

Министерство науки и высшего образования Российской Федерации  
федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Кузбасский государственный технический университет  
имени Т.Ф. Горбачева»

Институт профессионального образования  
Кафедра информатики и информационных систем

Александр Вениаминович Матисов  
Илья Валериевич Кулак

## **ОПЕРАЦИОННЫЕ СИСТЕМЫ И СРЕДЫ**

Методические материалы к практическим занятиям  
и самостоятельной работе

Рекомендовано цикловой методической комиссией по  
специальности СПО 09.02.07 Информационные системы  
и программирования в качестве электронного издания  
для использования в образовательном процессе

Кемерово 2024

Рецензенты: Чичерин И.В. – канд. тех. наук, доцент, заведующий кафедрой информационных и автоматизированных производственных систем ФГБОУ ВО «Кузбасский государственный технический университет имени Т.Ф. Горбачева»

**Матисов, А.В., Кулак, И.В. Операционные системы и среды:** методические материалы к практическим занятиям и самостоятельной работе для обучающихся специальности СПО 09.02.07 «Информационные системы и программирование» очной формы обучения/ сост. А. В. Матисов, И.В. Кулак; Кузбасский государственный технический университет имени Т.Ф. Горбачева». – Кемерово, 2024. – Текст: электронный.

В методических материалах приведено содержание практических занятий и самостоятельной работы. Назначение издания – помощь обучающимся в получении знаний по дисциплине «Операционные системы и среды» и организация практических занятий и самостоятельной работы.

© Кузбасский государственный  
технический университет  
имени Т.Ф. Горбачева, 2024  
©Матисов А. В.,Кулак И.В.  
составление, 2024

## ОГЛАВЛЕНИЕ

1. ИСПОЛЬЗОВАНИЕ СЕРВИСНЫХ ПРОГРАММ ПОДДЕРЖКИ ИНТЕРФЕЙСОВ. НАСТРОЙКА РАБОЧЕГО СТОЛА. НАСТРОЙКА СИСТЕМЫ С ПОМОЩЬЮ ПАНЕЛИ УПРАВЛЕНИЯ. РАБОТА СО ВСТРОЕННЫМИ ПРИЛОЖЕНИЯМИ.....	4
2. УПРАВЛЕНИЕ ПРОЦЕССАМИ С ПОМОЩЬЮ КОМАНД ОПЕРАЦИОННОЙ СИСТЕМЫ ДЛЯ РАБОТЫ С ПРОЦЕССАМИ .....	20
3. ИССЛЕДОВАНИЕ СООТНОШЕНИЯ МЕЖДУ ПРЕДСТАВЛЯЕМЫМ И ИСТИННЫМ ОБЪЁМОМ ЗАНЯТОЙ ДИСКОВОЙ ПАМЯТИ. ИЗУЧЕНИЕ ВЛИЯНИЯ КОЛИЧЕСТВА ФАЙЛОВ НА ВРЕМЯ, НЕОБХОДИМОЕ ДЛЯ ИХ КОПИРОВАНИЯ.....	25
4. УПРАВЛЕНИЕ ПАМЯТЬЮ .....	40
5. РАБОТА С ПРОГРАММОЙ «ФАЙЛ-МЕНЕДЖЕР ПРОВОДНИК». РАБОТА С ФАЙЛОВЫМИ СИСТЕМАМИ И ДИСКАМИ .....	66
6. РАБОТА С КОМАНДАМИ В ОПЕРАЦИОННОЙ СИСТЕМЕ. ИСПОЛЬЗОВАНИЕ КОМАНД РАБОТЫ С ФАЙЛАМИ И КАТАЛОГАМИ. РАБОТА С ДИСКАМИ .....	83
7. КОНФИГУРИРОВАНИЕ ФАЙЛОВ. УПРАВЛЕНИЕ ПРОЦЕССАМИ В ОПЕРАЦИОННОЙ СИСТЕМЕ. РЕЗЕРВНОЕ ХРАНЕНИЕ, КОМАНДНЫЕ ФАЙЛЫ.....	102
8. РАБОТА С ТЕКСТОВЫМ РЕДАКТОРОМ. РАБОТА С АРХИВАТОРОМ. РАБОТА С ОПЕРАЦИОННОЙ ОБОЛОЧКОЙ .....	115
9. УСТАНОВКА И НАСТРОЙКА СИСТЕМЫ. УСТАНОВКА ПАРАМЕТРОВ АВТОМАТИЧЕСКОГО ОБНОВЛЕНИЯ СИСТЕМЫ. УСТАНОВКА НОВЫХ УСТРОЙСТВ. УПРАВЛЕНИЕ ДИСКОВЫМИ РЕСУРСАМИ .....	120

10. ИЗУЧЕНИЕ ЭМУЛЯТОРОВ ОПЕРАЦИОННЫХ СИСТЕМ. УСТАНОВКА ОПЕРАЦИОННОЙ СИСТЕМЫ .....	124
11. МЕТОДИЧЕСКИЕ УКАЗАНИЯ К САМОСТОЯТЕЛЬНОЙ РАБОТЕ ОБУЧАЮЩЕГОСЯ .....	130
12. СПИСОК ЛИТЕРАТУРЫ.....	131

# **1. ИСПОЛЬЗОВАНИЕ СЕРВИСНЫХ ПРОГРАММ ПОДДЕРЖКИ ИНТЕРФЕЙСОВ. НАСТРОЙКА РАБОЧЕГО СТОЛА. НАСТРОЙКА СИСТЕМЫ С ПОМОЩЬЮ ПАНЕЛИ УПРАВЛЕНИЯ. РАБОТА СО ВСТРОЕННЫМИ ПРИЛОЖЕНИЯМИ**

## **1.1 ЦЕЛЬ РАБОТЫ**

Цель работы – рассмотреть:

- сервисные программы поддержки интерфейсов;
- основные настройки среды рабочего стола компьютера;
- основные настройки системы с помощью панели управления.

## **1.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

### *1.2.1 Понятие интерфейса*

Интерфейс – это способ общения пользователя с персональным компьютером (ПК), пользователя с прикладными программами и программ между собой. Интерфейс служит для удобства управления программным обеспечением компьютера. Интерфейсы бывают *однозадачные и многозадачные, однопользовательские и многопользовательские*. Интерфейсы отличаются между собой по удобству управления программным обеспечением, то есть по способу запуска программ.

В общем случае следует говорить о *связи с внешней средой*, поскольку, например, при использовании компьютеров в системах управления технологическими комплексами (производство, летательные аппараты, корабли и пр.) человек может быть исключен (полностью или частично) из контура управления и внешними устройствами для компьютера будут *датчики* (скорости, высоты, давления, температуры) и *эффе́кторы* (приводы рулей, манипуляторы, сервомоторы вентилей и пр.).

*Связь с пользователем* – последняя в списке, но не по важности функция операционной системы (ОС).

Связь с пользователем включает:

- командный (или иной) интерфейс по управлению системными процессами в вычислительной системе (собственно функции оператора ОС). Пользователь (привилегированный) осу-

ществляет запуск-останов программ, подключение-отключение устройств и прочие релевантные операции;

- интерфейс по управлению пользовательскими процессами (контроль состояния процесса, ввод-вывод данных в процесс/из процесса).

В состав *пользователей* в общем случае включаются следующие группы лиц, контактирующих с системой:

- администратор системы лицо или группа, отвечающая за сопровождение данных, назначение уровней доступа, включение/исключение пользователей;

- оператор системы, осуществляющий сопровождение вычислительного процесса,

- прочие пользователи (не обладающие привилегиями доступа к данным), в том числе:

- операторы подготовки данных (ОПД) – персонал, осуществляющий ввод данных с рабочих листов или документов, на основе соответствующих инструкций, в среде специальных программных интерфейсов;

- интерактивные пользователи (ИП) – лица, имеющие доступ на ввод, коррекцию, обновление, уничтожение и чтение данных в рамках ограниченной области базы данных (БД);

- конечные пользователи (КП) – лица, использующие БД для получения справок и решения задач.

Очевидно, что именно *оператор ПК является естественным пользователем ОС*, все же прочие пользователи становятся таковыми лишь вследствие расширения функций пользователя в связи с интеграцией (особенно в случае персональных компьютеров) функций конечного пользователя, администратора системы и оператора.

### *1.2.2 Сущность и назначение сервисных программ*

Пользовательский интерфейс (сервисные программы) – это программные надстройки операционной системы (оболочки и среды), предназначенные для упрощения общения пользователя с операционной системой.

Сервисные программы предназначены для выполнения различных вспомогательных операций – проверки исправности оборудования, архивации файлов, борьбы с вирусами, форматирова-

ния дисков (подготовки новых дисков к работе путем разметки на них дорожек и секторов).

Программы, обеспечивающие интерфейс, сохраняют форму общения (диалог) пользователя с операционной системой, но изменяют язык общения (обычно язык команд преобразуется в язык меню).

Сервисные программы представляют собой совокупность программных продуктов, которые повышают возможности ОС и предоставляют пользователю дополнительные услуги в работе с ПК. Некоторые сервисные программы могут автоматически загружаться в оперативную память компьютера при загрузке ОС и находиться в памяти до окончания сеанса работы. Такие сервисные программы называются резидентными.

Сервисные системы условно можно разделить на интерфейсные системы, автономные программы, оболочки операционных систем и утилиты.

*Интерфейсные системы* – это мощные сервисные системы, чаще всего графического типа, совершенствующие не только пользовательский, но и программный интерфейс операционных систем, в частности, реализующие некоторые дополнительные процедуры разделения дополнительных ресурсов.

К *автономным программам* можно отнести программы тестирования и контроля, которые являются программно-аппаратными средствами тестирования различных устройств компьютера.

*Оболочки операционных систем* предоставляют пользователю качественно новый, по сравнению с реализуемой операционной системой, интерфейс и делают необязательным знание последнего. Оболочка – это надстройка над операционной системой. Находясь в среде оболочки, пользователь может выполнять большинство команд по работе с операционной системе. Например, для операционных систем семейства Windows можно использовать оболочку FAR.

*Утилиты* автоматизируют выполнение отдельных типовых, часто используемых процедур, реализация которых потребовала бы от пользователя разработки специальных программ. Многие утилиты имеют развитый диалоговый интерфейс с пользователем и приближаются по уровню общения к оболочкам.

Утилита (англ. *utility* или *tool*) – программный продукт, предназначенный не для решения какой-либо прикладной задачи, а для решения вспомогательных задач.

Большинство утилит оформлены как встроенные служебные программы системы.

Они адаптированы к возможностям системы данного типа и используются:

- для проверки работы диска, его дефрагментации;
- создания рабочих архивов;
- для восстановления системы;
- очистки системы и ее модернизации.

Утилиты предназначены для расширения возможностей операционной системы и встроенных в систему служебных программ за счет введения новых или усовершенствования уже существующих функций.

Компьютерные утилиты можно разделить на три группы:

- утилиты сервисного обслуживания компьютера;
- утилиты расширения функциональности;
- информационные утилиты.

Утилиты последних поколений характеризуются быстрым действием, многофункциональностью, возможностью работы в среде современных операционных систем, интеграцией в приложения и браузеры. При модернизации операционной системы утилиты не исчезают, а исправно функционируют на базе усовершенствованного ядра.

Утилиты можно условно разделить на несколько разновидностей:

- программы для работы с дисками обеспечивают проверку работоспособности, структурирование, дефрагментацию, очистку дисков и сжатие данных;
- антивирусные программы предназначены для защиты файловой системы от повреждения компьютерными вирусами;
- программы для выявления неисправностей предназначены для слежения за работой системных компонентов, диагностики и подготовки отчетов об аппаратных неисправностях, а также для обнаружения программных ошибок и восстановления системы;
- программы для сжатия (архивации) файлов и резервного копирования предназначены для создания копий программ и доку-



ментов. Это необходимо для переноса данных на другой ПК, а также для создания резервного архива данных и программ;

- программы для ускоренного просмотра файлов позволяют просматривать файлы в различных форматах, не запуская полнофункциональных приложений и даже не имея их вовсе;

- программы для работы в локальной сети и Интернете предназначены для дистанционного доступа к ресурсам ПК и коллективного использования компонентов сети – базы данных, принтера. Программы этого типа обеспечивают ускорение обмена данными, подключение к различным услугам, предоставляемым в Интернете, контролируют использование ресурсов и защищают данные от несанкционированного доступа;

- программы компьютерной безопасности защищают ПК и хранящиеся в нем данные от несанкционированного проникновения (взлома). К таким программам относятся системы шифрования и наблюдения за данными;

- программы для работы с устройствами мультимедиа (видео, звуковой системой, видеокамерами и т.д.);

- программы-деинсталляторы предназначены для корректной очистки ОС от элементов удаляемых программ;

- «узкофункциональные» утилиты – загрузчики различных операционных систем, утилиты просмотра буфера обмена, печати информации.

Программные оболочки операционных систем подключают ПК к интерфейсу, обеспечивающему простой и быстрый доступ к программам. Такой оболочкой, в частности, является широко известная утилита Windows FAR Manager.

Наибольшие проблемы в системе возникают из-за ошибочных действий при модернизации системы, BIOS (от англ. *BasicInput-OutputSystem* – базовая система ввода-вывода), при дополнении ПК новыми устройствами или программами, а также из-за внесения некорректных записей в системный реестр.

К исчерпанию пространства дисковой памяти и потере работоспособности ПК приводит переполнение диска ненужными записями. Повреждение программ при удалении файлов или попытке установить не протестированные программы также приводит к ухудшению показателей системы. Подобных примеров множество.

Для организации глубоких проверок, а также восстановления работоспособности ПК служат специализированные утилиты. Удачно подобранный комплект утилит поможет вовремя обнаружить программные конфликты, поддержит работоспособность жесткого диска, аккуратно удалит ненужные файлы и папки, выполнит сортировку документов и приложений.

Неэффективность пакета диагностических программ не только грозит потерей времени, но и приводит к негативным последствиям, когда состояние ПК окажется намного хуже, чем до работ по восстановлению работоспособности.

Эффективно работающие программы диагностики должны расширять возможности служебных программ Windows следующими функциями:

- поиском проблем при работе с приложениями и автоматическим их исправлением;
- поиском аппаратных неисправностей, что реализуется комплектом тестов диагностики;
- более быстрой и эффективной оптимизацией параметров системы;
- защитой от "зависаний" и отказов системы, для чего в диагностические утилиты интегрированы программные модули защиты от аварийных отказов системы и средства для создания дискет, позволяющих восстановить систему;
- возможностью модернизации программ из Интернета.

Выделяют несколько категорий диагностических утилит:

- программы для проверки аппаратуры и подготовки отчетов по результатам проверок, например пакет программ проверки аппаратуры CheckItUtilities (SmithMicro).
- интегрированные диагностические комплексы с обширным перечнем возможностей для борьбы с компьютерными вирусами, неисправностями и средствами оптимизации систем. Примерами таких пакетов программ являются Norton System Works (Symantec), Nuts & Bolts (Network Associated), System Mechanics (Iolo), McAfee Utilities (McAfee), System Suite (Ontrack) и т.д.

На пакеты утилит возлагается выполнение пяти основных задач:

- диагностика и исправление ошибок в работе ПК;
- поддержка работоспособности жестких дисков;

- работа с файлами;
- резервное копирование системы;
- антивирусная защита.

К узкопрофессиональным программам относится огромное множество программ специального назначения, ориентированных на специалистов в определенной области. Например, для расчетов прочности строительных конструкций, управления работой атомной электростанции, бухгалтерских расчетов и т.д.

Однако независимо от рода деятельности любой работник часто сталкивается с необходимостью подготовки каких-то текстовых документов, например, заявлений, отчетов, деловых писем и т.д. Для этих целей используют специальные программы – текстовые редакторы, например, Word. Разновидностью текстовых редакторов являются издательские системы, используемые при издании книг, журналов, газет, рекламных объявлений.

Очень часто человек сталкивается с необходимостью выполнить какие-то расчеты или другие операции над данными в табличной форме. Вообще, таблицы сопровождают нас всю жизнь – расписание уроков, классный журнал, экзаменационная ведомость, расписание поездов, турнирная таблица футбольного чемпионата и т.д.

Для автоматизированной обработки данных в табличной форме используют специальные программы – электронные таблицы.

### *1.2.3 Рабочий стол Windows*

Рабочий стол компьютера – это основное рабочее место пользователя, появляющееся на экране сразу после загрузки операционной системы. Так же как и на обычном рабочем столе пользователь может разместить на нем все необходимые для работы инструменты, учитывая что речь идет о компьютерах точнее будет сказать программы, а также документы, медиа-файлы и все, что может понадобится ему для работы.

Естественно, чтобы за рабочим столом было удобно трудиться всё необходимое должно располагаться на нем в порядке, максимально подходящем для конкретного пользователя. Поэтому очень важно перед началом работы грамотно организовать свое рабочее место, а в нашем случае – настроить рабочий стол и все находящиеся на нем инструменты.

Рабочий стол появляется на экране компьютера после загрузки Windows. Он выглядит примерно так:

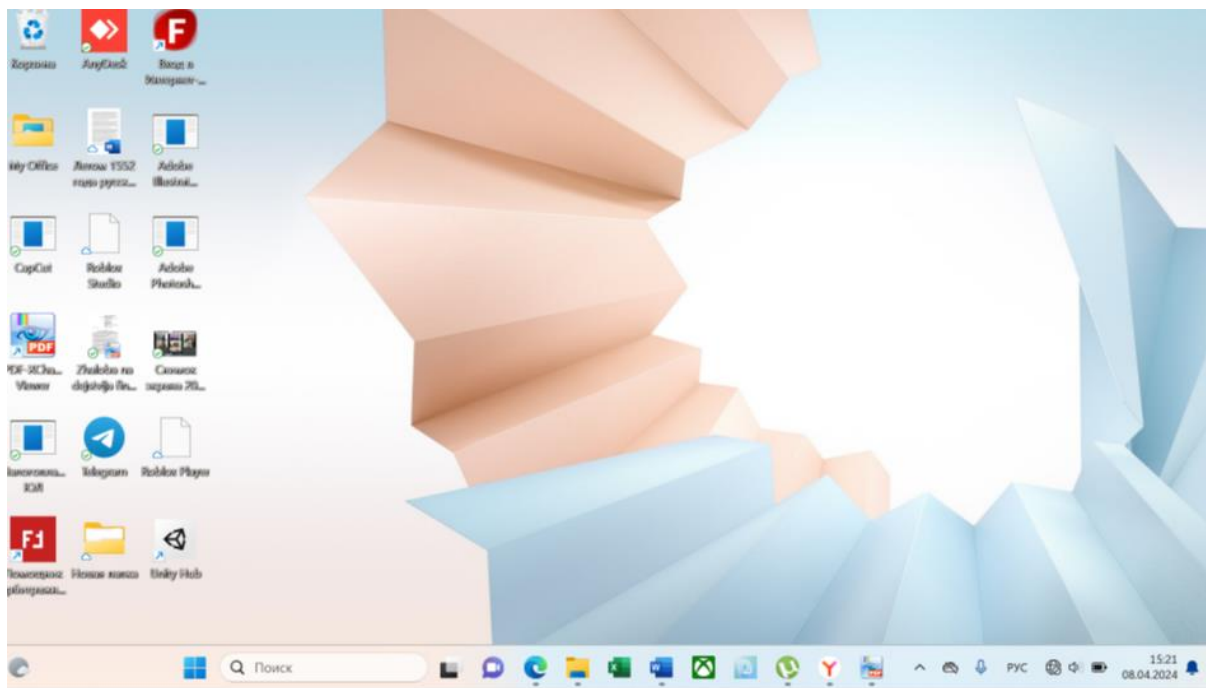


Рисунок 1 – Рабочий стол в Windows 11

В первую очередь стоит поместить на рабочий стол ярлыки всех программ и файлов, которые могут пригодиться в работе. Большинство программ при установке автоматически создают ярлык на рабочем столе, а если и нет, то создать его можно обратившись в меню «Пуск». Если какой то ярлык не нужен, то можно без проблем удалить его с рабочего стола, а при необходимости восстановить, пользуясь все тем же меню «Пуск». Разместить ярлыки в удобном для пользователя порядке можно просто перетаскивая их по экрану, удерживая при этом левой кнопкой мышки.

