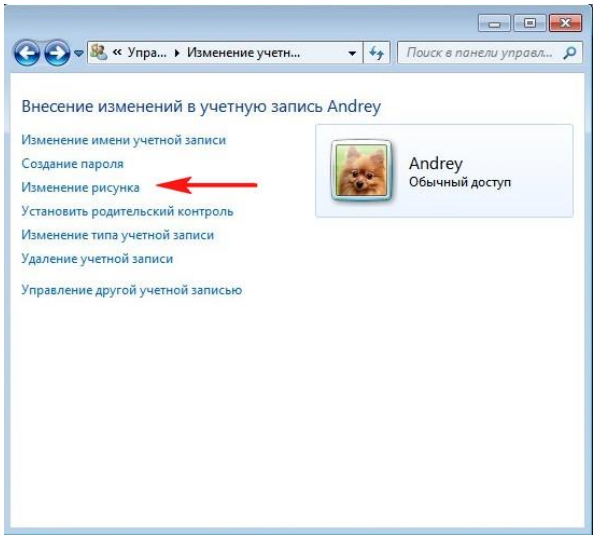


2.1.2 Изменение рисунка учётной записи

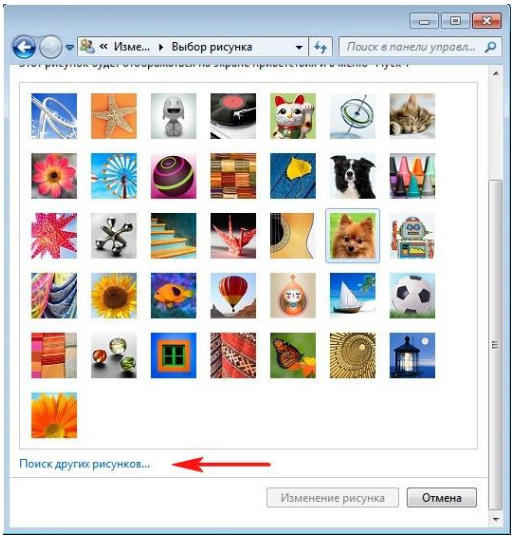
Всё, наша учётная запись Andrey создана. Изменяем рисунок учётной записи. Щёлкаем левой мышью на учётной записи Andrey



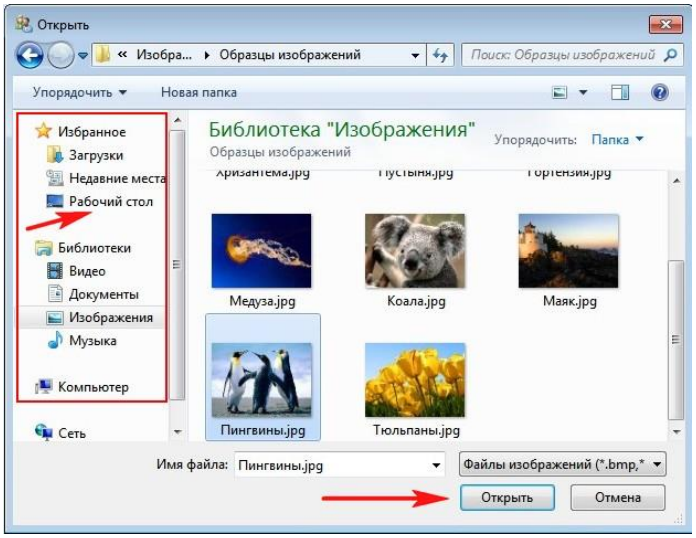
и нажимаем Изменение рисунка.



Если ни один из предложенных системой рисунков вам не нравится, значит жмём кнопку «Поиск других рисунков»



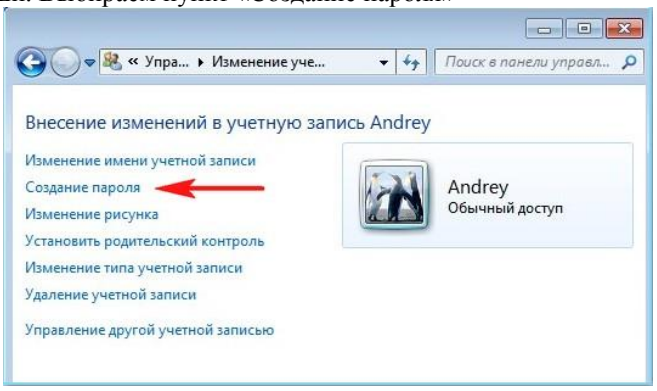
открывается «Библиотека Изображения» и проводник, в котором вы можете найти все картинки, находящиеся на вашем компьютере, например на рабочем столе.