Набор элементов, находящихся на рабочем столе, также как и фон, зависит от настройки Windows. Ниже кратко описаны важные элемента.

*Корзина* предназначена для временного хранения удаленных файлов. Она позволяет восстановить ошибочно удаленные файлы.

*Панель задач* предназначена для управления программами, выполняющимися в сеансе Windows (запуска программ и переключения между работающими программами). Кроме этого, она позволяет изменять расположение окон на рабочем столе и выполняет информационную функцию.

Находящаяся на панели задач кнопка *ПУСК* позволяет запустить программу, открыть документ, изменить настройку системы, получить справочные сведения, найти нужный файл и многое другое, в частности, открыть *Главное меню*.

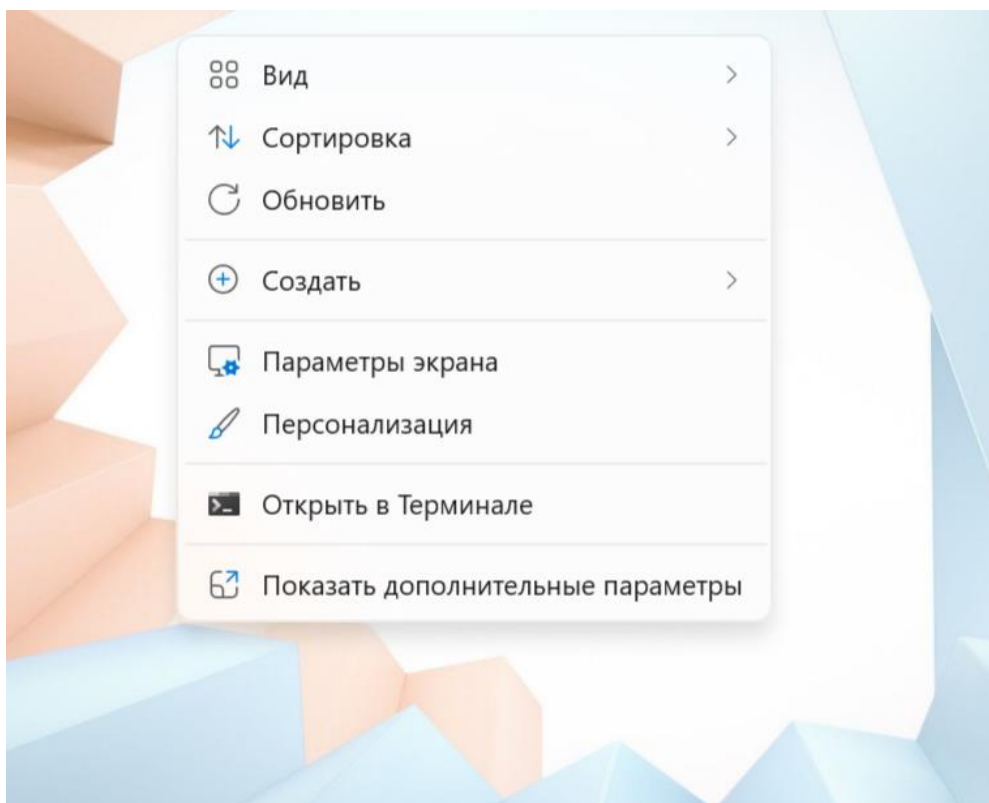


Рисунок 2 – Контекстное меню Рабочего стола

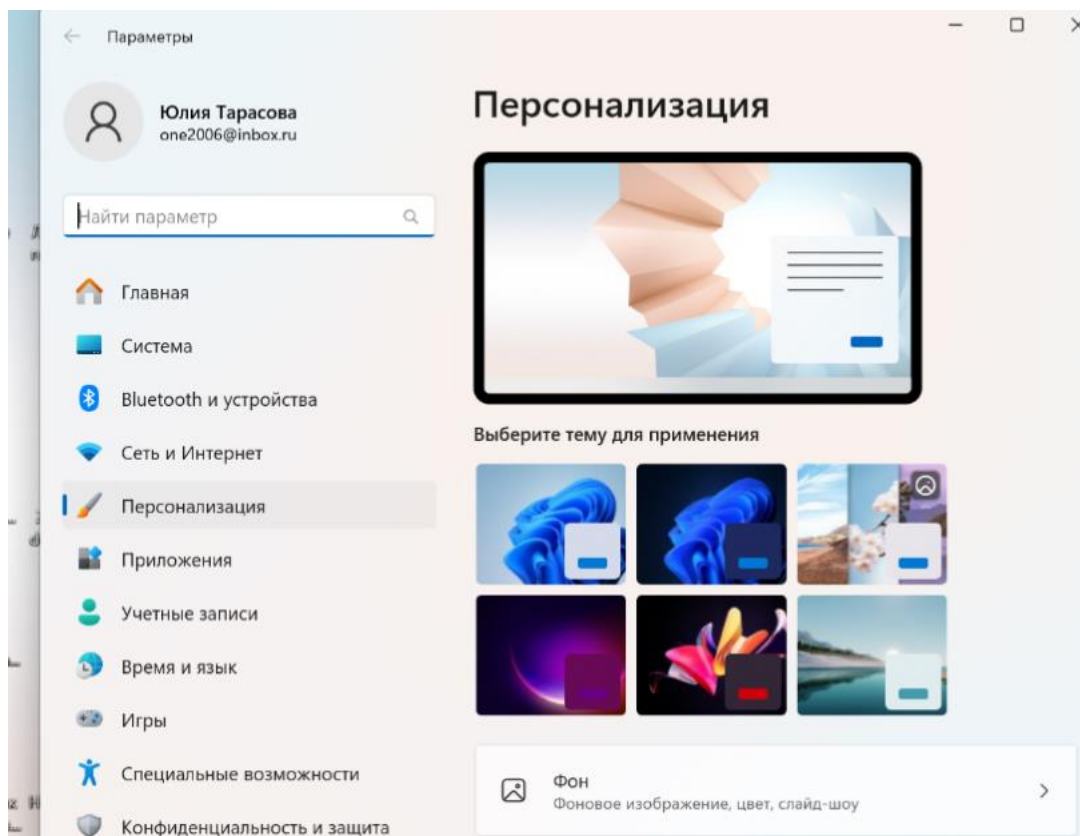


Рисунок 3 – Настройка параметров оформления

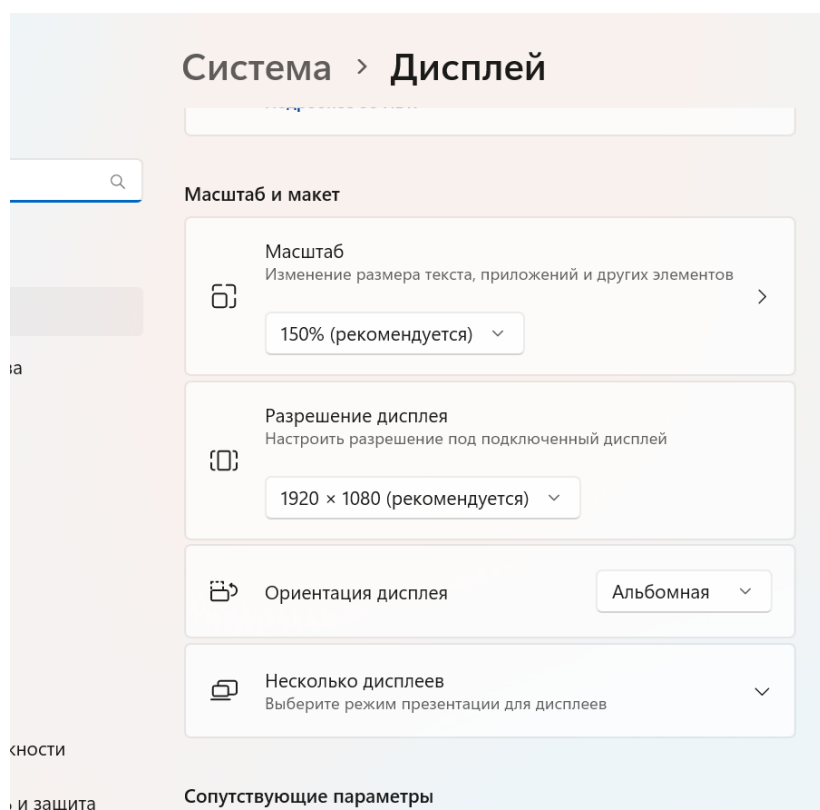


Рисунок 4 – Настройка параметров экрана

Простым размещением ярлыков настройка рабочего стола не заканчивается. Для того чтобы сделать работу максимально комфортной имеет смысл воспользоваться меню настроек. Для его вызова достаточно щелкнуть правой кнопкой мыши в любом свободном месте рабочего стола.

Пользуясь меню настроек можно увеличить или уменьшить размеры значков на рабочем столе, а также изменить размер текста и прочих элементов. Кроме этого, воспользовавшись разделом «Персонализация» можно сменить оформление рабочего стола, экранную заставку и «обои», внешний вид меню и папок. Одним словом рабочий стол можно сделать не просто функциональным, но и максимально привлекательным.

В нижней части рабочего стола находится так называемая «Панель задач». Она состоит из нескольких инструментов



Рисунок 6 – Элементы Панели задач

### *Настройка меню Пуск.*

В левом углу Панели задач находится кнопка «Пуск», при нажатии на которую открывается одноименное меню. Его использование упрощает доступ к часто используемым программам, интернету, электронной почте и так далее. Кроме этого меню «Пуск» дает доступ к панели управления компьютером, в которой можно произвести настройку всех его компонентов.

### *Настройка Панели задач.*

В Панели задач отображаются все открытые программы и файлы, а также находятся значки программ закрепленных в панели для обеспечения к ним быстрого доступа (один щелчок левой кнопкой мышки). Для настройки Панели задач, так же как для настройки меню «Пуск» необходимо нажать на нее правой кнопкой. Используя открывшееся меню можно закрепить или удалить программы из Панели задач, определить местоположение Панели задач на экране компьютера, скрыть панель в автоматическом режиме, изменить размер и компоновку значков и так далее.

### *Настройка панели часов.*

В правом углу Панели задач находятся часы. Для их

настройки можно использовать «Параметры» – «Время и язык», которая открывается из меню «Пуск», либо щелчок правой кнопкой мышки по часам и выбор пункта «Настроить дату и время». Воспользовавшись открывшимся меню можно установить время и дату, выбрать часовой пояс и так далее.

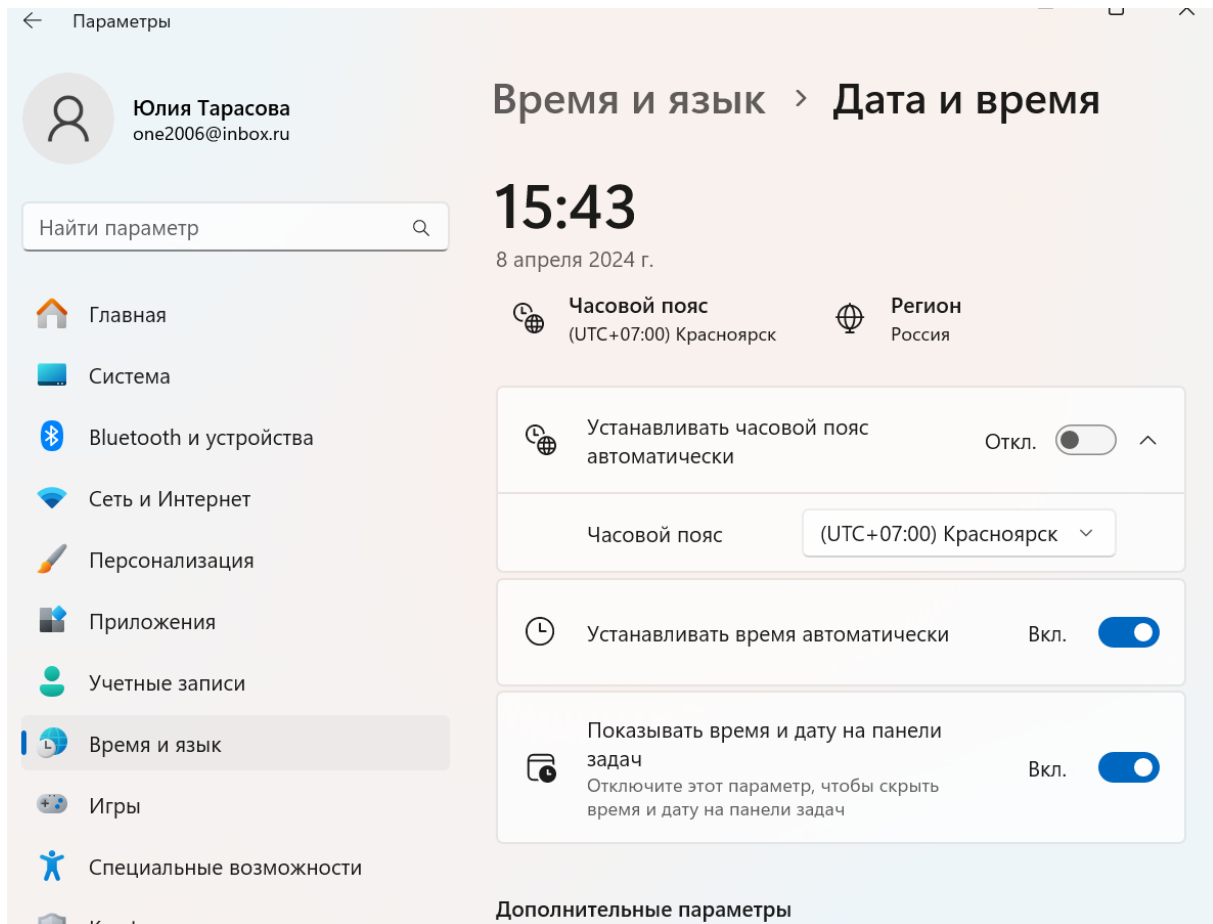


Рисунок 7 – Настройка даты и времени

### *Панель выбора языка.*

Рядом с часами на Панели задач находится языковая панель. На ней отображается текущий язык ввода текста (двумя первыми буквами). Для настройки языковой панели необходимо воспользоваться левой кнопкой мышки. Открывшееся меню даст возможность развернуть языковую панель (она будет отображаться в верхней части экрана). Кликнув по вкладке «Дополнительные настройки клавиатуры» можно выбрать язык ввода «по умолчанию», добавить или удалить языки, а также выбрать сочетание клавиш для их смены.



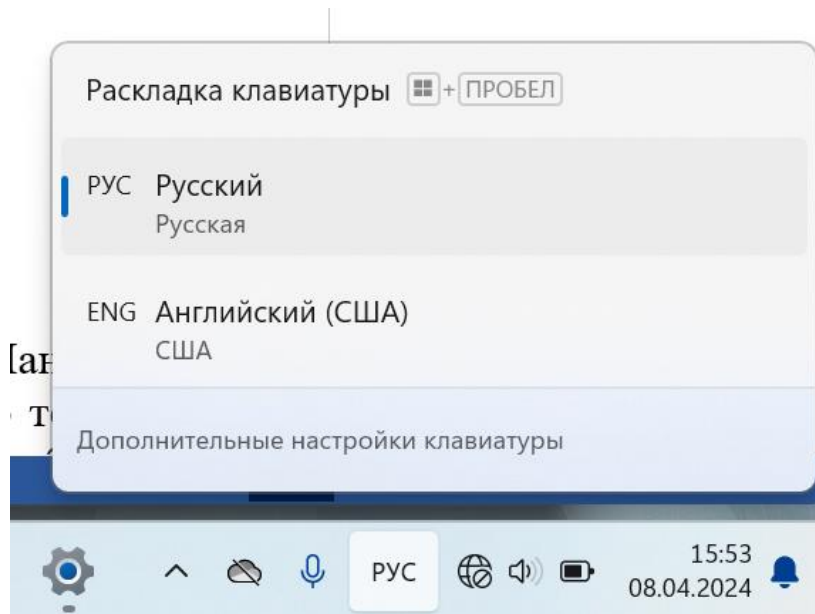


Рисунок 8 – Панель выбора языка

#### *Область уведомлений.*

Область уведомлений расположена в правой части Панели задач, и содержит в себе значки, которые информируют о текущем состоянии компьютера, подключению к интернету, работе антивирусов, получении электронной почты и так далее.



Рисунок 9 – Область уведомлений

#### *1.2.4 Параметры*

«Параметры» является частью пользовательского интерфейса Microsoft Windows. Они позволяют выполнять основные действия по настройке системы, такие, как добавление и настройка устройств, установка и деинсталляция программ, управление учётными записями, включение специальных возможностей, а также многие другие действия, связанные с управлением системой. Апплеты.

В «Параметрах» сосредоточены основные настройки Windows 11. Чтобы открыть эту папку, следует нажать кнопку «Пуск» – «Параметры».

Кратко рассмотрим основные разделы Параметров.

Выберите слева на панели раздел «Главная», чтобы отобразить список настроек в данной группе.

В данной группе можно выбрать часто используемые параметры: установленные приложения, настройки сетей и Bluetooth соединений, настройки облачного хранилища, варианты оформления рабочего стола, называемый темой, выбрать фоновый рисунок, заставку и т.д.

В разделе «Система» производятся настройки основных параметров системы: настройки дисплея, звука и системных уведомлений, настройки питания, памяти устройства, режима многозадачности, а также содержится информация об активации операционной системы и др.

Раздел «Bluetooth и устройства» предназначены для подключения к компьютеру и настройки беспроводных устройств: мышки, камеры, беспроводные гарнитуры, устройства ввода и т.д.

В разделе «Сеть и Интернет» отображаются настройки сетевых возможностей Windows, а именно: настройка соединения с Интернетом, создание и настройка домашней сети, соединение нескольких сетей – вот далеко не полный перечень задач, решаемых с помощью ссылок в этой папке.

В разделе «Персонализация» можно выполнить настройку и оформления индивидуального личного пространства, в том числе настроить и оформить рабочий стол, шрифты, фоновые рисунки и заставки, установить различные темы оформления и т.д.

Раздел «Приложения» содержит сведения об установленном программном обеспечении пользователя, а также системных приложениях. В данном разделе пользователь может самостоятельно произвести конфигурацию программного обеспечения своего компьютера соответственно установив или удалив соответствующие программы, а также получить системную информацию об установленных приложениях, объеме дискового пространства которые они занимают и т.д.

Раздел «Учетные записи» содержит информацию о пользователях компьютера и их правах доступа, реализует настройки идентификации и аутентификации пользователей, архивацию системы Windows, сопутствующие настройки политики безопасности, настройки облачного хранения данных в системе OneDrive.

Раздел «Время и язык» позволяет настроить дату и время, а также некоторые параметры, которые связаны с особенностями языка и страны вашего проживания. Например, вы можете установить формат чисел и дат, выбрать используемую денежную единицу. Также вы можете установить текущую дату и время. Кроме того, у вас есть возможность использовать несколько языков в работе, например русский и английский. Здесь вы можете выбрать дополнительные языки, а также указать способ переключения между языками.

Раздел «Игры» предназначен для дополнительной настройки системы для использования компьютера в различных играх, в том числе настройки дополнительных клавиш, джойстиков и геймпадов.

Раздел «Специальные возможности» содержит инструменты и их настройки для людей с ограниченными способностями и нарушениями зрения и слуха. Вы можете выбрать крупный шрифт надписей, высокую контрастность изображения, звуковые эффекты, а также настроить экранную клавиатуру.

Раздел «Конфиденциальность и защита» содержит соответствующие настройки в области безопасности данных, в том числе настройки файрвола, встроенного в систему антивируса. Включает функции диагностики системы, писка устройства, ведения журнала действий. Предоставляет возможность шифрования пользовательских данных, выдачи отдельным приложениям разрешений на соответствующие действия и доступ к данным пользователя.

Раздел «Центр обновлений Windows» предоставляет пользователю информацию о текущих обновлениях безопасности Windows, а также позволяет пользователю производить соответствующую настройку системы на получение и установку обновлений системы, производить откат неудачно установленных обновлений в ручную, производить анализ установленных обновлений системы.

Можно отметить, что в новой версии Windows 11 работа по

настройке системы с помощью панели управления стала очень удобной. Все похожие настройки сосредоточены в одном месте, тут же есть ссылки на связанные по смыслу настройки из различных групп. Выбор нужных настроек сводится к выбору группы и последующему выбору нужной задачи.

### 1.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

### 1.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Понятие интерфейса. Пользовательский интерфейс.
2. Сущность и назначение сервисных программ.
3. Для чего предназначены утилиты? На какие группы их можно разделить?
4. Что представляет собой Рабочий стол Windows?
5. Что представляет собой Панель задач? Где она располагается и как она настраивается?

## **2. УПРАВЛЕНИЕ ПРОЦЕССАМИ С ПОМОЩЬЮ КОМАНД ОПЕРАЦИОННОЙ СИСТЕМЫ ДЛЯ РАБОТЫ С ПРОЦЕССАМИ**

### **2.1 ЦЕЛЬ РАБОТЫ**

Цель работы – изучение возможностей контроля и управления процессами в операционных системах Windows, научиться работать с Диспетчером задач, ознакомиться с управлением процессами в ОС Windows с помощью утилиты Process Explorer.

### **2.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

Для правильного выполнения той или иной задачи в Windows необходимо, чтобы была запущена та или иная программа. В данной работе вы ознакомитесь с минимальным набором программ, которые должны быть запущены для корректной работы Windows. Для того чтобы увидеть полный список выполняемых задач в данный момент можно воспользоваться Диспетчером задач Windows или любой другой аналогичной программой (утилиты Process Explorer (procexp.exe.)). В этой работе мы ознакомимся только с Диспетчером задач, который можно запустить нажатием комбинации CTRL+ALT+DELETE.

В Диспетчере задач отображаются сведения о программах и процессах, выполняемых на компьютере.

На вкладке Приложения отображается состояние выполняющихся на компьютере программ. На этой вкладке имеется возможность завершить или запустить программу, а также перейти в окно программы.

На вкладке Процессы отображаются сведения о выполняющихся на компьютере процессах. Например, допускается отображение сведений об использовании ЦП и памяти, ошибках страницы, счетчике дескрипторов и некоторые другие параметры. На вкладке Быстродействие динамически отображаются следующие сведения о быстродействии компьютера.

1. Графики загрузки ЦП и использования памяти.

2. Общее число дескрипторов, процессов, выполняющихся на компьютере.

3. Общий объем физической памяти, памяти ядра и выделения памяти в килобайтах.

Если имеется подключение к сети, можно просматривать состояние сети и параметры ее работы.

Если к компьютеру подключились несколько пользователей, можно увидеть их имена, какие задачи они выполняют, а также отправить им сообщение.

Для того чтобы увидеть все программы, загруженные в оперативную память нужно перейти с вкладки Приложения на вкладку Процессы.

Перечисленные здесь процессы – это программы, которые на данный момент загружены в оперативную память. Это могут быть специальные служебные программы, без которых Windows не будет работать, программы, отвечающие за предоставление каких либо услуг, например сверка системного времени с сервером времени в сети Internet, и т.д. В ниже приведённой таблице есть сведения о названии некоторых процессов и назначение данной программы.

Процесс Бездействие системы представляет собой отдельный поток, выполняющийся на каждом процессоре и имеющий единственную задачу – заполнение процессорного времени, когда система не обрабатывает другие потоки. В Диспетчере задач данный процесс занимает большую часть процессорного времени.

Имя процесса и его описание.

*Explorer.exe*

Программа проводник, отвечает за отображение на экране рабочего стола, открытие главного меню (если открываете окно проводника, появляется ещё один процесс).

*Spoolsv.exe*

Программа отвечает за очередь печати (постановка документов в очередь, удаление очереди отслеживание количества напечатанных листов).

*services.exe*

Позволяет компьютеру распознавать изменения в установленном оборудовании и подстраиваться под них, либо не требуя вмешательства пользователя, либо сводя его к минимуму. Оста-

новка или отключение этой службы может привести к нестабильной работе системы.(Plug and Play). А так же обеспечивает поддержку сообщений журналов событий, выдаваемых Windows-программами и компонентами системы, и просмотр этих сообщений.

*svchost.exe*

Позволяет настраивать расписание автоматического выполнения задач на этом компьютере.

*svchost.exe*

Управляет объектами папки "Сеть и удаленный доступ к сети", отображающей свойства локальной сети и подключений удаленного доступа.

*svchost.exe*

Управляет синхронизацией даты и времени на всех клиентах и серверах в сети. Если эта служба остановлена, синхронизация даты и времени не будет доступна.

*svchost.exe*

Обеспечивает поддержку общий доступ к файлам, принтерам и именованным каналам для данного компьютера через сетевое подключение. Если служба остановлена, такие функции не удастся выполнить.

*svchost.exe*

Позволяет удаленным пользователям изменять параметры реестра на этом компьютере.

*mdm.exe*

Управляет местной и удаленной отладкой для отладчиков Visual Studio

*lsass.exe*

Хранит информацию о безопасности для учетной записи локального пользователя.

*Winlogon.exe*

Программа входа в систему Windows NT. Изменение вида окна Диспетчера задач, выбор для отображения тех или иных параметров производится с помощью пунктов меню. Всю информацию о работе с Диспетчером задач можно найти в пункте меню «Справка».

Управление процессами и потоками в ОС Windows с помощью утилиты Process Explorer фирмы SysInternals. Утилита пока-

зывает не просто список активных процессов, но и файлы динамических библиотек, связанные с процессом, приоритет процесса, нагрузку на процессор отдельно для каждой программы и т.д. Помимо этого, с помощью программы можно изменить приоритет процесса, просмотреть информацию о DLL-файле и принудительно завершить безнадежно зависшую программу.

Утилита содержит 2 окна. В верхнем отображается список активных процессов (в т.ч. идентификатор процесса – PID, процент загрузки процессора – CPU, описание – Description, наименование аккаунта владельца – Owner, приоритет процесса – Priority, Handles, Windows Title). Информация, показываемая в нижнем окне, зависит от режима Process Explorer – если он находится в режиме handle mode, Вы можете видеть handles (файлы для Windows 9x/Me), которые открыл процесс, выбранный в верхнем окне; если это режим DLL (DLL mode) – Вы можете видеть DLL, которые загрузил данный процесс. Переключение между режимами осуществляется "горячими клавишами" или с помощью соответствующих пунктов меню: Вы можете сортировать процессы по любому критерию, щелкая мышкой на соответствующей колонке; либо представить процессы в виде дерева процессов (process tree) путем выбора пункта меню View – Show Process Tree. Щелкнув правой кнопкой мыши по выбранному процессу, с помощью появившегося контекстного меню Вы можете изменить базовый приоритет процесса (Set Priority), принудительно завершить процесс (Kill Process) и просмотреть дополнительные параметры процесса (Properties): С помощью пункта меню Options – Highlight Services можно выделить процессы, которые обслуживают хост. Для выделения процессов текущего пользователя выберите пункт меню Options – Highlight Own Processes. Запустив утилиту, запустите несколько приложений (например, Far, Word, Paint, Notepad и т.д.), обратите внимание на изменения в окне процессов. Прокомментируйте их. Приведите копию экрана и опишите процесс, порожденный запущенным приложением.

*Задания. Копии экрана с выполненным заданием и описание выполненных действий привести в отчете.*



Задание 1. На вкладке Процессы Диспетчера задач измените количество столбцов, запишите выполненные для этого операции. Какие из процессов запущены Пользователем?

Задание 2. Сколько процессов активно на момент выполнения практической работы, насколько загружен центральный процессор, какой объем памяти выделен на текущие процессы?

Задание 3. Просмотреть справочную систему Диспетчера задач. Найти информацию о запуске новых программ, завершении текущих программ с использованием Диспетчера и выписать в тетрадь.

Задание 4. Выполните следующие действия с помощью утилиты Process Explorer. Отсортируйте процессы по заданному критерию. Опишите один из системных процессов. Запустите указанное приложение. Опишите возникший процесс по заданным характеристикам. Принудительно завершите указанный процесс. Выполняемые действия иллюстрируйте копиями экранов.

## 2.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

## 2.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое процесс?
2. Опишите общие сведения о Диспетчере задач?
3. Что означает параметр «бездействие системы»?
4. Можно ли изменить внешний вид вкладки Процессы в Диспетчере задач?
5. Как завершить процесс?
6. Опишите возможности работы с помощью утилиты Process Explorer.
7. Записать процессы и их описание из таблицы в тетрадь.

### **3. ИССЛЕДОВАНИЕ СООТНОШЕНИЯ МЕЖДУ ПРЕДСТАВЛЯЕМЫМ И ИСТИННЫМ ОБЪЁМОМ ЗАНЯТОЙ ДИСКОВОЙ ПАМЯТИ. ИЗУЧЕНИЕ ВЛИЯНИЯ КОЛИЧЕСТВА ФАЙЛОВ НА ВРЕМЯ, НЕОБХОДИМОЕ ДЛЯ ИХ КОПИРОВАНИЯ**

#### **3.1 ЦЕЛЬ РАБОТЫ**

Цель работы – изучение методов распределения памяти с использованием дискового пространства и без использования дискового пространства.

#### **3.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

*3.2.1 Основные понятия. Классификация методов распределения памяти*

Все методы управления памятью могут быть разделены на два класса: методы, которые используют перемещение процессов между оперативной памятью и диском, и методы, которые не делают этого. Начнем с последнего, более простого класса методов.

Методы без использования внешней памяти делятся на три класса: фиксированными разделами, динамическими разделами, перемещаемыми разделами.

Методы с использованием внешней памяти также делятся на три класса: страничное распределение, сегментное распределение, сегментно-страничное распределение.

Самым простым способом управления оперативной памятью является разделение ее на несколько разделов фиксированной величины. Это может быть выполнено вручную оператором во время старта системы или во время ее генерации. Очередная задача, поступившая на выполнение, помещается либо в общую очередь (рисунок 10,а), либо в очередь к некоторому разделу (рисунок 10,б).

Подсистема управления памятью в этом случае выполняет следующие задачи:

- сравнивая размер программы, поступившей на выполнение, и свободных разделов, выбирает подходящий раздел,
- осуществляет загрузку программы и настройку адресов.

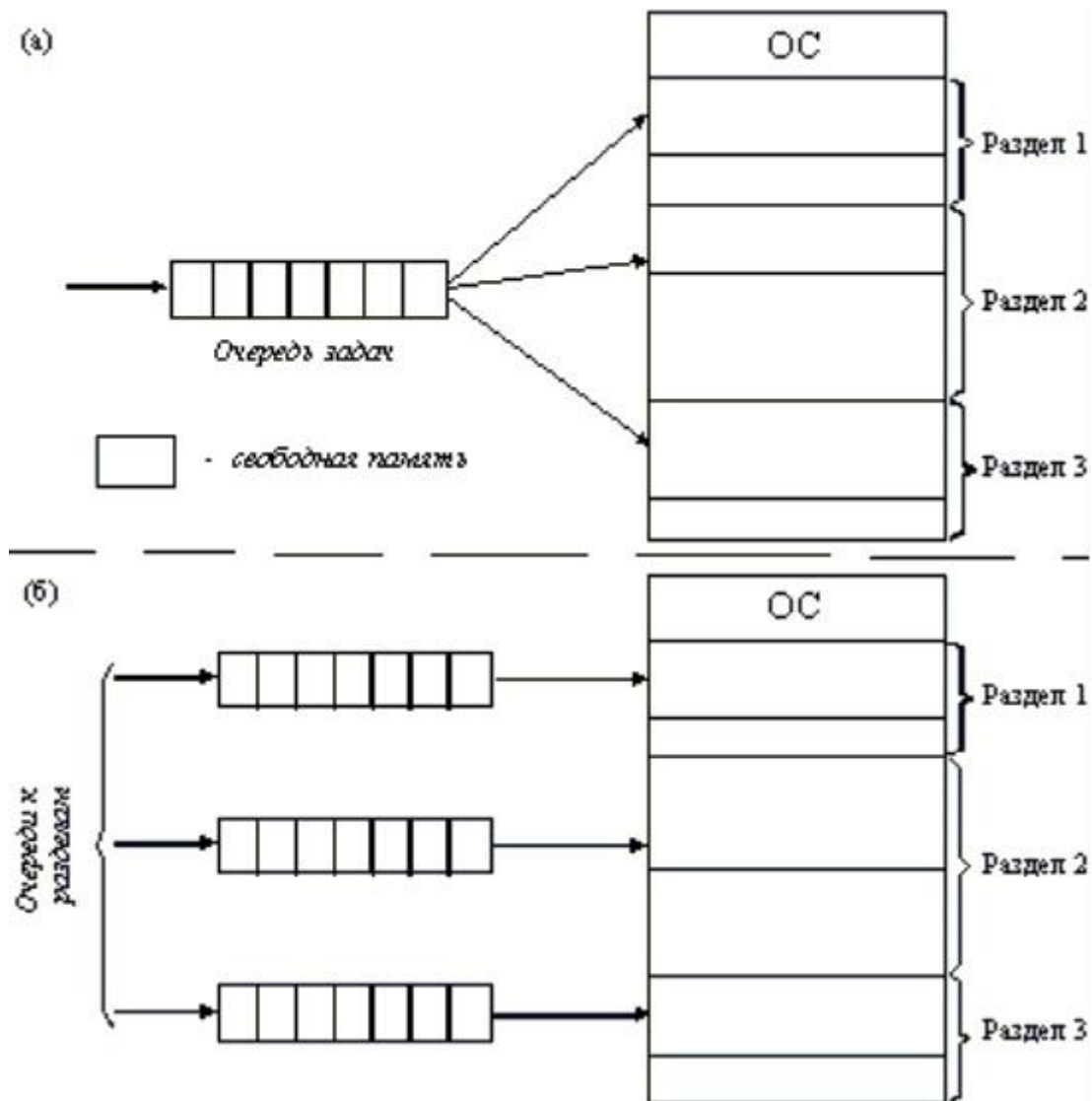


Рисунок 10 – Распределение памяти фиксированными разделами:

а – с общей очередью; б – с отдельными очередями

При очевидном преимуществе – простоте реализации – данный метод имеет существенный недостаток – жесткость. Так как в каждом разделе может выполняться только одна программа, то уровень мультипрограммирования заранее ограничен числом разделов независимо от того, какой размер имеют программы. Даже если программа имеет небольшой объем, она будет занимать весь раздел, что приводит к неэффективному использованию памяти. С другой стороны, даже если объем оперативной памяти машины позволяет выполнить некоторую программу, разбиение памяти на разделы не позволяет сделать этого.

В этом случае память машины не делится заранее на разделы. Сначала вся память свободна. Каждой вновь поступающей задаче выделяется необходимая ей память. Если достаточный объем памяти отсутствует, то задача не принимается на выполнение и стоит в очереди. После завершения задачи память освобождается, и на это место может быть загружена другая задача.

Таким образом, в произвольный момент времени оперативная память представляет собой случайную последовательность занятых и свободных участков (разделов) произвольного размера. На рисунке 11 показано состояние памяти в различные моменты времени при использовании динамического распределения. Так в момент  $t_0$  в памяти находится только ОС, а к моменту  $t_1$  память разделена между 5 задачами, причем задача П4, завершаясь, покидает память. На освободившееся после задачи П4 место загружается задача П6, поступившая в момент  $t_3$ .

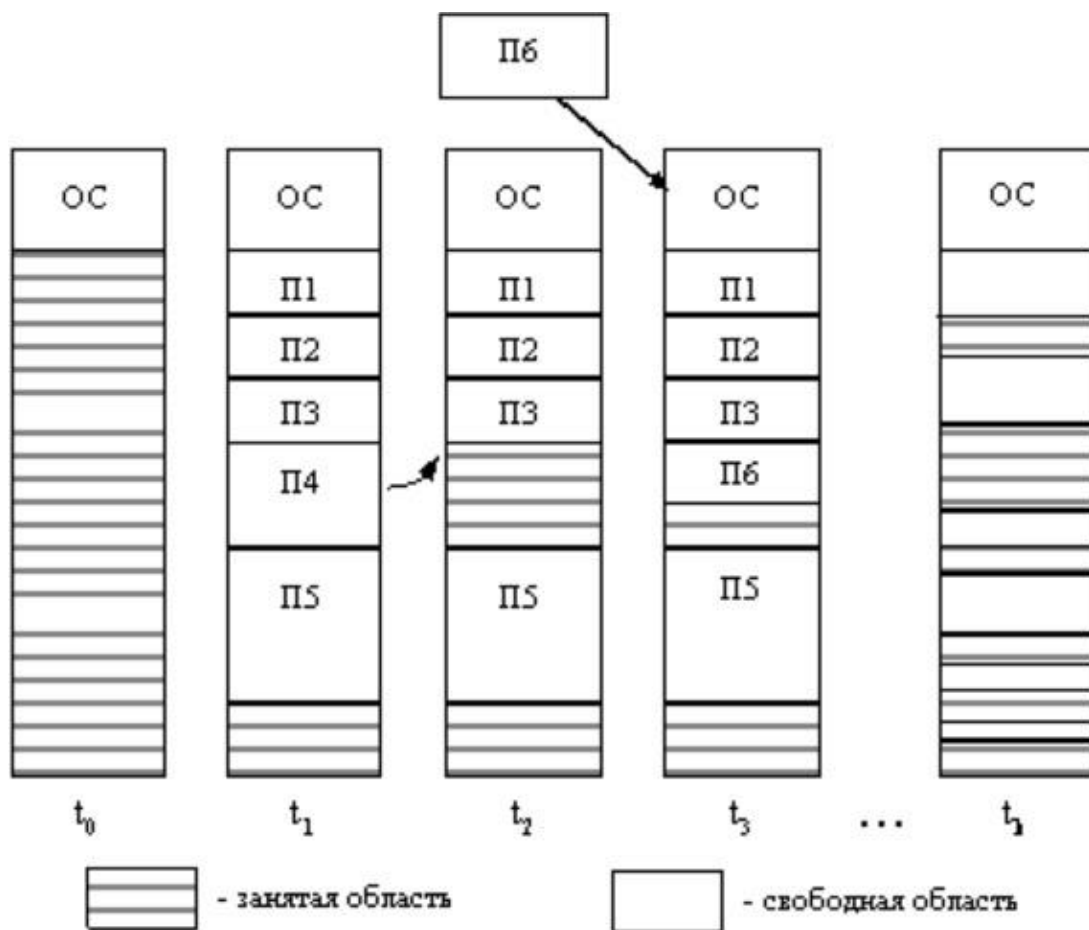


Рисунок 11 – Распределение памяти динамическими разделами

Задачами операционной системы при реализации данного метода управления памятью являются:

- ведение таблиц свободных и занятых областей, в которых указываются начальные адреса и размеры участков памяти,
- при поступлении новой задачи – анализ запроса, просмотр таблицы свободных областей и выбор раздела, размер которого достаточен для размещения поступившей задачи,
- загрузка задачи в выделенный ей раздел и корректировка таблиц свободных и занятых областей,
- после завершения задачи корректировка таблиц свободных и занятых областей.