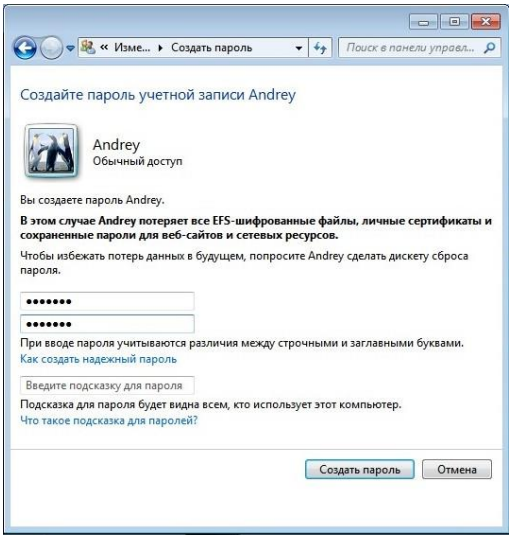
Выбирайте нужную вам и жмите Открыть. Всё, нужная картинка назначена.

2.1.3 Назначение пароля учётной записи

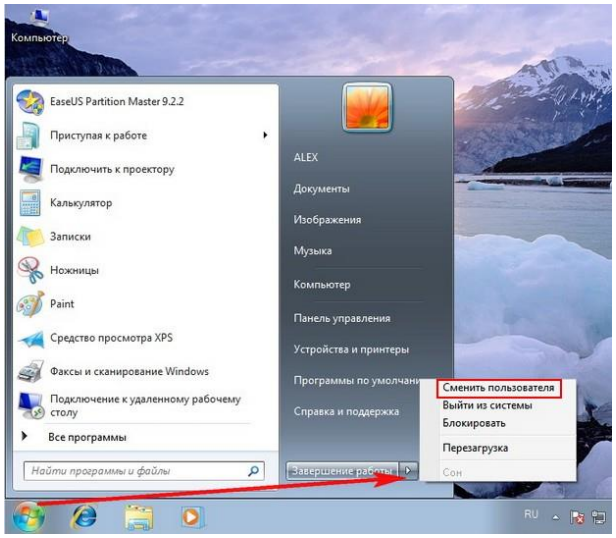
Теперь назначим учётной записи пароль, сделать это нужно обязательно. Щёлкаем на учётной записи Andrey двойным щелчком левой мыши. Выбираем пункт «Создание пароля»



Назначаем пароль и жмём «Создать пароль».



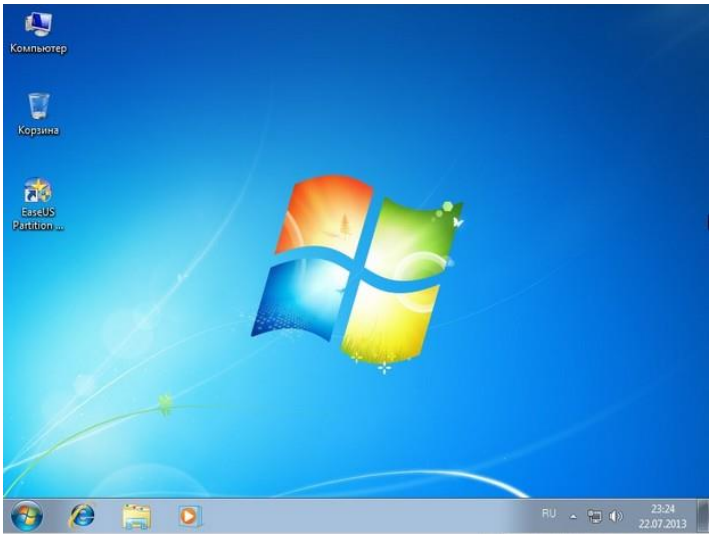
Чтобы войти в только что созданную учётную запись, необязательно перезагружать компьютер. Выбираем Пуск->Сменить пользователя.



Щёлкаем мышью на нашей учётной записи,

вводим пароль

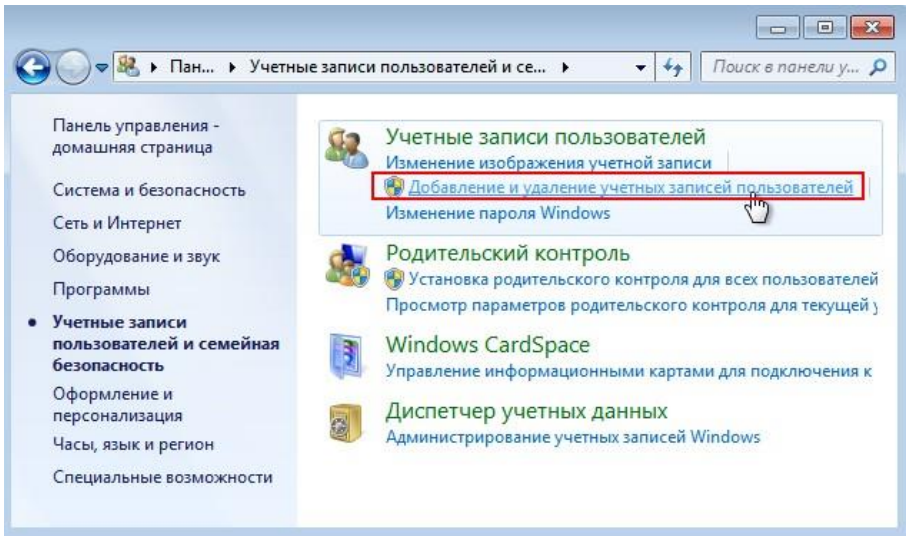
и вот, пожалуйста, наш индивидуальный рабочий стол.



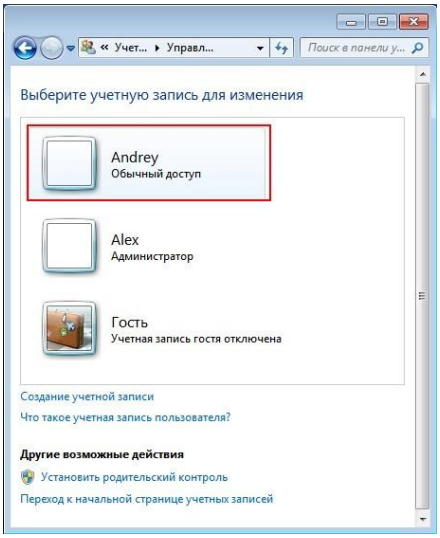
Или, если вы перезагрузите компьютер, опять же выйдет это меню входа в систему, в котором нужно выбрать нужную учётную запись и ввести пароль.

2.1.4 Изменение типа учётной записи

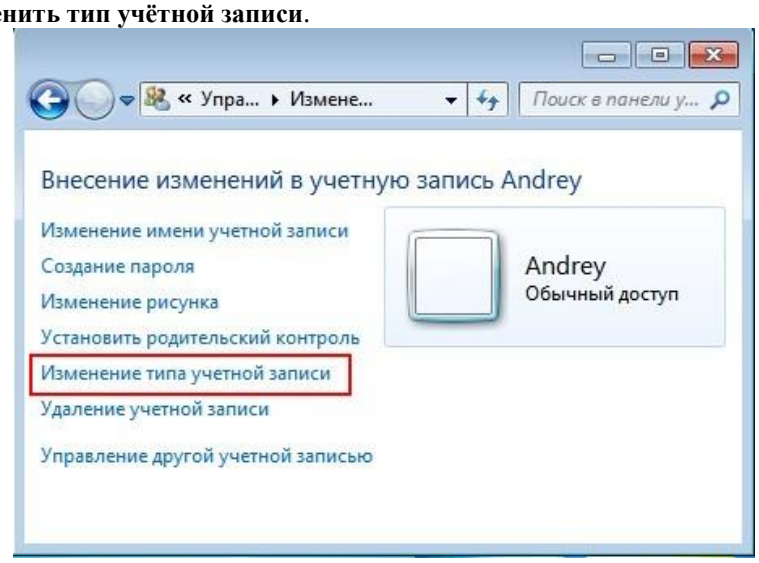
Если вы обладаете учётной записью с правами администратора, то вы можете изменить тип учётной записи любого пользователя. Например, пользователя с обычным доступом, вы можете сделать полноценным администратором компьютера. Но делать это нужно осторожно, если пользователь недостаточно опытен, то с такими правами он может наделать серьёзных дел на вашем компьютере. Также администратор может лишить любого пользователя административных прав. Как это всё происходит. Открываем меню **Пуск->Панель управления->Добавление и удаление учётных записей пользователей.**



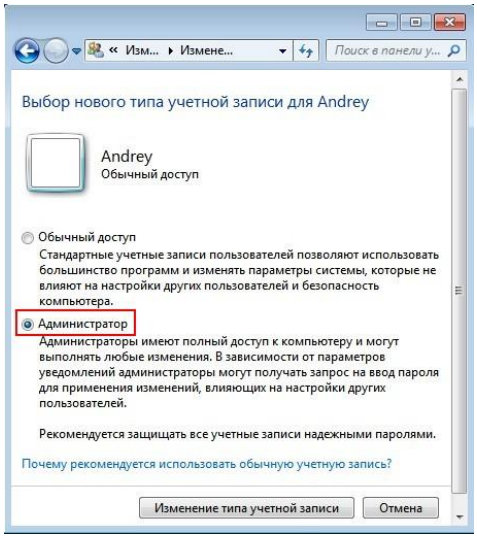
Нажимаем левой кнопкой мыши на любую учётную запись.