Программный код не перемещается во время выполнения, то есть может быть проведена единовременная настройка адресов посредством использования перемещающего загрузчика.

Выбор раздела для вновь поступившей задачи может осуществляться по разным правилам, таким, например, как "первый попавшийся раздел достаточного размера", или "раздел, имеющий наименьший достаточный размер", или "раздел, имеющий наибольший достаточный размер". Все эти правила имеют свои преимущества и недостатки.

По сравнению с методом распределения памяти фиксированными разделами данный метод обладает гораздо большей гибкостью, но ему присущ очень серьезный недостаток – фрагментация памяти. Фрагментация – это наличие большого числа несмежных участков свободной памяти очень маленького размера (фрагментов). Настолько маленького, что ни одна из вновь поступающих программ не может поместиться ни в одном из участков, хотя суммарный объем фрагментов может составить значительную величину, намного превышающую требуемый объем памяти.

Одним из методов борьбы с фрагментацией является перемещение всех занятых участков в сторону старших либо в сторону младших адресов, так, чтобы вся свободная память образовывала единую свободную область (рисунок 12). В дополнение к функциям, которые выполняет ОС при распределении памяти переменными разделами, в данном случае она должна еще время от времени копировать содержимое разделов из одного места памя-

ти в другое, корректируя таблицы свободных и занятых областей. Эта процедура называется "сжатием". Сжатие может выполняться либо при каждом завершении задачи, либо только тогда, когда для вновь поступившей задачи нет свободного раздела достаточного размера. В первом случае требуется меньше вычислительной работы при корректировке таблиц, а во втором – реже выполняется процедура сжатия. Так как программы перемещаются по оперативной памяти в ходе своего выполнения, то преобразование адресов из виртуальной формы в физическую должно выполняться динамическим способом.

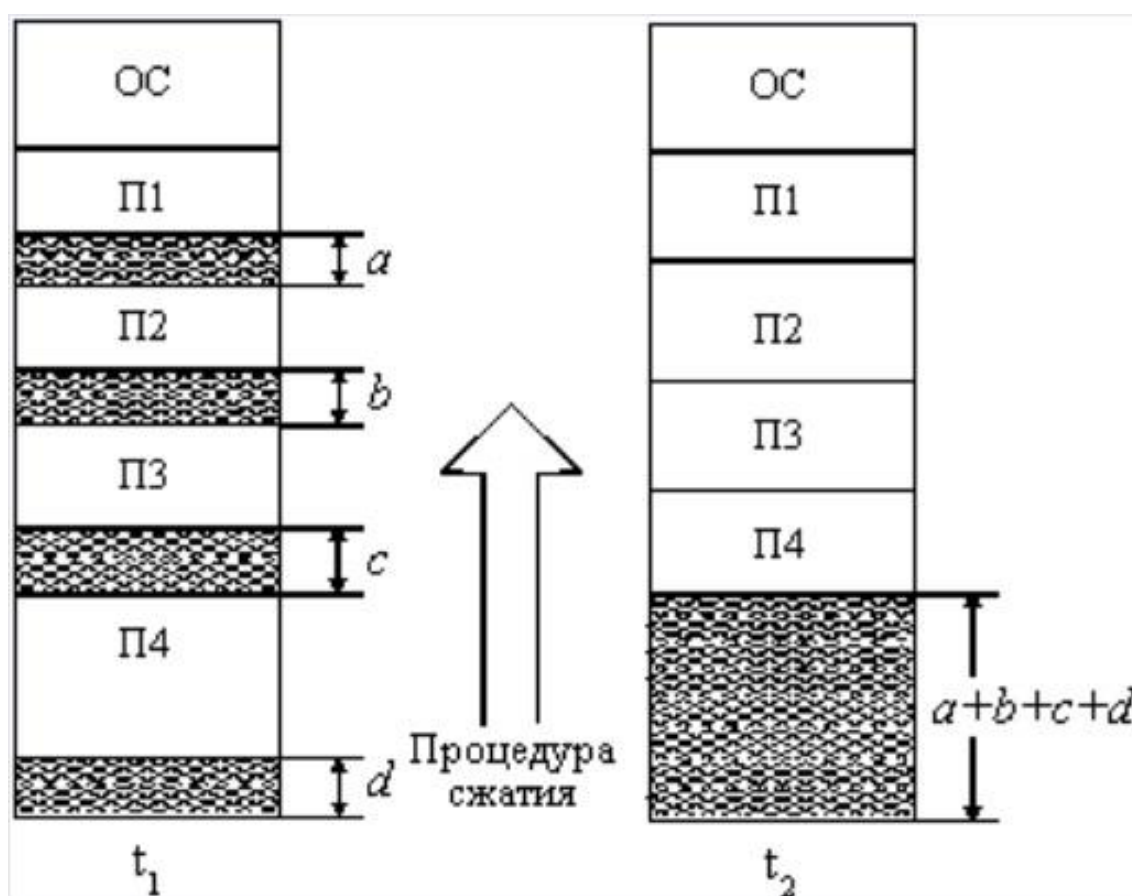


Рисунок 12 – Распределение памяти перемещаемыми разделами

Хотя процедура сжатия и приводит к более эффективному использованию памяти, она может потребовать значительного времени, что часто перевешивает преимущества данного метода.

### 3.2.2 Понятие виртуальной памяти

Уже достаточно давно пользователи столкнулись с проблемой размещения в памяти программ, размер которых превышал

имеющуюся в наличии свободную память. Решением было разбиение программы на части, называемые *оверлеями*. 0-ой оверлей начинал выполняться первым. Когда он заканчивал свое выполнение, он вызывал другой оверлей. Все оверлеи хранились на диске и перемещались между памятью и диском средствами операционной системы. Однако разбиение программы на части и планирование их загрузки в оперативную память должен был осуществлять программист.

Развитие методов организации вычислительного процесса в этом направлении привело к появлению метода, известного под названием виртуальная память. Виртуальным называется ресурс, который пользователю или пользовательской программе представляется обладающим свойствами, которыми он в действительности не обладает. Так, например, пользователю может быть предоставлена виртуальная оперативная память, размер которой превосходит всю имеющуюся в системе реальную оперативную память. Пользователь пишет программы так, как будто в его распоряжении имеется однородная оперативная память большого объема, но в действительности все данные, используемые программой, хранятся на одном или нескольких разнородных запоминающих устройствах, обычно на дисках, и при необходимости частями отображаются в реальную память.

Таким образом, виртуальная память – это совокупность программно-аппаратных средств, позволяющих пользователям писать программы, размер которых превосходит имеющуюся оперативную память; для этого виртуальная память решает следующие задачи:

- размещает данные в запоминающих устройствах разного типа, например, часть программы в оперативной памяти, а часть на диске;
- перемещает по мере необходимости данные между запоминающими устройствами разного типа, например, подгружает нужную часть программы с диска в оперативную память;
- преобразует виртуальные адреса в физические.

Все эти действия выполняются автоматически, без участия программиста, то есть механизм виртуальной памяти является прозрачным по отношению к пользователю.

Наиболее распространенными реализациями виртуальной памяти является страничное, сегментное и странично-сегментное распределение памяти, а также свопинг.

### *3.2.3 Страничное распределение памяти*

На рисунке 4 показана схема страничного распределения памяти. Виртуальное адресное пространство каждого процесса делится на части одинакового, фиксированного для данной системы размера, называемые виртуальными страницами. В общем случае размер виртуального адресного пространства не является кратным размеру страницы, поэтому последняя страница каждого процесса дополняется фиктивной областью.

Вся оперативная память машины также делится на части такого же размера, называемые физическими страницами (или блоками).

Размер страницы обычно выбирается равным степени двойки: 512, 1024 и т.д., это позволяет упростить механизм преобразования адресов.

При загрузке процесса часть его виртуальных страниц помещается в оперативную память, а остальные – на диск. Смежные виртуальные страницы не обязательно располагаются в смежных физических страницах. При загрузке операционная система создает для каждого процесса информационную структуру – таблицу страниц, в которой устанавливается соответствие между номерами виртуальных и физических страниц для страниц, загруженных в оперативную память, или делается отметка о том, что виртуальная страница выгружена на диск. Кроме того, в таблице страниц содержится управляющая информация, такая как признак модификации страницы, признак невыгружаемости (выгрузка некоторых страниц может быть запрещена), признак обращения к странице (используется для подсчета числа обращений за определенный период времени) и другие данные, формируемые и используемые механизмом виртуальной памяти.

При активизации очередного процесса в специальный регистр процессора загружается адрес таблицы страниц данного процесса.

При каждом обращении к памяти происходит чтение из таблицы страниц информации о виртуальной странице, к которой



произошло обращение. Если данная виртуальная страница находится в оперативной памяти, то выполняется преобразование виртуального адреса в физический.

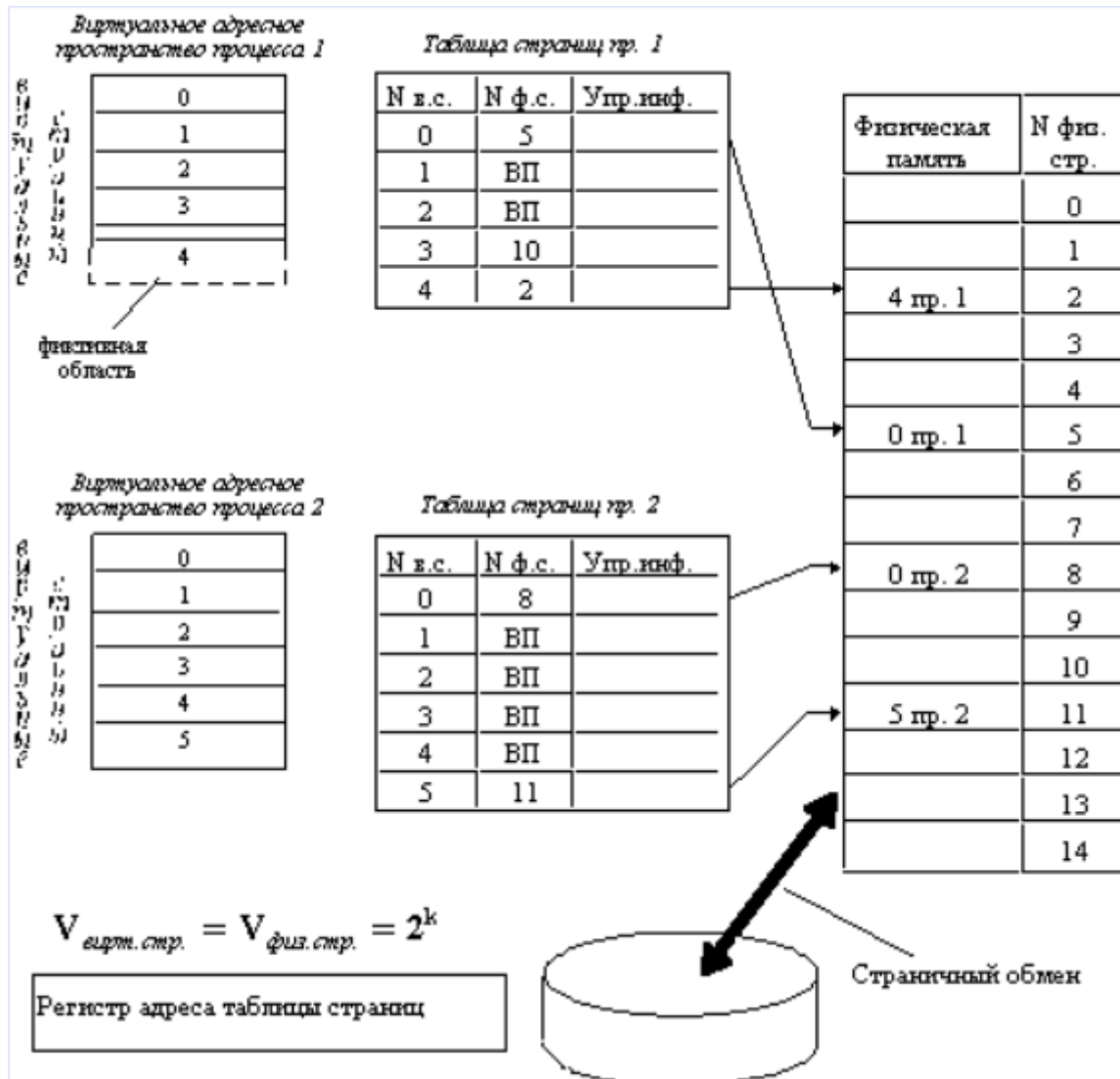


Рисунок 13 – Страничное распределение памяти

Если же нужная виртуальная страница в данный момент выгружена на диск, то происходит так называемое страничное прерывание. Выполняющийся процесс переводится в состояние ожидания, и активизируется другой процесс из очереди готовых процессов. Параллельно программа обработки страничного прерывания находит на диске требуемую виртуальную страницу и пытается загрузить ее в оперативную память. Если в памяти имеется свободная физическая страница, то загрузка выполняется

немедленно, если же свободных страниц нет, то решается вопрос, какую страницу следует выгрузить из оперативной памяти.

В данной ситуации может быть использовано много разных критериев выбора, наиболее популярные из них следующие:

- дольше всего не использовавшаяся страница,
- первая попавшаяся страница,
- страница, к которой в последнее время было меньше всего обращений.

В некоторых системах используется понятие рабочего множества страниц. Рабочее множество определяется для каждого процесса и представляет собой перечень наиболее часто используемых страниц, которые должны постоянно находиться в оперативной памяти и поэтому не подлежат выгрузке.

После того, как выбрана страница, которая должна покинуть оперативную память, анализируется ее признак модификации (из таблицы страниц). Если выталкиваемая страница с момента загрузки была модифицирована, то ее новая версия должна быть переписана на диск. Если нет, то она может быть просто уничтожена, то есть соответствующая физическая страница объявляется свободной.

Рассмотрим механизм преобразования виртуального адреса в физический при страничной организации памяти (рисунок 14).

Виртуальный адрес при страничном распределении может быть представлен в виде пары  $(p, s)$ , где  $p$  – номер виртуальной страницы процесса (нумерация страниц начинается с 0), а  $s$  – смещение в пределах виртуальной страницы. Учитывая, что размер страницы равен  $2^k$ , смещение  $s$  может быть получено простым отделением  $k$  младших разрядов в двоичной записи виртуального адреса. Оставшиеся старшие разряды представляют собой двоичную запись номера страницы  $p$ .

При каждом обращении к оперативной памяти аппаратными средствами выполняются следующие действия:

1. на основании начального адреса таблицы страниц (содержимое регистра адреса таблицы страниц), номера виртуальной страницы (старшие разряды виртуального адреса) и длины записи в таблице страниц (системная константа) определяется адрес нужной записи в таблице,



Рисунок 14 – Механизм преобразования виртуального адреса в физический при страничной организации памяти

2. из этой записи извлекается номер физической страницы,
3. к номеру физической страницы присоединяется смещение (младшие разряды виртуального адреса).

Использование в пункте (3) того факта, что размер страницы равен степени 2, позволяет применить операцию конкатенации (присоединения) вместо более длительной операции сложения, что уменьшает время получения физического адреса, а значит повышает производительность компьютера.

На производительность системы со страничной организацией памяти влияют временные затраты, связанные с обработкой

страничных прерываний и преобразованием виртуального адреса в физический.

При часто возникающих страничных прерываниях система может тратить большую часть времени впустую, на свопинг страниц. Чтобы уменьшить частоту страничных прерываний, следовало бы увеличивать размер страницы. Кроме того, увеличение размера страницы уменьшает размер таблицы страниц, а значит, уменьшает затраты памяти. С другой стороны, если страница велика, значит, велика и фиктивная область в последней виртуальной странице каждой программы. В среднем на каждой программе теряется половина объема страницы, что в сумме при большой странице может составить существенную величину. Время преобразования виртуального адреса в физический в значительной степени определяется временем доступа к таблице страниц. В связи с этим таблицу страниц стремятся размещать в "быстрых" запоминающих устройствах. Это может быть, например, набор специальных регистров или память, использующая для уменьшения времени доступа ассоциативный поиск и кэширование данных.

Страничное распределение памяти может быть реализовано в упрощенном варианте, без выгрузки страниц на диск. В этом случае все виртуальные страницы всех процессов постоянно находятся в оперативной памяти. Такой вариант страничной организации хотя и не предоставляет пользователю виртуальной памяти, но почти исключает фрагментацию за счет того, что программа может загружаться в несмежные области, а также того, что при загрузке виртуальных страниц никогда не образуется остатков.

#### *3.2.4 Сегментное распределение*

При страничной организации виртуальное адресное пространство процесса делится механически на равные части. Это не позволяет дифференцировать способы доступа к разным частям программы (сегментам), а это свойство часто бывает очень полезным. Например, можно запретить обращаться с операциями записи и чтения в кодовый сегмент программы, а для сегмента данных разрешить только чтение. Кроме того, разбиение программы на "осмысленные" части делает принципиально возмож-

ным разделением одного сегмента несколькими процессами. Например, если два процесса используют одну и ту же математическую подпрограмму, то в оперативную память может быть загружена только одна копия этой подпрограммы.

Рассмотрим, каким образом сегментное распределение памяти реализует эти возможности (рисунок 15). Виртуальное адресное пространство процесса делится на сегменты, размер которых определяется программистом с учетом смыслового значения содержащейся в них информации. Отдельный сегмент может представлять собой подпрограмму, массив данных и т.п. Иногда сегментация программы выполняется по умолчанию компилятором.

При загрузке процесса часть сегментов помещается в оперативную память (при этом для каждого из этих сегментов операционная система подыскивает подходящий участок свободной памяти), а часть сегментов размещается в дисковой памяти. Сегменты одной программы могут занимать в оперативной памяти несмежные участки.

Во время загрузки система создает таблицу сегментов процесса (аналогичную таблице страниц), в которой для каждого сегмента указывается начальный физический адрес сегмента в оперативной памяти, размер сегмента, правила доступа, признак модификации, признак обращения к данному сегменту за последний интервал времени и некоторая другая информация. Если виртуальные адресные пространства нескольких процессов включают один и тот же сегмент, то в таблицах сегментов этих процессов делаются ссылки на один и тот же участок оперативной памяти, в который данный сегмент загружается в единственном экземпляре.

Система с сегментной организацией функционирует аналогично системе со страничной организацией: время от времени происходят прерывания, связанные с отсутствием нужных сегментов в памяти, при необходимости освобождения памяти некоторые сегменты выгружаются, при каждом обращении к оперативной памяти выполняется преобразование виртуального адреса в физический. Кроме того, при обращении к памяти проверяется, разрешен ли доступ требуемого типа к данному сегменту.

Виртуальный адрес при сегментной организации памяти может быть представлен парой  $(g, s)$ , где  $g$  – номер сегмента, а  $s$  – смещение в сегменте. Физический адрес получается путем сложения начального физического адреса сегмента, найденного в таблице сегментов по номеру  $g$ , и смещения  $s$ .