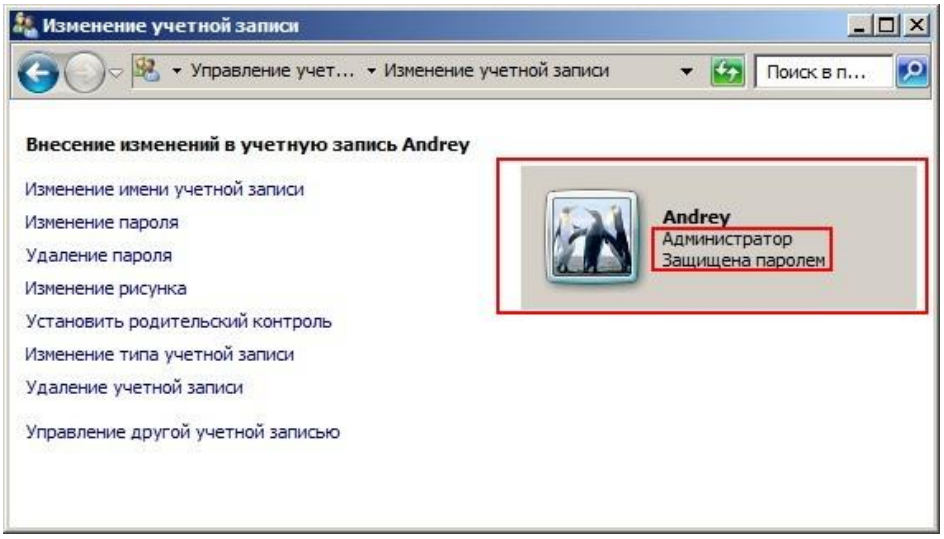
Выбираем **Изменить тип учётной записи**.



Отмечаем пункт **Администратор** и **Изменить тип учётной записи**.



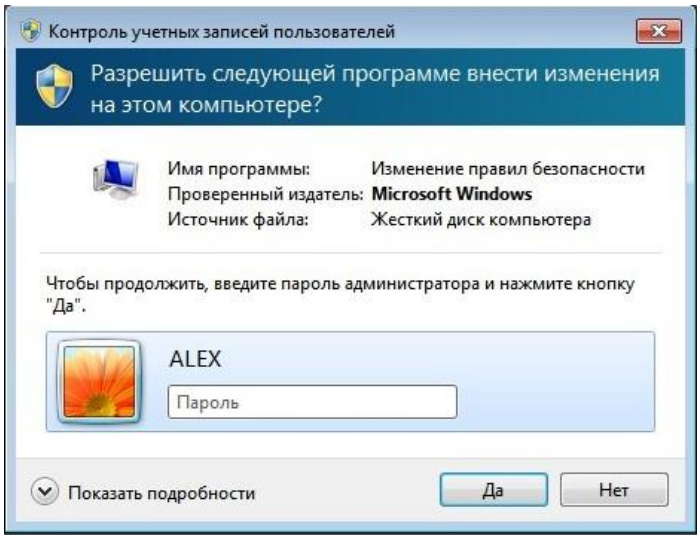
Наш пользователь Andrey становится пользователем с правами администратора.



Точно таким же образом, вы можете сделать его учётную запись обычной.

2.1.5 Как работать под учётной записью с ограниченными правами

Друзья! Если вы, работая в учётной записи с ограниченными правами, захотите установить или удалить какую-то программу, или удалить какие-нибудь НЕ принадлежащие вам файлы, естественно это затронет других пользователей вашего компьютера. В большинстве случаев, у вас при этом выйдет вот такое окно Контроля учётных записей пользователя, в котором вы должны ввести пароль администратора. «Чтобы продолжить, введите пароль администратора и нажмите кнопку «ДА».

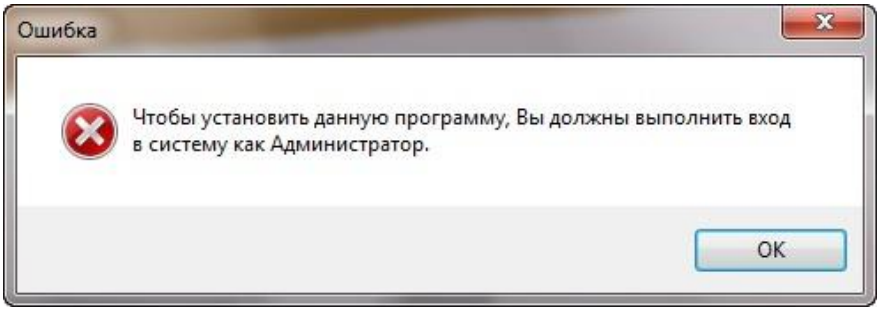


Введя пароль администратора, процесс инсталляции или удаления программы, или файлов продолжится. Если вы не знаете пароль, значит вам придётся обратиться к пользователю с административными правами, и он уже будет решать, можно ли удалять или устанавливать ту или иную программу. И ничего здесь сделать нельзя. Ещё раз напоминаю вам, что ничего обидного здесь нет и сделано это специально для тех случаев, когда компьютером пользуются несколько человек с разным уровнем подготовки или когда вашим компьютером пользуются дети.