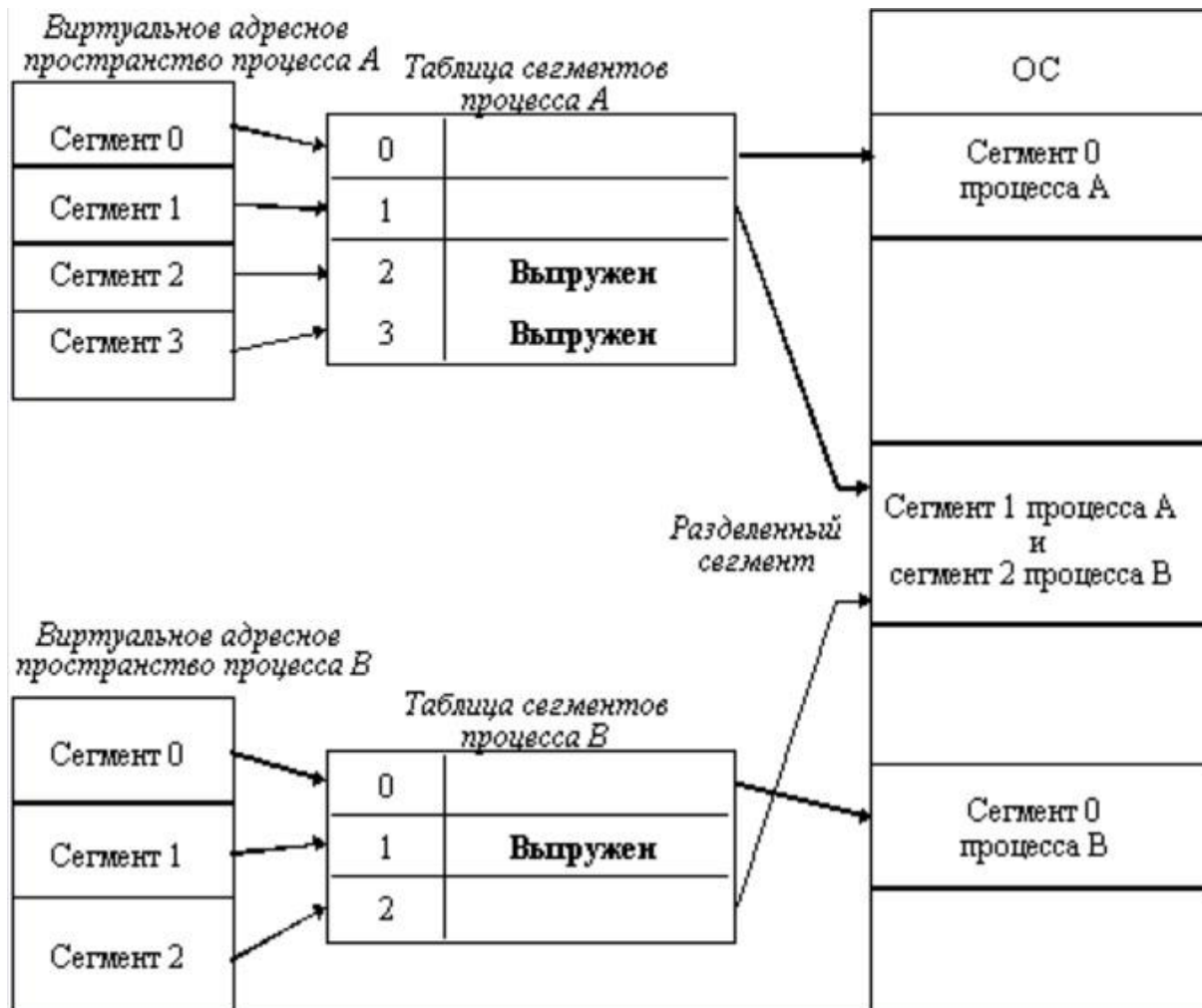


Рисунок 15 – Распределение памяти сегментами

Недостатком данного метода распределения памяти является фрагментация на уровне сегментов и более медленное по сравнению со страничной организацией преобразование адреса.

### 3.2.5 Странично-сегментное распределение

Как видно из названия, данный метод представляет собой комбинацию страничного и сегментного распределения памяти и, вследствие этого, сочетает в себе достоинства обоих подходов.

Виртуальное пространство процесса делится на сегменты, а каждый сегмент – на виртуальные страницы, которые нумеруются в пределах сегмента. Оперативная память делится на физические страницы. Загрузка процесса выполняется постранично, при этом часть страниц размещается в оперативной памяти, а часть на диске. Для каждого сегмента создается своя таблица страниц, структура которой совпадает со структурой таблицы страниц, используемой при страничном распределении. Для каждого процесса создается таблица сегментов, в которой указываются адреса таблиц страниц для всех сегментов данного процесса. Адрес таблицы сегментов загружается в специальный регистр процессора, когда активизируется соответствующий процесс. На рисунке 16 показана схема преобразования виртуального адреса в физический.



Рисунок 16 – Схема преобразования виртуального адреса в физический для сегментно-страничной организации памяти

### 3.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

### 3.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Классификация методов распределения памяти.
2. Распределение памяти фиксированными разделами
3. Распределение памяти динамическими разделами?
4. Распределение памяти перемещаемыми разделами.
5. Страничное распределение памяти.
6. Сегментное распределение памяти.
7. Странично-сегментное распределение памяти.



## 4. УПРАВЛЕНИЕ ПАМЯТЬЮ

### 4.1 ЦЕЛЬ РАБОТЫ

Цель работы – Изучение способов и методов организации памяти вычислительной машины.

### 4.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

#### *4.2.1 Понятие об организации и управлении физической памятью в операционных системах*

Организация и управление основной (первичной, физической, реальной) памятью вычислительной машины – один из важнейших факторов, определяющих построение операционных систем. В англоязычной технической литературе память обозначается синонимами *memory* и *storage*.

В операционных системах различают два вида памяти: основная (первичная) и внешняя (вторичная).

Основная память (*main storage*) – оперативная память центрального процессора или ее часть, представляющее собой единое пространство памяти.

Внешняя память (*external storage*) – память, данные в которой доступны центральному процессору посредством операций ввода-вывода.

Для непосредственного выполнения программ или обращения к данным необходимо, чтобы они размещались в основной памяти. Внешняя память имеет, как правило, большую емкость, чем основная, стоит дешевле и позволяет хранить данные и программы, которые должны быть наготове для обработки.

Кроме основной и внешней памяти в современных ЭВМ существует дополнительная быстродействующая память, называемая кэш-памятью.

Все три перечисленных вида памяти образуют иерархию памяти вычислительной машины (рисунок 17).

## Иерархия памяти ЭВМ

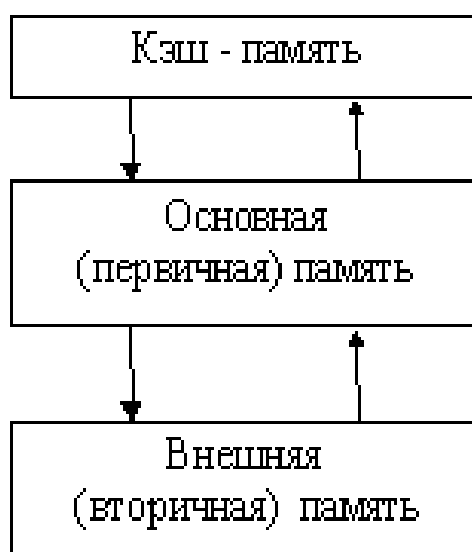


Рисунок 17 – Иерархия памяти вычислительной машины

Операционным системам с несколькими уровнями иерархии памяти свойственна высокая интенсивность челночных обменов программами и данными между физическими устройствами памяти различных уровней. Такие обмены отнимают системные ресурсы (например, время центрального процессора), которые можно было бы использовать более продуктивно.

Основная память представляет собой один из самых дорогостоящих ресурсов. Главной задачей при разработке ОС считается оптимальное использование основной памяти на основе рациональной организации и управления ею.

Под организацией памяти понимается то, каким образом представляется и как используется основная память.

В операционных системах применяются следующие виды представления основной памяти:

- фиксированными блоками равного размера;
- фиксированными разделами неодинакового размера;
- динамическими разделами, размеры которых изменяются в ходе работы вычислительной системы.

Использование основной памяти может осуществляться следующими способами:

- размещение в памяти одновременно только одной программы пользователей;
- размещение в памяти одновременно нескольких программ пользователей;
- размещение программ пользователей в конкретном заранее заданном разделе основной памяти;
- размещение каждой программы пользователя в одном непрерывном (односвязном) пространстве основной памяти;
- размещение программы пользователя в несмежных областях оперативной памяти (при этом ОС осуществляет разбиение размещаемых там программ на отдельные блоки и обеспечивает связь этих блоков между собой).

В операционных системах может применяться любая комбинация перечисленных видов представления и способов использования основной памяти ЭВМ.

Независимо от того, какая схема организации памяти принята для конкретной ОС, необходимо решить, какие стратегии следует применять для достижения оптимальных характеристик.

*Стратегии управления памятью* определяют, как будет работать память с конкретной схемой организации при различных подходах к решению следующих вопросов:

- когда следует поместить новую программу в память;
- какое место основной памяти будет размещаться очередная программа;
- как разместить очередную программу в памяти (с минимизацией потерь памяти или с максимизацией скорости размещения);
- какую из находящихся в памяти программ следует вывести из памяти, если необходимо обязательно разместить новую программу, а память уже заполнена.

В существующих ОС реализованы стратегии управления, по-разному отвечающие на перечисленные выше вопросы, что в немалой степени обусловлено имеющимися в распоряжении разработчиков аппаратурными и программными средствами.

Стратегии управления памятью делятся на следующие категории:

- стратегии выборки;
- стратегии размещения;

– стратегии замещения.

В свою очередь стратегии выборки разделяют на две подкатегории:

- стратегии выборки по запросу (по требованию);
- стратегии упреждающей выборки.

*Стратегии выборки* ставят своей целью определить, когда следует “втолкнуть” очередную программу (или блок программы) или данные в основную память.

*Стратегии размещения* ставят своей целью определить, в какое место основной памяти следует размещать поступающую программу. Наиболее распространенными являются стратегии размещения, реализующие принципы занятия “первого подходящего”, “наиболее подходящего” и “наименее подходящего” по размерам свободного участка памяти.

*Стратегии замещения* ставят своей целью определить, какой блок программы или данных следует вывести (“вытолкнуть”) из основной памяти, чтобы освободить место для размещения вновь поступающих программ или данных.

При реализации стратегий размещения операционные системы часто учитывают требования связного распределения памяти для программ и данных.

*Связное распределение памяти* – такое распределение основной памяти ЭВМ, при котором каждая программа занимает один непрерывный (связный) блок ячеек памяти.

*Несвязное распределение памяти* – такое распределение основной памяти ЭВМ, при котором программа пользователя разбивается на ряд блоков (сегментов, страниц), которые могут размещаться в основной памяти в участках, не обязательно соседствующих друг с другом (в несмежных участках). В этом случае обеспечивается более эффективное использование пространства основной памяти.

Эффективность той или иной стратегии размещения можно оценить с помощью коэффициента использования памяти  $h$

$$h = \frac{V_{\Pi}}{V_{\text{ОП}} - V_{\text{ОС}}} = \frac{V_{\Pi}}{V_0}; \quad (1)$$

где  $V_{\Pi}$  – объем памяти, занимаемый программами пользователя;

$V_{\text{ОП}}$  – полный объем основной памяти;

$V_{oc}$  – объем памяти, занимаемый операционной системой;  
 $V_o$  – объем памяти, доступный для распределения.

#### 4.2.2 Методы связного распределения основной памяти

##### *Связное распределение памяти для одного пользователя*

Связное распределение памяти для одного пользователя, называемое также одиночным непрерывным распределением, применяется в ЭВМ, работающих в пакетном однопрограммном режиме под управлением простейшей ОС.

Вся основная часть ЭВМ, не занятая программами операционной системы, выделяется программе единственного на данном отрезке времени пользователя. Размер программы в этом случае ограничивается размером доступной основной памяти, но существует возможность выполнения программ, размер которых превышает размер основной памяти, используя механизм оверлеев.

Организация памяти при связном распределении для одного пользователя показана на рисунке 18.

Связное распределение памяти  
для одного пользователя

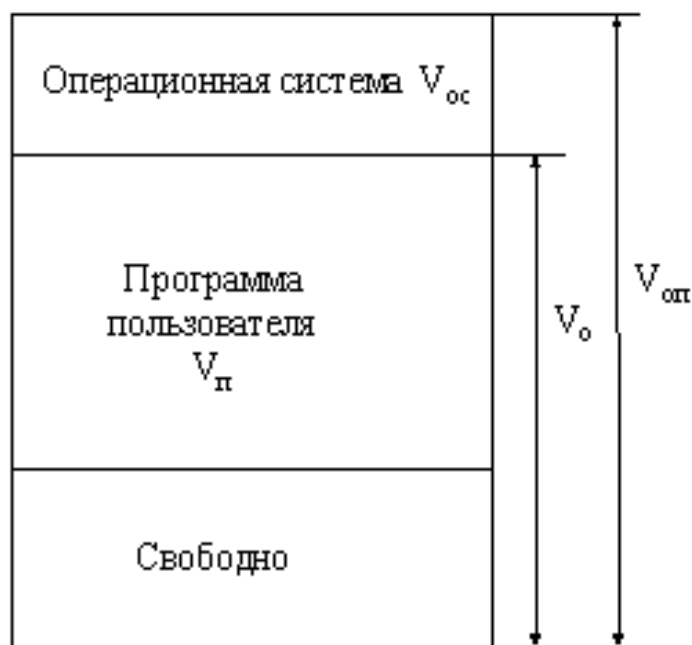


Рисунок 18 – Организация памяти при связном распределении для одного пользователя

Коэффициент использования памяти для рассматриваемого случая вычисляется по формуле:

$$h_{c1} = \frac{V_{\Pi}}{V_0}; \quad (2)$$

где  $V_{\Pi}$  – размер программы пользователя;

$V_0$  – объем доступной для распределения основной памяти ЭВМ.

Функциями ОС в данном случае являются:

- выделение программе необходимого пространства памяти;
- защита памяти;
- освобождение памяти.

Функция выделения памяти сводится к предоставлению программе всей доступной памяти ЭВМ.

Защита памяти в однопрограммных системах заключается в установке защиты областей памяти, занятых операционной системой, от воздействия программ пользователя. Эта функция реализуется при помощи одного регистра границы, встроенного в центральный процессор.

Регистр границы содержит либо старший адрес команды, относящийся к операционной системе, либо младший адрес доступной программе основной памяти (адрес начала программы). Если программа пользователя пытается войти в область операционной системы, то вырабатывается прерывание по защите памяти, и программа аварийно завершается.

#### *4.2.3 Связное распределение памяти при мультипрограммной обработке*

При мультипрограммной обработке в памяти компьютера размещается сразу несколько заданий. Распределение памяти между заданиями в этом случае может быть выполнено следующими способами:

- распределение фиксированными разделами;
- распределение переменными разделами;
- распределение со свопингом.

Распределение фиксированными разделами имеет две модификации:

- а) с загрузкой программ в абсолютных адресах;
- б) с загрузкой перемещаемых модулей.

При загрузке перемещаемых модулей вся оперативная память машины разбивается на некоторое количество фиксирован-

ного размера. Размеры разделов могут не совпадать. В каждом разделе может быть размещено только одно задание.

В случае загрузки программ в абсолютных адресах при их подготовке указывается начальный адрес загрузки программ, совпадающий с начальным адресом раздела, в котором эта программа будет выполняться.

В случае загрузки перемещаемых модулей раздел, в котором будет размещено задание, либо автоматически определяется операционной системой в соответствии с реализованной в нем стратегией выбора раздела (“первый подходящий”, “самый подходящий”, “самый неподходящий”), либо указывается операционной системе специальными командами языка управления заданиями.

В обоих случаях задание монопольно владеет всем объемом оперативной памяти раздела, в который оно было помещено операционной системой.

Коэффициенты использования памяти при распределении с фиксированными разделами вычисляется по формуле:

$$h_{CM} = \frac{1}{V_0} \sum_{i=1}^{N_\Phi} V_{\Pi_i}; \quad (3)$$

где  $h_{CM}$  – коэффициент использования памяти раздела;

$V_{\Pi_i}$  – длина программы, помещенной в  $i$ -ый раздел;

$N_\Phi$  – количество разделов;

$V_0$  – общий объем оперативной памяти, доступной для распределения.

Основным недостатком распределения памяти фиксированными разделами является неэффективное использование ресурсов вычислительной системы из-за возможного появления длинных очередей заданий, ожидающих освобождения конкретного раздела в то время, как остальные разделы пусты. Подобная ситуация изображена на рисунке 19. Задания, ожидающие освобождения раздела С, могли бы разместиться и в разделах А или В, однако операционная система не позволяет им это сделать, т.к. в управляющей информации указан конкретный раздел, в котором эти задания должны выполняться – раздел С.

Способ распределения памяти фиксированными разделами используется в операционных системах ОС ЕС и IBM/360 в режиме MFT, в котором загрузка программ выполняется перемещаемыми модулями.

Неэффективное использование памяти  
при распределении фиксированными разделами

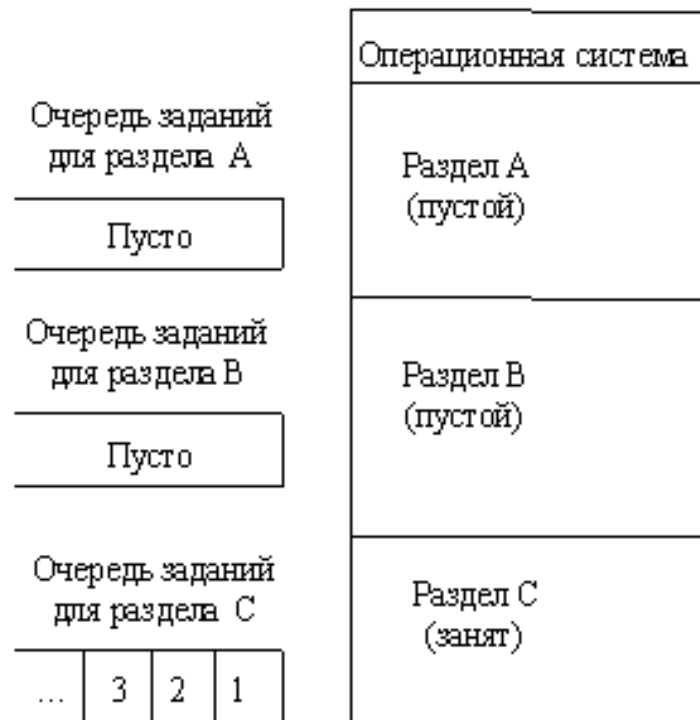


Рисунок 19 – Задания, ожидающие освобождения раздела С

Защита памяти при распределении фиксированными разделами выполняется аналогично защите памяти для одного пользователя, только теперь необходимо наличие нескольких *граничных* регистров – по два регистра на каждый раздел. В одном из граничных регистров указывается нижняя граница раздела, а во втором – его верхняя граница. Если программа пользователя пытается обратиться к данным, расположенным вне области адресов данного раздела, то вырабатывается прерывание по защите памяти.

В мультипрограммных системах с фиксированными разделами наблюдается явление фрагментации памяти.