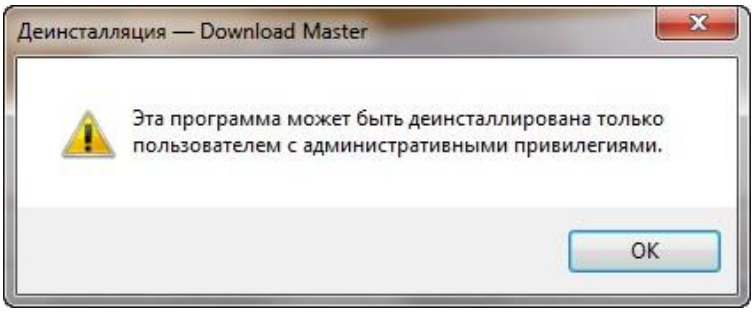
Создайте детям свою учётную запись с ограниченными правами, но не говорите пароль администратора, когда они захотят установить на ваш компьютер 100 игр сразу или удалить папку WINDOWS, вы об этом узнаете первым, как администратор компьютера. Кстати, у Вас есть помощник «Родительский контроль». С помощью родительского контроля вы будете контролировать деятельность детей за компьютером. Назначайте время использования детьми компьютера, разрешайте в каких программах и играх им можно работать. Как это сделать, подробно написано в конце статьи.

2.1.6 Какие ошибки могут возникнуть при работе в учётной записи с ограниченными правами

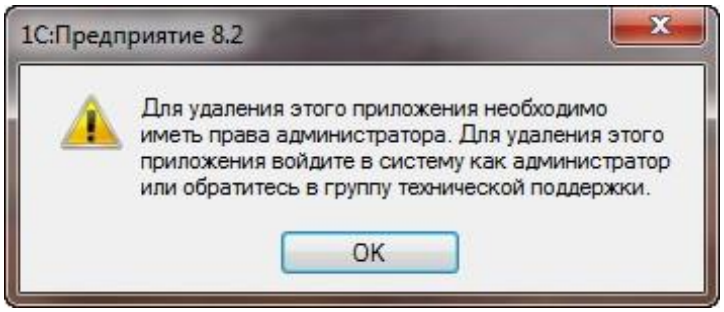
При установке программы на компьютер, даже если вы выберите при установке пункт «Запуск от имени администратора» вместо окна Контроля учётных записей пользователя, в котором вы должны ввести пароль администратора, у вас может выйти вот такая ошибка. «Чтобы установить данную программу. Вы должны выполнить вход в систему как Администратор».



Если вы захотите удалить программу, то появится предупреждение «Эта программа может быть деинсталлирована только пользователем с административными привилегиями».

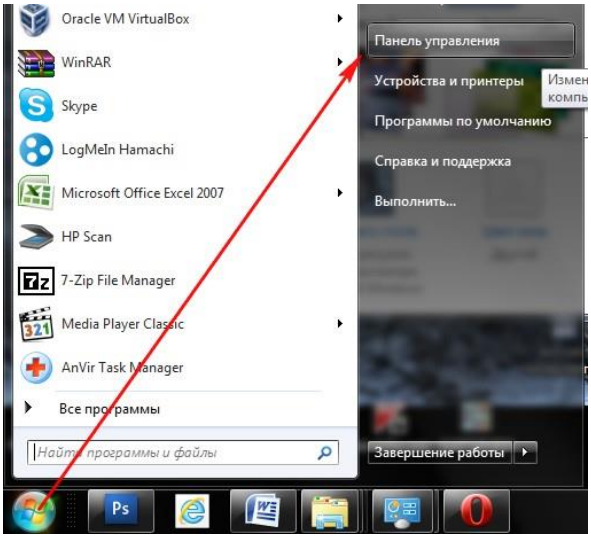


Или «Для удаления этого приложения необходимо иметь права администратора»,

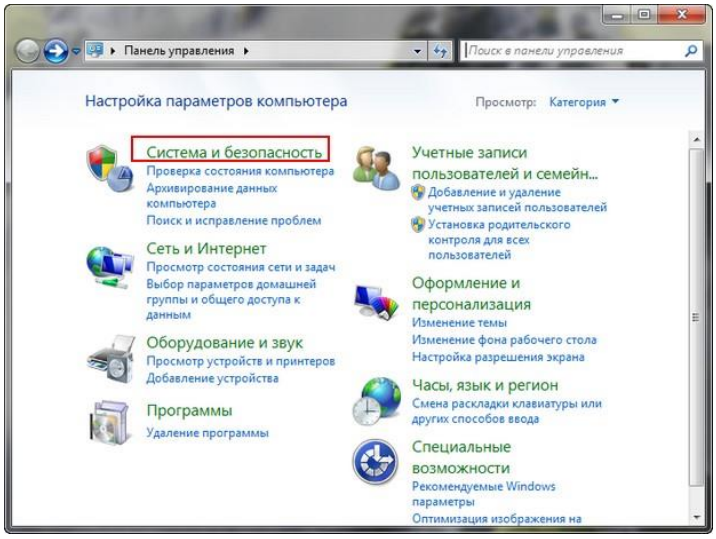


Всё это происходит потому, что в учётной записи администратора (будучи в которой вы создали себе учётную запись с ограниченными правами), у вас отключен Контроль учетных записей (UAC), который используется для уведомления пользователя перед внесением изменений, требующих прав администратора. Его нужно включить.