Фрагментация памяти – появление в памяти вычислительной машины чередования занятых и незанятых (свободных) участков оперативной памяти.

При распределении фиксированными разделами появление фрагментации обусловлено тем, что либо задания пользователей



не полностью занимают выделенные им разделы, либо часть разделов остается незанятой.

На рисунке 20 показано проявление фрагментации оперативной памяти.

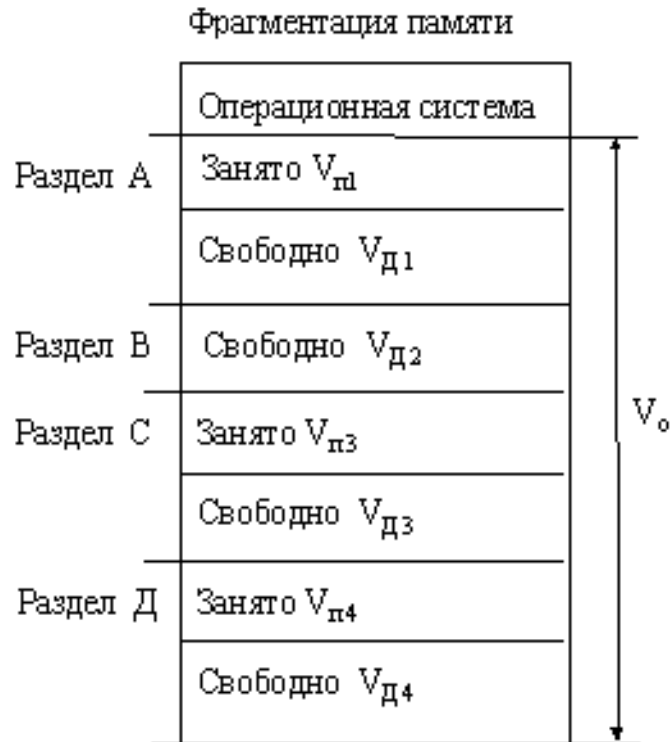


Рисунок 20 – Фрагментации оперативной памяти

Уровень фрагментации можно оценить коэффициентом фрагментации  $K_f$ , вычисляемый по формуле:

$$K_f = \frac{1}{V_0} \sum_{i=1}^{N_f} V_{дi}; \quad (4)$$

где  $V_{дi}$  – размер  $i$ -ой дыры, т.е.  $i$ -го участка свободной памяти, ограниченного программами пользователей;

$N_f$  – количество “дыр”, т.е. участков свободной памяти, лежащих между программами пользователей;

$V_0$  – объем оперативной памяти, доступной для распределения.

Фрагментация памяти представляет собой нарушение односвязности пространства свободной памяти ЭВМ, что приводит к снижению эффективности использования памяти как одного из основных ресурсов вычислительной машины.

Распределение памяти переменными разделами предназначено для повышения эффективности использования оперативной памяти ЭВМ. Суть способа распределения памяти переменными разделами состоит в том, что заданиям, когда они поступают, выделяется такой объем памяти, который им требуется, т.е. размер раздела оперативной памяти, выделяемой каждому заданию, в точности соответствует размеру этого задания. Поэтому “перерасхода” памяти, как это происходит при распределении фиксированными разделами, в данном способе не наблюдается.

Имеется две модификации способа распределения переменными разделами:

- распределение переменными неперемещаемыми разделами;

- распределение переменными перемещаемыми разделами.

При распределении памяти *переменными неперемещаемыми разделами* (динамическими разделами) операционная система создает две таблицы: таблицу учета распределенных областей памяти и таблицу учета свободных областей памяти (“дыр”).

При поступлении очередного задания память для него отводится на этапе долгосрочного планирования, причем выделение памяти осуществляется по информации из таблицы учета “дыр” в соответствии с принятой в ОС стратегией размещения (“первый подходящий”, “самый подходящий”. “самый неподходящий”). При успешном распределении ОС корректирует обе таблицы – распределенных и свободных областей.

После окончания какого-либо задания занимаемый им участок памяти освобождается, и операционная система корректирует таблицу распределенных областей, вычеркивая из нее информацию о закончившемся задании, а также заносит в таблицу свободных областей данные о вновь появившейся “дыре”.

При распределении памяти переменными перемещаемыми разделами операционная система осуществляет действия, называемые уплотнением памяти, состоящими в перемещении всех занятых участков к одному или другому краю основной памяти. Благодаря этому вместо большого количества небольших “дыр”, образующихся при использовании распределения переменными неперемещаемыми разделами, формируется единый (связный) участок свободной памяти.

Дефрагментация памяти, применяемая при распределении перемещаемыми разделами, имеет свои недостатки:

- требуются дополнительные затраты времени;
- во время уплотнения памяти система должна прекращать (приостанавливать) все другие работы, что зачастую может оказаться неприемлемым;
- необходимость перемещения заданий в памяти требует хранения значительного объема информации, связанной с размещением программ в памяти, что увеличивает требования к памяти со стороны ОС;
- при интенсивном потоке коротких программ может возникнуть необходимость частой дефрагментации памяти, так что затачиваемые на эти цели системные ресурсы могут оказаться неоправданными получаемой выгодой.

Распределение памяти со свопингом (от англ. swapping – подкачка) характеризуется тем, что в отличие от рассмотренных ранее способов распределения программы пользователей не остаются в основной памяти до момента их завершения. В простейшей системе со свопингом в каждый момент времени только одно задание пользователя находится в основной памяти и занимает ее до тех пор, пока оно может выполняться, а затем освобождает как память, так и центральный процессор для следующего задания.

Таким образом, вся память целиком на короткий период выделяется одному заданию, затем в некоторый момент времени это задание выводится (вытаскивается, т.е. осуществляется «откачка»), а очередное задание вводится (втаскивается, т.е. осуществляется «подкачка»). В обычном случае каждое задание, еще до своего завершения, будет много раз перекачиваться из внешней памяти в основную и обратно.

Для обеспечения свопинга во внешней памяти ОС создает один или несколько файлов подкачки, где хранятся образы оперативной памяти находящихся в работе заданий пользователей. Способ распределения памяти со свопингом применяется в простейших ОС, работающих в режиме разделения времени.

#### *Стратегии размещения информации в памяти*

Стратегии размещения информации в памяти предназначены для того, чтобы определить, в какое место основной памяти

следует помещать поступающие программы и данные при распределении памяти неперебрасываемыми разделами. Наиболее часто применяются следующие стратегии:

- размещение с выбором первого подходящего (стратегия «первый подходящий»);
- размещение с выбором наиболее подходящего (стратегия «самый подходящий»);
- алгоритм с выбором наименее подходящего (стратегия «самый неподходящий»).

Стратегия «первый подходящий» состоит в выполнении следующих шагов:

- упорядочить таблицу свободных областей в порядке возрастания адресов;
- поместить информацию в первый встретившийся участок основной памяти размером не менее требуемого.

Стратегия «самый подходящий» реализует следующую последовательность действий:

- упорядочить таблицу свободных областей в порядке возрастания размеров свободных областей;
- поместить информацию в первый встретившийся участок свободной памяти размером не менее требуемого.

Стратегия «самый неподходящий» выполняет следующие действия:

- упорядочить таблицу свободных областей в порядке убывания размеров областей;
- поместить информацию в первый встретившийся участок свободной памяти размером не менее требуемого.

Строгих доказательств преимущества той или иной стратегии перед остальными не существует, так что их применение в операционных системах основано на интуитивных аргументах разработчиков ОС.

#### *4.2.4 Организация виртуальной памяти. Основные концепции виртуальной памяти*

Термин виртуальная память обычно ассоциируется с возможностью адресовать пространство памяти, гораздо большее, чем емкость первичной (реальной, физической) памяти конкретной вычислительной машины. Концепция виртуальной памяти

впервые была реализована в машине, созданной в 1960 г. в Манчестерском университете (Англия). Однако широкое распространение системы виртуальной памяти получили лишь в ЭВМ четвертого и последующих поколений.

Существует два наиболее известных способа реализации виртуальной памяти – страничная и сегментная. Применяется также их комбинация – странично-сегментная организация виртуальной памяти.

Все системы виртуальной памяти характеризуются тем, что адреса, формируемые выполняемыми программами, не обязательно совпадают с адресами первичной памяти. Виртуальные адреса, как правило, представляют гораздо большее множество адресов, чем имеется в первичной памяти.

Суть концепции виртуальной памяти заключается в том, что адреса, к которым обращается выполняющийся процесс, отделяются от адресов, реально существующих в первичной памяти.

Адреса, на которые делает ссылки выполняющийся процесс, называются виртуальными адресами.

Адреса, которые реально существуют в первичной памяти, называются реальными (физическими) адресами.

Диапазон виртуальных адресов, к которым может обращаться выполняющийся процесс, называется пространством виртуальных адресов  $V$  этого процесса.

Диапазон реальных адресов, существующих в конкретной вычислительной машине, называется пространством реальных адресов  $R$  этой ЭВМ.

Несмотря на то, что процессы обращаются только к виртуальным адресам, в действительности они должны работать с реальной памятью. Для установления соответствия между виртуальными и реальными адресами разработаны механизмы динамического преобразования адресов ДПА (или ДАТ – от англ. Dynamics Address Transformation), обеспечивающие преобразование виртуальных адресов в реальные во время выполнения процесса. Все подобные системы обладают общим свойством – смежные адреса виртуального адресного пространства процесса не обязательно будут смежными в реальной памяти.

Это свойство называют «искусственной смежностью». Тем самым пользователь освобождается от необходимости рассматривать физическую память с ее уникальными характеристиками.

Виртуальная память строится, как правило, по двухуровневой схеме.

Первый уровень – это реальная память, в которой находятся выполняемые процессы и в которой должны размещаться данные, к которым обращаются эти процессы.

Второй уровень – это внешняя память большой емкости, например, накопители на магнитных дисках, способные хранить программы и данные, которые не могут все сразу уместиться в реальной памяти из-за ограниченности ее объема. Память второго уровня называют вторичной или внешней.

В мультипрограммных режимах реальная память разделяется между многими процессами. Поскольку каждый процесс может иметь гораздо большее пространство виртуальных адресов, чем реальная память, то в текущий момент времени в реальной памяти имеется возможность держать лишь небольшую часть программных кодов и данных каждого процесса, причем даже эти небольшие части кодов и данных не обязательно будут размещаться сплошным массивом реальной памяти (свойство «искусственной смежности»).

Механизм динамического преобразования адресов ведет учет того, какие ячейки виртуальной памяти в данный момент находятся в реальной памяти и где именно они размещаются. Это осуществляется с помощью таблиц отображения, ведущихся механизмом ДПА.

Информация, перемещаемая из виртуальной памяти в реальную, механизмом ДПА группируется в блоки, и система следит за тем, в каких местах реальной памяти размещаются различные блоки виртуальной памяти. Размер блока влияет на то, какую долю реальной памяти ДПА будет использовать непроизводительно, для своих целей.

Если блоки имеют одинаковый размер, то они называются страницами, а соответствующая организация виртуальной памяти называется страничной. Если блоки могут быть различных размеров, то они называются сегментами, а соответствующая организация виртуальной памяти называется сегментной. В некото-

рых системах оба подхода комбинируются, т.е. сегменты реализуются как объекты переменных размеров, формируемые из страниц фиксированного размера. Такая организация виртуальной памяти называется либо сегментно-страничной, либо странично-сегментной.

Адреса в системе поблочного отображения являются двухкомпонентными (двумерными). Чтобы обратиться к конкретному элементу данных, программа указывает блок, в котором расположен этот элемент, и смещение элемента относительно начала блока.

Преобразование адреса виртуальной памяти  $n = (b, d)$  в адрес реальной памяти  $r$  осуществляется следующим образом (см. рисунок 21). Каждый процесс имеет собственную таблицу отображения блоков, которую операционная система ведет в реальной памяти. Реальный адрес  $a$  этой таблицы загружается в специальный регистр центрального процессора, называемый регистром начального адреса таблицы отображения блоков процесса.

Таблицы отображения блоков содержат по одной строке для каждого блока процесса, причем эти блоки идут последовательно: сначала блок 0, затем блок 1 и т.д. Номер блока  $b$  суммируется с начальным адресом  $a$  таблицы, образуя реальный адрес строки таблицы для блока  $b$ . Найденная строка содержит реальный адрес  $b$  начала блока  $b$  в реальной памяти. К этому начальному адресу  $b$  прибавляется смещение  $d$ , так что образуется искомый реальный адрес  $r = b' + d$ .

## Преобразование виртуального адреса

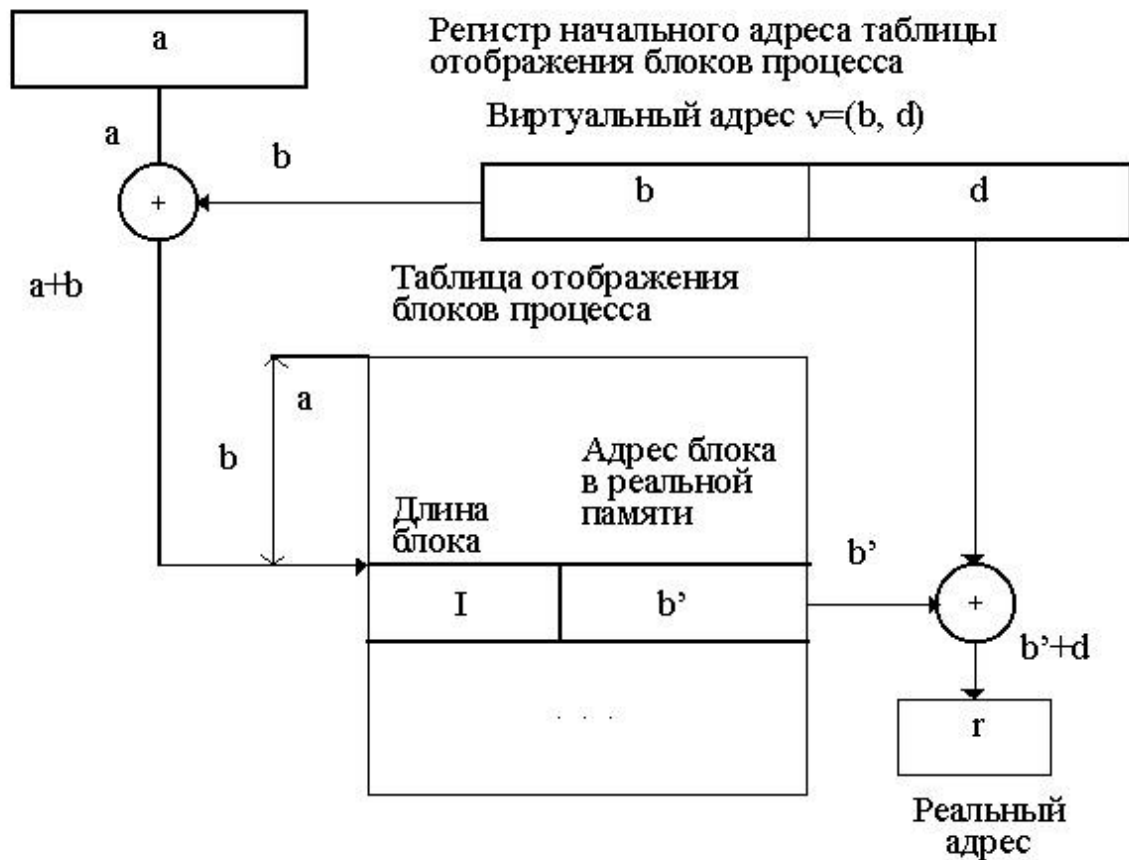


Рисунок 21 – Преобразование виртуального адреса

Все методы поблочного отображения, применяемые в системах с сегментной, страничной и комбинированной странично-сегментной организациями, подобны схеме отображения, показанной на рисунке 5, называемой схемой прямого отображения.

*Страничная организация виртуальной памяти.*

Виртуальный адрес при чисто страничной организации памяти – это упорядоченная пара  $(p, d)$ , где  $p$  – номер страницы в виртуальной памяти, а  $d$  – смещение в рамках страницы  $p$ . Процесс может выполняться, если его текущая страница находится в первичной памяти. Страницы переписываются из внешней памяти в первичную и размещаются в ней в блоках, называемых *страничными кадрами* и имеющих точно такой же размер, как у поступающих страниц. Страничные кадры начинаются в реальной памяти с адресов, кратных фиксированному размеру страниц.



*Поступающая страница может быть помещена в любой свободный страничный кадр.*

Для обеспечения работы механизма отображения страниц формируется таблица отображения страниц, каждая строка которой содержит информацию об отображаемой странице виртуальной памяти:

$r$  – признак наличия страницы в первичной памяти ( $r=0$  – страницы в первичной памяти нет; 1 – страница находится в первичной памяти):

$S$  – адрес страницы во внешней памяти (при  $r=0$ ):

$p'$  – номер страничного кадра в первичной памяти, где размещена виртуальная страница с номером  $p$ .

*Сегментная организация виртуальной памяти.*

Виртуальный адрес при сегментной организации виртуальной памяти – это упорядоченная пара  $n = (s, d)$ , где  $s$  – номер сегмента виртуальной памяти, а  $d$  – смещение в рамках этого сегмента. Процесс может выполняться только в том случае, если его текущий сегмент находится в первичной памяти. Сегменты передаются из внешней памяти в первичную целиком. Все ячейки, относящиеся к сегменту, занимают смежные адреса первичной памяти. Для размещения поступающих из внешней памяти сегментов в свободные участки первичной памяти применяются те же стратегии размещения, как и при распределении переменными неперемещаемыми разделами – «первый подходящий», «самый подходящий», «самый неподходящий». Динамическое преобразование виртуальных адресов в реальные адреса осуществляется в соответствии со схемой прямого отображения, приведенной на рисунке 21.

*Странично-сегментная организация виртуальной памяти.*

Системы со странично-сегментной организацией обладают достоинствами обоих способов реализации виртуальной памяти. Сегменты обычно содержат целое число страниц, причем не обязательно, чтобы все страницы сегмента находились в первичной памяти одновременно, а смежные страницы виртуальной памяти не обязательно должны оказаться смежными в первичной памяти. В системе со странично-сегментной организацией применяется трехкомпонентная (трехмерная) адресация. Виртуальный адрес  $n$  здесь определяется как упорядоченная тройка  $n = (s, p, d)$ , где  $s$  –

номер сегмента,  $p$  – номер страницы, а  $d$  – смещение в рамках страницы, где находится нужный элемент.

Операционная система для каждого процесса формирует, во-первых, одну таблицу сегментов процесса, и, во-вторых, таблицы страниц сегментов (по одной на каждый сегмент процесса).

Таблица сегментов процесса содержит в своих строках информацию о количестве страниц в сегменте и о начальных адресах  $s'$  размещения таблиц страниц сегментов в первичной памяти ЭВМ.

Каждая страница таблиц сегмента содержит в своих строках информацию о начальном адресе  $p'$  размещения в первичной памяти страничного кадра для данной страницы виртуальной памяти.

Динамическое преобразование виртуальных адресов в системах со странично-сегментной организацией отличается от преобразования по схеме наличием еще одного уровня вычисления адреса, как это показано на схеме рисунке 6, и появлением таблиц страниц для каждого сегмента процесса.

#### *4.2.5 Управление виртуальной памятью. Стратегии управления виртуальной памятью.*

Стратегии управления виртуальной памятью, так же как и стратегии управления физической памятью, разделяются на три категории: стратегии вталкивания, стратегии размещения и стратегии выталкивания.

Целью *стратегий вталкивания* является определить, в какой момент следует переписать страницу или сегмент из вторичной памяти в первичную.

Целью *стратегий размещения* является определить, в какое место первичной памяти помещать поступающую страницу или сегмент.

Целью *стратегий выталкивания* является решить, какую страницу или сегмент следует удалить из первичной памяти, чтобы освободить место для помещения поступающей страницы или сегмента, если первичная память полностью занята.

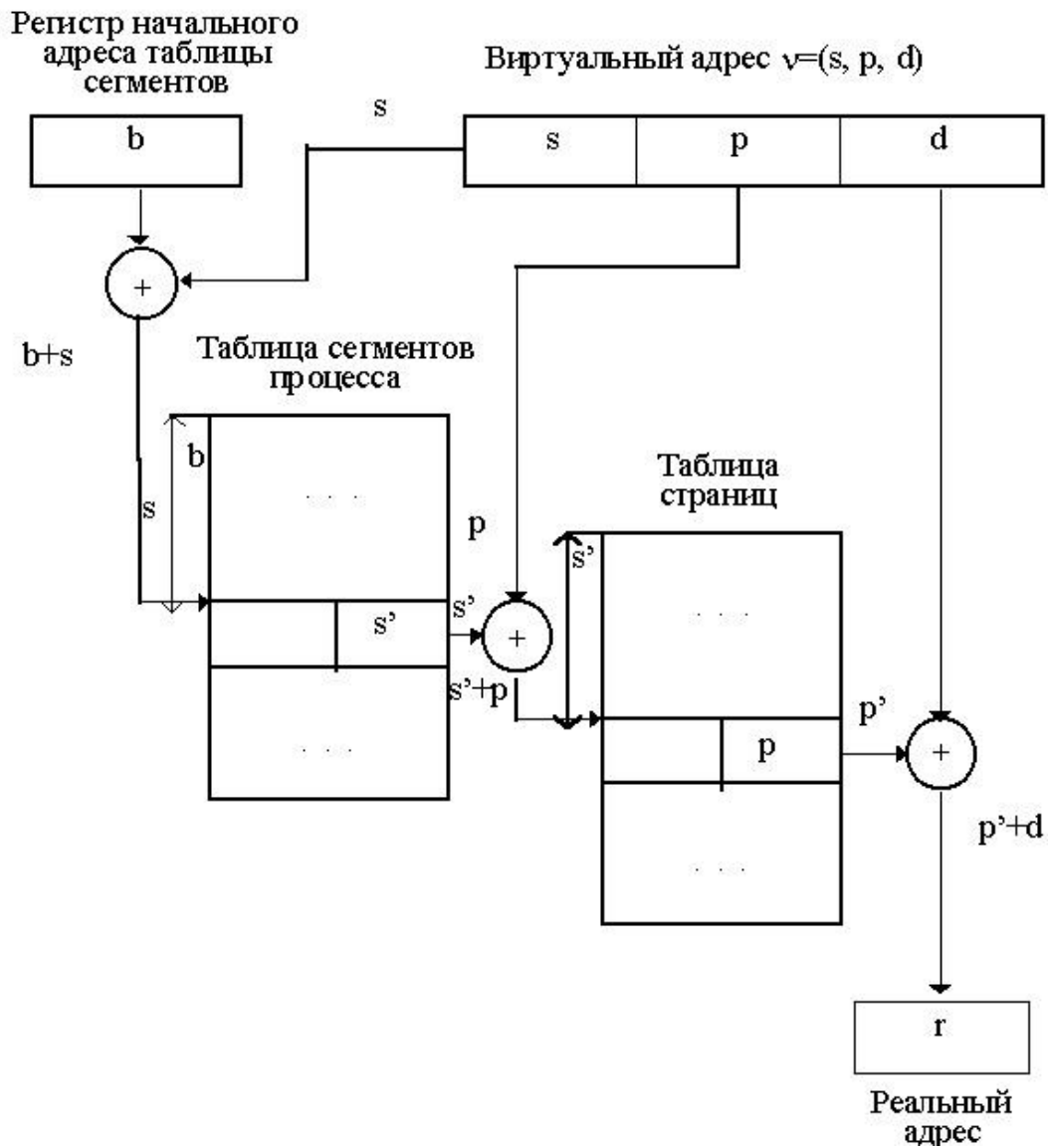


Рисунок 22 – Преобразование виртуального адреса при странично-сегментной организации

Большинство стратегий управления виртуальной памятью базируется на концепции *локальности*, суть которой заключается в том, что *распределение запросов процессов на обращение к памяти имеет, как правило, неравномерный характер с высокой степенью локальной концентрации*.

Свойство локальности проявляется как во времени, так и в пространстве.

*Локальность во времени* означает, что к ячейкам памяти, к которым недавно производилось обращение, с большой вероятностью будет обращение в ближайшем будущем.

*Локальность в пространстве* означает, что обращения к памяти, как правило, концентрируются так, что в случае обращения к некоторой ячейке памяти с большой вероятностью можно ожидать обращение к близлежащим ячейкам.

Свойство локальности наблюдается не только в прикладных программах, но и в работе программ операционной системы. Свойство это скорее эмпирическое (наблюдаемое на практике), чем теоретически обоснованное. Локальность никак нельзя гарантировать, однако ее вероятность достаточно велика. Самым важным следствием локализации является то, что программа может эффективно работать, если в первичной памяти находится подмножество, включающее наиболее “популярные” ее страницы или сегменты.

Для оценивания эффективности стратегий управления памятью в операционных системах применяют показатель “пространство-время”, вычисляемый по формуле:

$$S = VT; \quad (5)$$

где  $S$  – показатель “пространство-время”;  $V$  – объем первичной памяти, занимаемый процессом;  $T$  – длительность ожидания процессом подкачки необходимой страницы или сегмента.

*Уменьшение значения показателя  $S$  за счет снижения периодов ожидания процессом нужных ему страниц или сегментов является важнейшей целью всех стратегий управления памятью.*

*Стратегии вталкивания (подкачки).*

Для управления вталкиванием применяются следующие стратегии:

- вталкивание (подкачка) по запросу (по требованию);
- вталкивание (подкачка) с упреждением (опережением).

*Вталкивание (подкачка) по запросу* предполагает, что система ждет ссылки на страницу или сегмент от выполняющегося процесса и только после появления такой ссылки начинает переписывать данную страницу или сегмент в первичную память. Подкачка по запросу имеет положительные и отрицательные стороны.

К положительным сторонам относятся:

- гарантировано, что в первичную память будут переписываться только те страницы (сегменты), которые необходимы для работы процесса;

- накладные расходы на то, чтобы определить, какие страницы или сегменты следует передавать в первичную память, минимальны.

К недостаткам подкачки по запросу относится тот факт, что процесс в этом случае накапливает в первичной памяти требуемые ему страницы (сегменты) по одной. При появлении ссылки на каждую новую страницу (сегмент) процессу приходится ждать, когда эта страница (или сегмент) будет передана в первичную память. В зависимости от того, сколько страниц (сегментов) данного процесса уже находится в первичной памяти, эти периоды ожидания будут, как это следует из формулы, обходиться все более дорого, поскольку ожидающие процессы будут занимать все больший объем памяти.

*Вталкивание (подкачка) с упреждением* предполагает, что система пытается заблаговременно определить, к каким страницам или сегментам будет обращаться процесс. Если вероятность обращения высока и в первичной памяти имеется свободное место, то соответствующие страницы или сегменты будут переписываться в первичную память еще до того, как к ним будет явно производиться обращение. При правильном выборе страниц (сегментов) для упреждающей подкачки удастся существенно сократить общее время выполнения данного процесса и уменьшить значение показателя “пространство-время”.

К недостаткам стратегии подкачки с упреждением можно отнести тот факт, что, согласно теории вычислимости, точно предсказать путь, по которому будет развиваться процесс, в общем случае невозможно. Поэтому вполне возможны ситуации, когда решения о выборе страниц (сегментов) для упреждающей подкачки будет в большинстве случаев приниматься неверно для одного или нескольких процессов, развивающихся в системе, что в свою очередь приведет к резкому снижению скорости работы этих процессов из-за увеличения времени ожидания необходимых им страниц или сегментов.

*Стратегии размещения.*

В системах со *страничной* организацией виртуальной памяти решение о размещении вновь загружаемых страниц принимается достаточно просто: *новая страница может быть помещена в любой свободный страничный кадр.*

Для систем с *сегментной* организацией виртуальной памяти применяются такие же стратегии размещения, какие используются в системах распределения памяти переменными разделами, а именно:

- размещение с выбором первого подходящего свободного участка;
- размещение с выбором самого подходящего свободного участка;
- размещение с выбором наименее подходящего свободного участка.

*Стратегии выталкивания.*

В мультипрограммных системах вся первичная память бывает, как правило, занята. В этом случае программа управления памятью должна решать, какую страницу или какой сегмент следует удалить из первичной памяти, чтобы освободить место для поступающей страницы или сегмента. В настоящее время применяются следующие стратегии выталкивания (откачки) страниц (сегментов):

- выталкивание случайных страниц или сегментов;
- выталкивание первой пришедшей страницы или сегмента (FIFO);
- выталкивание дольше всего не использовавшихся страниц или сегментов (LRU);
- выталкивание наименее часто использовавшихся страниц или сегментов (LFU);
- выталкивание не использовавшихся в последнее время страниц или сегментов (NUR).

*Стратегия выталкивания случайных страниц или сегментов* является наиболее простой в реализации, обладает малыми издержками и не является дискриминационной по отношению к каким-либо процессам, работающим в системе. В соответствии с этой стратегией любые страницы или сегменты, находящиеся в первичной памяти, могут быть выбраны для выталкивания с равной вероятностью, в том числе даже следующая страница или

сегмент, к которому будет производиться обращение (и которые, естественно, удалять из памяти наиболее нецелесообразно). Поскольку подобная стратегия, по сути, рассчитана на “слепое” ведение, в реальных системах она применяется редко.

*Стратегия выталкивания первой пришедшей страницы или сегмента (FIFO-стратегия)* реализует принцип “первый пришел – первый ушел”. В этом случае в момент поступления каждой страницы (сегмента) в первичную память ей (ему) присваивается метка времени. Когда появляется необходимость удалить из первичной памяти какую-либо страницу (сегмент), выбирается та страница (сегмент), у которой метка времени имеет наименьшее значение. Аргументом в пользу такой стратегии выталкивания является довод, что у данной страницы уже были возможности “использовать свой шанс”, и пора дать подобные возможности другой странице. Однако стратегия FIFO с большой вероятностью будет приводить к удалению из первичной памяти активно используемых страниц (сегментов), поскольку тот факт, что страница (сегмент) находится в первичной памяти в течение длительного времени, вполне может означать, что эта страница или сегмент постоянно находится в работе.

*Стратегия выталкивания дольше всего не использовавшихся страниц или сегментов (LRU-стратегия)* предусматривает, что для выталкивания следует выбирать те страницы (сегменты), которые не использовались дольше других. Стратегия LRU требует, чтобы при каждом обращении к страницам (сегментам) их метки времени обновлялись. Это может быть сопряжено с существенными издержками, поэтому LRU-стратегия, несмотря на свою привлекательность, в современных операционных системах реализуется достаточно редко. Кроме того, при реализации LRU-стратегии может быть так, что страница (сегмент), к которой дольше всего не было обращений, в действительности станет следующей используемой страницей (сегментом), если программа к этому моменту очередной раз пройдет большой цикл, охватывающий несколько страниц или сегментов.

*Стратегия выталкивания реже всего используемых страниц или сегментов (LFU-стратегия)* является одной из наиболее близких к рассмотренной выше LRU-стратегии. В соответствии с LFU-стратегией из первичной памяти выталкиваются наименее

часто (наименее интенсивно) использовавшиеся к данному времени страницы или сегменты. Здесь контролируется интенсивность использования страниц (сегментов). Для этого каждой странице (сегменту) назначается счетчик, значение которого увеличивается на единицу при каждом обращении к данной странице (сегменту). LFU-стратегия, будучи интуитивно оправданной, имеет те же недостатки, что и стратегия LRU: во-первых, велика вероятность того, что из первичной памяти будут удалены страницы или сегменты, которые потребуются процессам при следующем обращении к памяти и, во-вторых, ее реализация может быть сопряжена со значительными затратами на организацию контроля интенсивности использования страниц или сегментов.

*Стратегия выталкивания не использовавшихся в последнее время страниц или сегментов (NUR-стратегия)* также является близкой к стратегии LRU и характеризуется относительно небольшими издержками на свою реализацию. Согласно NUR-стратегии из первичной памяти выталкиваются те страницы (сегменты), к которым не было обращений в последнее время. В соответствии со свойством локальности во времени к страницам (сегментам), не использовавшимся в последнее время, вряд ли будет обращение в ближайшем будущем, так что их можно заменить на вновь поступающие страницы.

Поскольку желательно заменять те страницы (сегменты), которые в период нахождения в основной памяти не изменялись, реализация NUR-стратегии предусматривает введение двух аппаратных бит-признаков на страницу (сегмент):

- бит-признак  $b_0$  обращения к странице (сегменту);
- бит-признак  $b_1$  модификации страницы (сегмента).

Первоначально все  $b_0$  и  $b_1$  устанавливаются в 0. При обращении к странице (сегменту) соответствующий бит-признак  $b_0$  устанавливается в 1. В случае изменения содержимого страницы (сегмента) соответствующий бит-признак  $b_1$  устанавливается в 1. NUR-стратегия предусматривает существование четырех групп страниц (сегментов).

В первую очередь из первичной памяти выталкиваются страницы (сегменты), принадлежащие группам с меньшими номерами.



Учет времени, в течение которого к страницам (сегментам) не было обращений, осуществляется периодическим сбрасыванием в 0 всех битов-признаков, выполняемым операционной системой.

Практически любая стратегия выталкивания страниц (сегментов) не исключает опасности нерациональных решений. Это объясняется тем, операционная система не может точно прогнозировать будущее поведение любого из процессов, поступивших к ней на обработку.

### 4.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на вопросы.

### 4.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Часто единственным достоинством виртуальной памяти называют возможность обеспечить для процесса объем виртуального адресного пространства, превышающий объем реальной памяти. Назовите другие достоинства виртуальной памяти.

2. В чем достоинства и недостатки преобразования виртуальных адресов в реальные во время выполнения программы? Какая часть работы по этому преобразованию выполняется аппаратным обеспечением, а какая – ОС?

3. Иногда считают, что виртуальная память может быть обеспечена только в системах с аппаратной поддержкой динамической трансляции адреса. Докажите, что это не так.

4. Почему при поиске свободной памяти стратегия "самый подходящий" оказывается хуже, чем "первый подходящий".

5. Сравните сегментную и страничную модели виртуальной памяти. Какая из них представляется Вам лучшей и почему?

6. Дополните приведенные в разделе 3.5. соображения по поводу выбора размера страницы.

7. Смоделируйте ситуацию применения дисциплины вытеснения FCFS, в которой увеличение числа реальных страниц приведет к увеличению числа страничных отказов.

8. Что такое кластерная подкачка страниц? Почему в современных ОС она становится все более популярной?

9. Каким образом ОС может определять, к каким страницам будут обращения в ближайшее время?

10. Большой размер виртуальной памяти процесса может приводить к тому, что даже таблица страниц не будет помещаться в реальной памяти. Какими путями решается эта проблема в современных ОС?

11. Каким образом снижение стоимости памяти влияет на дисциплины управления памятью?

12. Какие принципиальные изменения в концепции памяти может повлечь за собой увеличение разрядности адреса?

## **5. РАБОТА С ПРОГРАММОЙ «ФАЙЛ-МЕНЕДЖЕР ПРОВОДНИК». РАБОТА С ФАЙЛОВЫМИ СИСТЕМАМИ И ДИСКАМИ**

### **5.1 ЦЕЛЬ РАБОТЫ**

Цель работы – изучение основных приемов работы в программе Проводник ОС Windows; приобретение практических навыков в работе файловыми менеджерами Total Commander и FAR Manager.

### **5.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

#### *5.2.1 Введение*

Вся информация, будь то программы или тексты, музыка или картинки, хранится в компьютере в виде файлов. Часто возникает необходимость что-либо с этими файлами сделать: скопировать, удалить, переместить в другую папку, модифицировать и т.п. Все эти функции обязательно присутствуют в любой ОС. Для облегчения работ, связанных с файловыми операциями, используются файловые менеджеры, которые предоставляют удобный интерфейс.

В Windows используется встроенный файл-менеджер Проводник (*Explorer*), но во многих случаях пользоваться им не очень удобно. Скажем, двухпанельное меню намного удобнее при копировании файлов, чем однопанельное. Встроенные функции сортировки файлов, сравнения, архивирования и т.п. – тоже не помешают. Именно поэтому с момента появления на свет Windows независимые разработчики продолжали упорно создавать альтернативы Проводнику.

При всем многообразии современных файл-менеджеров их можно разделить на две большие группы. В первую входят подоби́я (или клоны) Проводника с добавлением некоторых полезных функций. А вторая группа представлена программами, имитирующими интерфейс самого популярного файлового менеджера прошлых лет – Norton Commander.

### 5.2.2 Работа с файловым менеджером Проводник

Интерфейс программы Проводник.

Программа Проводник – служебная программа, относящаяся к категории *файловых менеджеров* и предназначенная для навигации по файловой структуре компьютера и ее обслуживания.

Проводник глубоко интегрирован в ОС. Для запуска Проводника следует щелкнуть правой кнопкой мыши по кнопке Пуск и выбрать в контекстном меню команду Проводник.

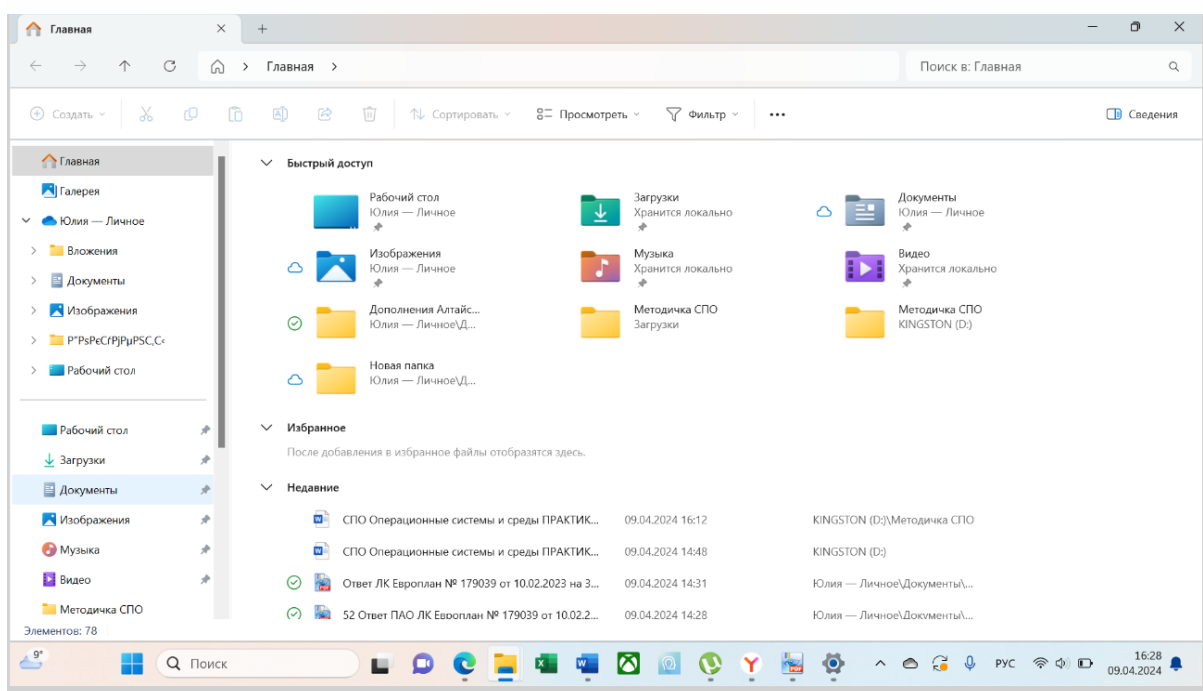


Рисунок 23 – Окно программы Проводник

Интерфейс Проводника представлен на рисунке 23. Окно Проводника имеет две рабочих области: левую панель, называемую *панелью папок*, и правую панель, называемую *панелью содержимого*.