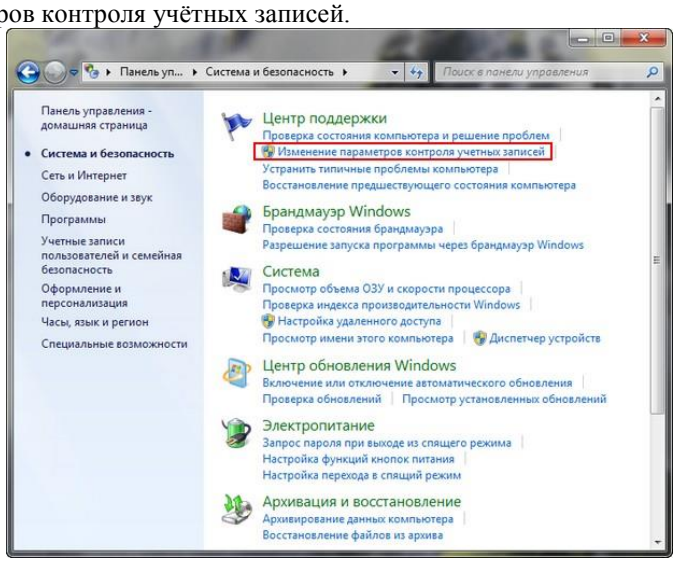
Пуск->Панель управления->



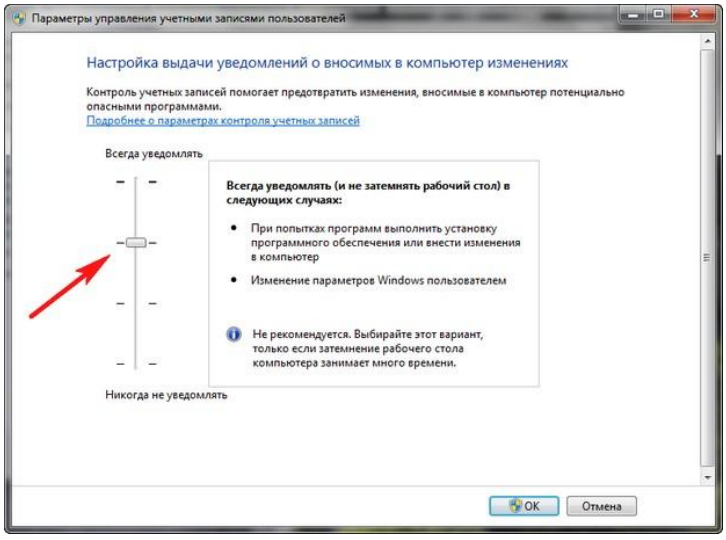
Система и безопасность->



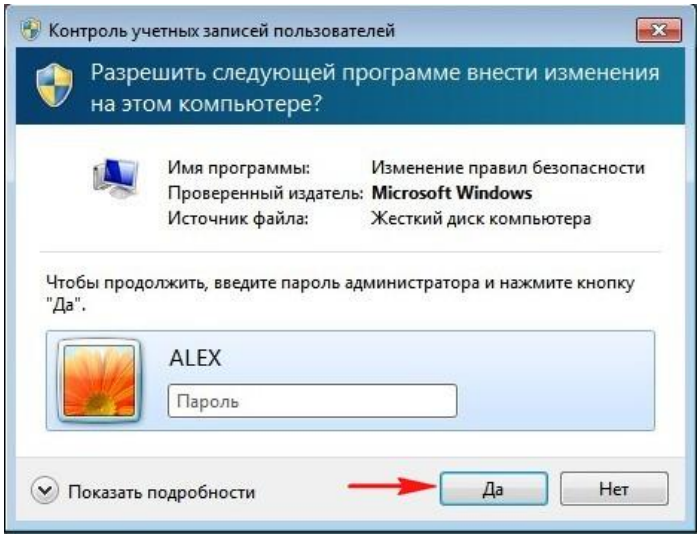
Изменение параметров контроля учётных записей.



Поднимаем шкалу на вторую позицию и нажимаем ОК.



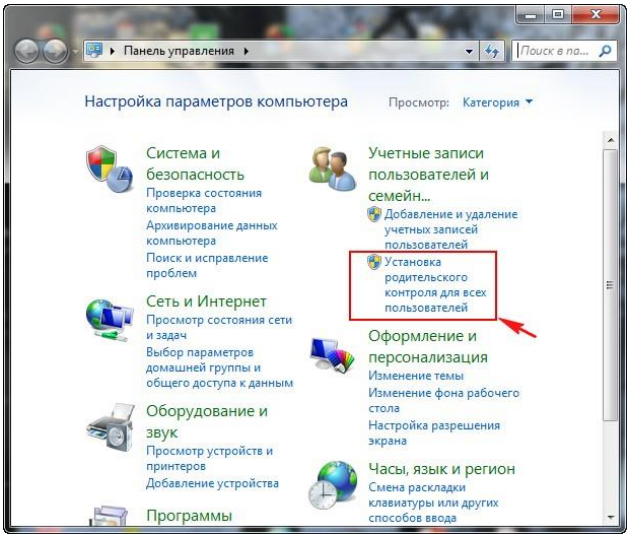
Да.



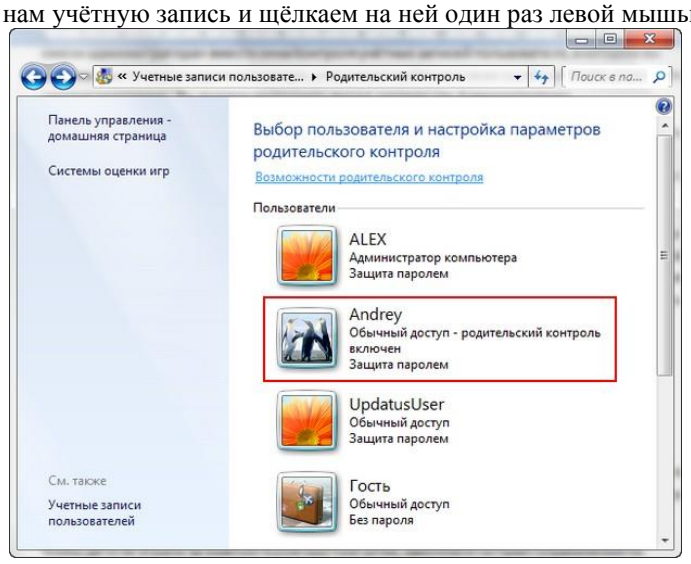
2.1.7 Родительский контроль

При помощи родительского контроля мы можем контролировать деятельность детей за компьютером. Например, ограничить время использования детьми компьютера и так далее.

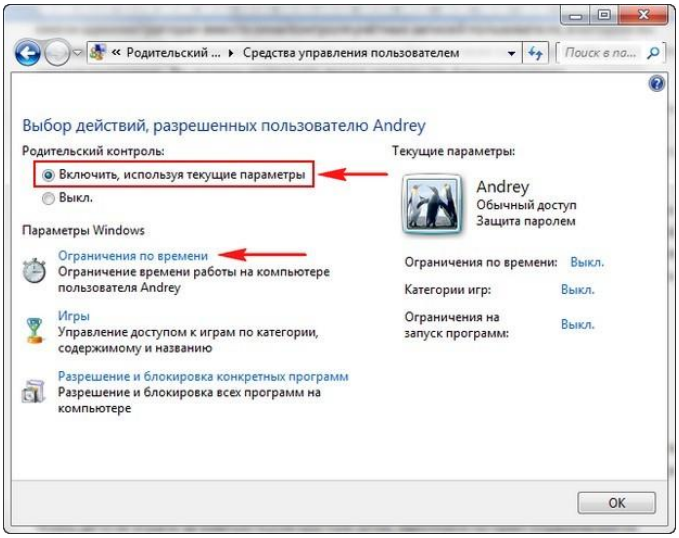
Выбираем Пуск->Панель управления->Установка родительского контроля для всех пользователей.



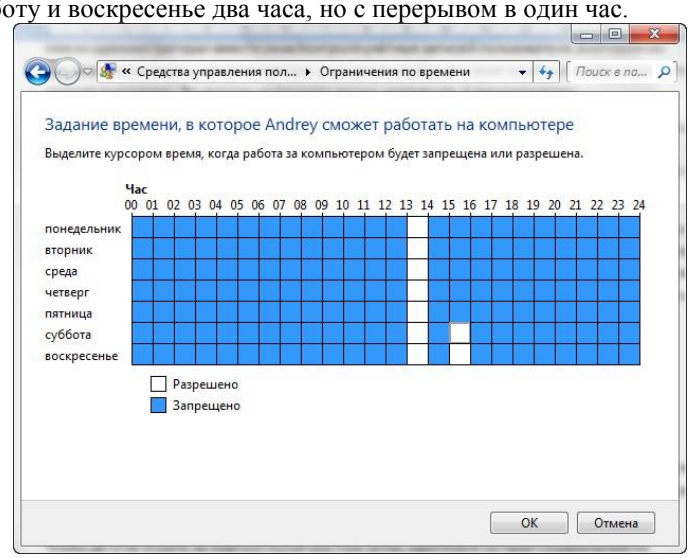
Выбираем нужную нам учётную запись и щёлкаем на ней один раз левой мышью.



Отмечаем пункт Включить, используя текущие параметры. Чтобы дети не играли за компьютером круглые сутки, нажимаем на пункт Ограничения по времени



и пожалуйста, можем настроить каждый день недели. Например, во все дни недели разрешим детям играть один час, с 13 до 14. В субботу и воскресенье два часа, но с перерывом в один час.



Также вы можете управлять доступом к играм и определённым программам!

Задание

1 Создайте резервную копию системы;
2 Создайте несколько учетных записей;
3 Ограничьте права доступа для обычных пользователей;

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №15:

1. Что такое отказоустойчивость?
2. Расскажите о пассивной и активной отказоустойчивости.
3. Как создать резервную копию ОС?
4. Что чего применяются средства восстановления системы?
5. Как разграничить права пользователей? Для чего это нужно?

Лабораторная работа №16
«Контрольная работа»

Цель работы:

1. Закрепить полученные знания.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ЛАБОРАТОРНОЙ РАБОТЕ №15:

1. Как перевести из десятичной системы счисления в любую другую?
2. Как перевести из двоичной системы счисления в любую другую?
3. Правило перевода числа из восьмеричной или шестнадцатеричной системы счисления?
4. Что такое триада и тетрада?
5. Что такое кодирование?
6. Чем отличается вероятностный подход от алфавитного?
7. Формула измерения количества информации.
8. Наименьшая единица количества информации?
9. Как перевести Мбайт в бит?
10. Что такое и для чего используется виртуальная машина?
11. Как создать общую папку? Для чего она используется?
12. Расскажите о возможностях ВМ?
13. Как выполнить арифметическое сложение двоичных чисел?
14. Как выполнить арифметическое вычитание двоичных чисел?
15. Как выполнить арифметическое умножение двоичных чисел?
16. Как выполнить арифметическое деление двоичных чисел?
17. Что такое интерфейс командной строки (CLI)?
18. Какой формат имеют команды в ОС UNIX?
19. Какие дополнительные средства CLI ОС UNIX вы знаете?
20. Что такое системный реестр?
21. Где находится системный реестр?
22. Из каких файлов состоит реестр? Где они расположены?
23. Назначение реестра;
24. Способы редактирования реестра;
25. Структура реестра;
26. Структура основного раздела Hkey_Clauses_Root;
27. Основные разделы и их назначение;
28. Параметры ключей. Типы параметров и их значения;
29. Назовите ключи, имеющие псевдонимы;
30. Способы восстановления реестра;
31. Описать виды ключей системного реестра и их значения?
32. Дать описание основных разделов системного реестра;
33. В каких файлах хранится информация о реестре и где они находятся?
34. Поясните особенности «троянских программ».
35. Почему профилактика «троянских программ» связана с системным реестром?
36. Какие разделы и ключи являются потенциальными местами записей «троянских программ»?
37. Какие программы называются файловыми менеджерами?
38. Какая информация отражается в области просмотра программы Konqueror?
39. Как создать новое окно с помощью программы Konqueror?
40. Перечислите задачи по управлению файловой системой, которые можно решать с помощью диспетчера файлов?
41. Перечислите стандартные функции KDE.
42. Что является компонентом рабочего стола KDE?
43. Назовите функции панели рабочего стола.
44. Как получить справку в диалоговом режиме?
45. Какие функции предоставляет центр управления KDE?
46. Дайте понятие процессу в операционной системе;
47. Дайте понятие службе в операционной системе;
48. Для чего служит диспетчер задач?
49. Причислите основные команда работы с процессами при помощи командной строки
50. Что такое регистр?
51. Предназначение регистра?
52. Какие регистры процессора вы знаете?
53. Расскажите о группах регистров;
54. Что такое ассемблер?
55. Как составляется команда ассемблера?
56. Что может использоваться в качестве операндов?

57. Для чего предназначены регистры mov, int, pop, push?
58. Расскажите о регистре Flags.
59. Что такое регистр процессора?
60. Какие регистры микропроцессорной памяти используются для адресации данных, команд программы, стековой памяти?
61. Назовите основные компоненты языка ассемблер, приведите структуру команд.
62. Приведите структуру ассемблерной программы и дайте краткую характеристику основных структурных фрагментов этой программы.
63. Каково назначение отладчика программ? Назовите основные его возможности.
64. Для чего используется сегментация памяти?
65. Для чего используется файл подкачки?
66. Как изменить размер файла подкачки?
67. Что такое своп-файл? Для чего отключают файл-подкачки?
68. Что называется, файлом и папкой?
69. По каким правилам составляются имена файлов и папок?
70. Как записать путь к файлу?
71. Какие вы знаете файловые системы?
72. Что такое маски имен?
73. Почему MS-DOS остается актуальным?
74. Назовите основные команды для работы с файлами в MS-DOS.
75. Доступны ли файлы одного каталога из другого каталога?
76. Как просмотреть содержимое каталога? Файла?
77. Какова структура командной строки?
78. Как получить справку о команде?
79. Что такое символы подстановки в имени файла и как они используются? Привести пример.
80. Как осуществляется контроль доступа над файлами?
81. Расскажите о синхронизации процессов.
82. Назовите программы для ограничения доступа к файлам и настройкам ОС;
83. Что такое отказоустойчивость?
84. Расскажите о пассивной и активной отказоустойчивости.
85. Как создать резервную копию ОС?
86. Что чего применяются средства восстановления системы?
87. Как разграничить права пользователей? Для чего это нужно?

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Попов, И. И. Операционные системы, среды и оболочки. / И. И. Попов, Т. Л. Партыка. – М. : Форум, 2011. – 544 с.
2. Бушмелева, Н. А. Изучаем информацию, ее измерения и кодирование : учебное пособие. / Н. А. Бушмелева. – Киров. : Питер, 2011. – 25 с.
3. http://www.intuit.ru