Навигация по файловой структуре – доступ к нужной папке и ее содержимому. Навигацию по файловой структуре выполняют на левой панели Проводника, на которой отображается структура папок. Папки могут быть *развернуты* или *свернуты*, а также раскрыты или закрыты. Если папка имеет вложенные папки, то на левой панели рядом с папкой отображается узел, отмеченный знаком «>». Щелчок мышью на узле разворачивает папку. Таким же образом папки и сворачиваются.

Для того чтобы раскрыть папку, надо щелкнуть на ее значке. Содержимое раскрытой папки отображается на правой панели.

Программа Проводник управляет файлами и обеспечивает один из способов просмотра файловой системы. Она позволяет открывать папки, копировать, перемещать, удалять, переименовывать и упорядочивать папки и файлы, запускать программы на выполнение, открывать документы и многое другое.



Рисунок 24 – Инструментальная панель программы Проводник

Для поиска папок и файлов, а также любой другой информации, которая может находиться как на локальном компьютере, так и в локальной сети или в Интернете используется кнопка Поиск, нажатие которой раскрывает диалоговое окно Результаты поиска (рисунок 2).

В поисковом предписании на поиск файлов и папок задаются:

- имя файла или папки (можно указывать часть имени файла или папки, а также использовать метасимволы «\*» и «?»);
- фрагмент текста (для текстовых файлов);
- месторасположение искомого объекта (диск);
- тип файла;
- дата создания объекта;
- размер файла.

Найденные объекты отобразятся в поле результатов поиска. Их открытие осуществляется традиционно – двойным щелчком мышью.

#### *Приемы работы с мышью*

Основным средством выполнения операций с объектами в окне программы Проводник является мышь, для которой существует ряд приемов работы:

- наведение указателя – указатель мыши навести на объект или элемент управления на экране. Если задержать указатель мыши несколько секунд, то часто рядом с указателем открывает-

ся всплывающая подсказка, кратко описывающая назначение указываемого элемента;

- щелчок – навести указатель мыши на объект и щелкнуть (нажать и быстро отпустить) левой кнопкой мыши. Щелчком приводятся в действие элементы управления. Если щелчок выполнен на объекте, то объект выделяется (готовится к использованию);

- двойной щелчок – это два последовательных быстрых щелчка левой кнопкой. Щелкать нужно достаточно быстро, во время двойного щелчка мышь должна быть неподвижна. Этим приемом выполняются операции с объектами: программы запускаются, файлы данных открываются, звуковые или видеофайлы воспроизводятся и т.д.;

- перетаскивание – навести указатель мыши на объект, нажать левую клавишу мыши, переместить указатель в нужное место и только после этого отпустить левую кнопку мыши. Перетаскиванием выполняется перемещение объектов: значков, окон и т.п.;

- протягивание – как и перетаскивание, выполняется при нажатой левой кнопке мыши, но при этом объект не перемещается, а изменяется по форме (обычно этим приемом изменяется размер окна);

- правый щелчок – выполняется при наведении указателя на какой-либо объект окна и щелчка правой кнопкой мыши. При этом открывается контекстное меню – элемент управления, содержащий пункты команд, которые можно выполнить с объектом;

- специальное перетаскивание – прием, эквивалентный перетаскиванию, но выполняемый при нажатой правой кнопке мыши. Кроме перемещения в новое место, в момент освобождения правой кнопки открывается меню, из которого можно выбрать определенное действие, например, копирование или перемещение объекта, создание ярлыка и т.п.

Для выбора нескольких объектов надо поставить указатель мыши в левый верхний угол воображаемого прямоугольника с выделяемыми объектами, нажать левую кнопку и тянуть мышь в направлении правого нижнего угла, после чего отпустить кнопку.

Для выбора нескольких объектов, расположенных подряд, надо щелкнуть мышью по первому объекту, нажать клавишу [SHIFT], щелкнуть по последнему объекту и отпустить клавишу [SHIFT].

Для выбора нескольких несмежных объектов надо щелкать по ним мышью при нажатой клавише [CTRL].

### *Настройка Проводника*

Существует возможность тонкой подстройки этой программы под свои запросы и вкус. Для этого вам стоит выбрать пункт *Параметры/Параметры папок* на панели текстового меню Проводника.

На вкладке Общие (рисунок 25) можно изменить режим отображения файлов и папок не только в Проводнике, но и на Рабочем столе Windows.

Вкладка Вид (рисунок 26) предоставляет возможность «тонкой настройки» внешнего вида Проводника. Например, вы можете включить режим отображения скрытых и системных файлов, а также заставить Проводник показывать расширения файлов в дополнение к специальным иконкам (для этого снимите галочки с пунктов *Скрывать защищенные системные файлы* и *Скрывать расширения для зарегистрированных типов файлов*). Для того чтобы получить полный контроль над содержимым вашего компьютера, пометьте еще и пункт *Показывать скрытые файлы и папки*.

Включенный режим отображения расширения может помочь нам в дальнейшем – с его помощью мы, в частности, сможем отловить некоторые виды вирусов, распространяющихся по электронной почте.

Вкладка Типы файлов представит вам полный перечень всех зарегистрированных в Windows типов файлов с информацией о том, какая именно программа используется для просмотра и редактирования каждого.

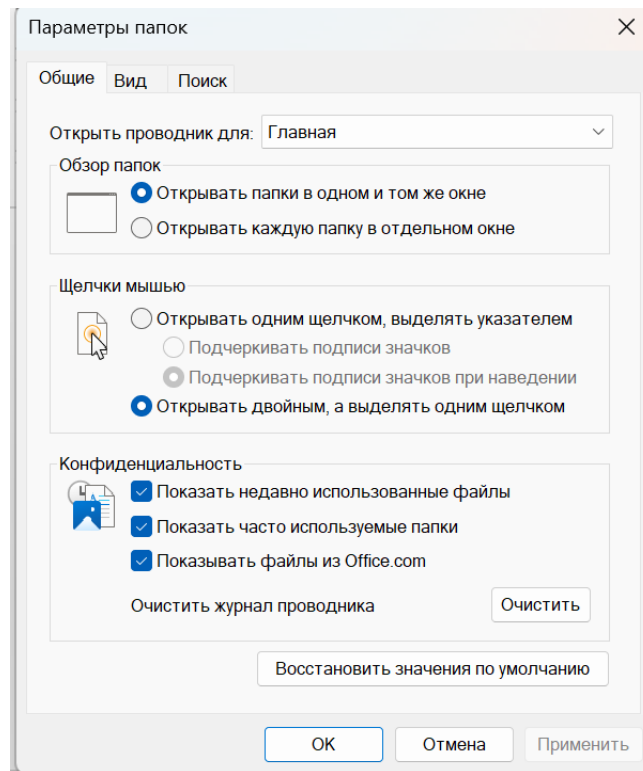


Рисунок 25 – Вкладка Общие диалогового окна Параметры папок

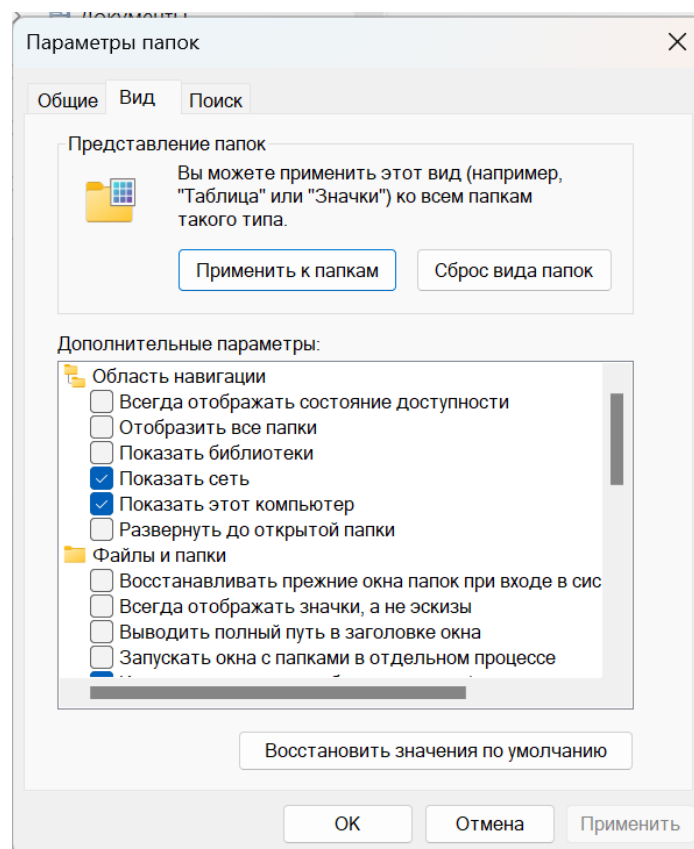


Рисунок 26 – Вкладка Вид диалогового окна Параметры папок



### 5.2.3 Программа *Total Commander*

Программа запускается из главного меню или двойным щелчком мыши по ярлыку программы на рабочем столе. После запуска открывается окно программы.

Интерфейс программы *Total Commander* (рисунок 27) составляют:

- заголовок окна;
- строка меню;
- инструментальная панель;
- панель кнопок дисков;
- две информационных панели, отображающие структуру дисков и папок;
- строка состояния;
- командная строка;
- строка функциональных клавиш.

Включение тех или иных элементов интерфейса осуществляется на вкладке Вид диалогового окна Настройка, которое открывается командами меню Конфигурация Р Настройка.

*Total Commander* располагает контекстными меню для некоторых элементов интерфейса. Все эти меню могут быть открыты щелчком правой кнопки мыши:

В файловых панелях вы можете вызвать контекстное меню, нажав SHIFT+F10. Если правая кнопка мыши используется для выделения файлов, вы можете с её помощью вызвать и контекстное меню, удерживая кнопку нажатой немного дольше (примерно 1 секунду).

С помощью команд Копировать и Вырезать можно копировать/перемещать выделенные файлы, используя буфер обмена. После применения одной из этих команд вам нужно будет просто выбрать команду Вставить в контекстном меню пути назначения.

Панель инструментов также содержит контекстное меню для редактирования или удаления отдельных кнопок.

В *Windows* вы можете вызвать контекстное меню любого открытого в панели каталога или любого доступного диска щелчком правой кнопки мыши на заголовке панели или, соответственно, на кнопке диска. При щелчке на списке дисков вызывается

контекстное меню для текущего диска. Кнопка F8 имеет контекстное меню системной Корзины.

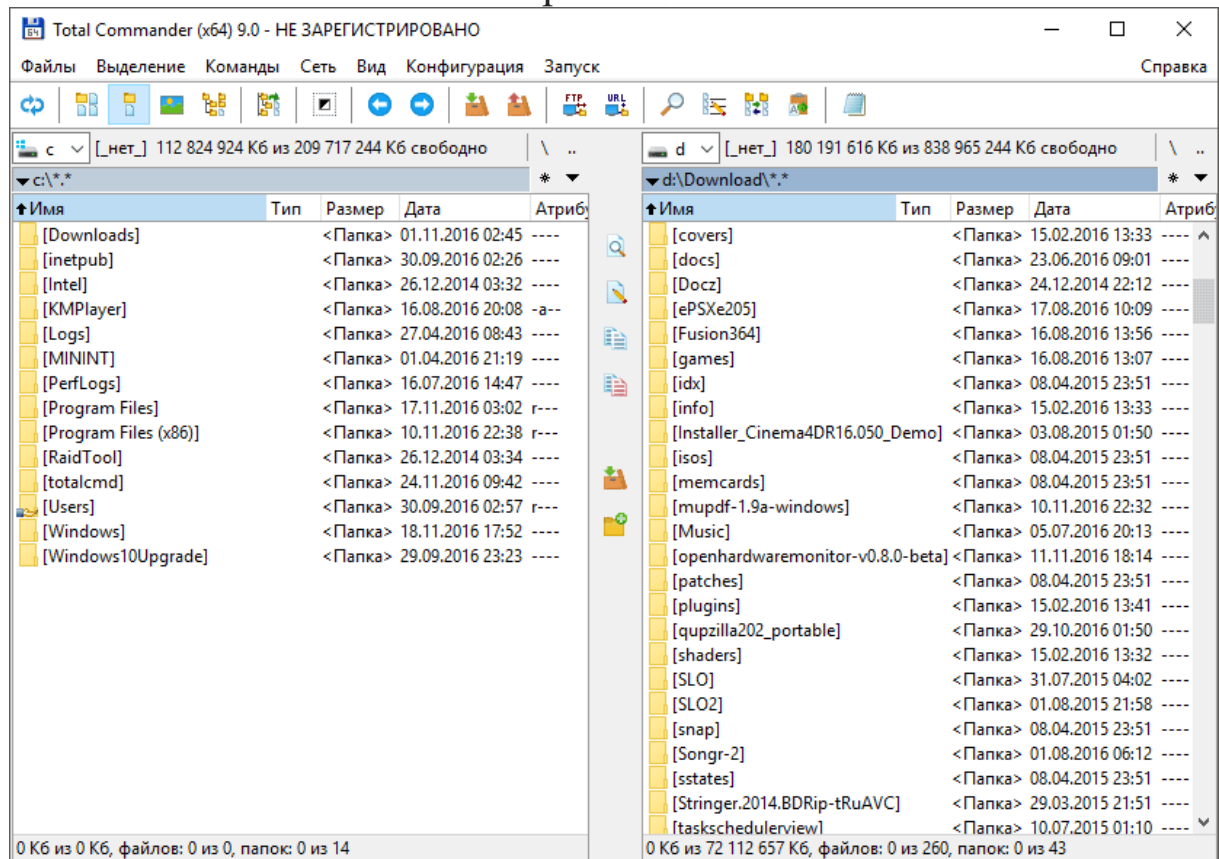


Рисунок 27 – Окно файлового менеджера Total Commander

### *Операции с файлами.*

#### *Выделение файлов.*

Чтобы выделить файлы или каталоги, просто щёлкните по ним мышью или переместитесь на них с помощью клавиш курсора и нажмите клавишу INSERT. Если в диалоге конфигурации вы выбрали для выделения объектов левую кнопку мыши, правая кнопка служит только для открытия контекстного меню; если же для выделения выбрана правая кнопка мыши, можно выделять объекты и левой кнопкой. Если вы выбираете каталог, используя клавишу ПРОБЕЛ, показывается размер дискового пространства, занятого этим каталогом.

Для выделения *нескольких последовательных объектов* щёлкните по первому файлу или каталогу, предназначенному для выделения. После этого нажмите клавишу SHIFT и, удерживая её, щёлкните левой кнопкой мыши на последнем объекте, который хотите выделить.

Для выделения *нескольких несмежных объектов* щелкайте левой кнопкой мыши любые несмежные файлы или каталоги, держа при этом нажатой клавишу CTRL (снятие выделения с отдельного файла/каталога выполняется точно так же).

Для выделения (или отмены выделения) определённых типов файлов нажмите клавишу Num + (или Num –) или выберите одну из команд выделения (Выделить группу / Снять выделение группы) в меню Выделение. Затем в появившемся диалоге введите нужный вам тип файла (например, \*.txt). Вы можете также указать несколько типов файлов, и даже те типы файлов, которые не должны быть выделены. Их следует отделить символом вертикальной черты "|".

Например, w\*.\*|\*.bak \*.old – выделить все файлы, которые начинаются с w и не заканчиваются .bak или .old; |\*.exe – выделить все файлы, кроме программ.

Выделить всё содержимое файловой панели можно при помощи комбинации клавиш CTRL+A.

Если необходимо выделить совокупность файлов, имеющих одинаковое расширение, то следует выбрать файл с нужным вам расширением и нажать сочетание клавиш ALT+Num + (или ALT+Num –, чтобы снять выделение).

Команда Инвертировать выделение отмечает все файлы в исходном каталоге, которые не были отмечены, и снимает выделение у ранее отмеченных файлов. Для вызова команды нажмите клавишу Num \*.

Выберите файлы, которые хотите просмотреть, и нажмите F3. Встроенная программа просмотра файлов (она называется *Lister*) показывает выделенные файлы или файл под курсором (в зависимости от настроек). Удерживая клавишу SHIFT при нажатии F3, вы выбираете альтернативный способ (т.е. просмотр выделенных файлов, если по умолчанию задан просмотр файла под курсором, и наоборот). При просмотре нескольких файлов *Lister* даёт вам возможность последовательно переключаться между файлами нажатием клавиш N (вперёд) и P (назад), ESC закрывает *Lister* и возвращает вас в Total Commander.

Если исходный каталог отображает содержание архива, вас спросят, действительно ли вы хотите распаковать и просмотреть отмеченные файлы. Файлы распаковываются в каталог, опреде-

лённый системной переменной `temp`. Вы можете установить в вашем `autoexec.bat`, например, `set temp=c:\windows\temp`. Если эта переменная не задана, используется каталог программы Total Commander. Создаётся подкаталог `\_tc`, где сохраняются все временные файлы. При закрытии Total Commander этот каталог удаляется (если он пустой).

### *Правка*

Поместите курсор на файл, который вы хотите редактировать, и нажмите F4. При этом запускается выбранный вами в диалоге настройки редактор, а в него загружается выбранный файл. По умолчанию запускается стандартный Блокнот Windows. Он может работать только с текстовыми файлами ограниченного размера. Если вы хотите редактировать файлы других типов, просто дважды щёлкните на файле или нажмите ENTER. Запустится программа, ассоциированная с файлом. Вы можете ассоциировать расширения файлов, содержащих данные, с программами, используя Файлы F Открыть с помощью...

Если исходный каталог показывает содержимое архива, файл под курсором распаковывается во временный каталог и загружается в предварительно выбранный редактор. Если вы измените файл и затем закроете редактор, Total Commander запросит, хотите ли вы заново упаковать файл в архив. Если вы выберете да, архив будет обновлён, а временный каталог очищен. Если вы выберете нет, временный файл будет удалён, архив же останется неизменённым.

### *Копирование*

Выделите файлы (каталоги), которые вы хотите скопировать, и нажмите F5. При этом откроется диалоговое окно, в котором вы можете ввести каталог назначения и маску файлов. По умолчанию в этом поле указан каталог второй панели с маской файлов `*.*`. В качестве конечного имени файла вы можете использовать любое корректное DOS-совместимое имя, включая символы подстановки (`*` и `?`). В поле ввода, находящемся ниже, вы можете определить, какие именно файлы будут скопированы.

Чтобы скопировать файл в тот же самый каталог (под другим именем), нажмите SHIFT+F5.

Кнопкой Дерево вы можете выбрать каталог назначения из дерева каталогов. Если вы хотите выбрать каталог на другом дис-

ке, вы можете указать этот диск (включая двоеточие ':') в диалоговом окне перед тем, как нажать кнопку Дерево.

Вы также можете добавлять один файл к другому. Удостоверьтесь, что подтверждение перезаписи не отключено, затем просто скопируйте файл, который вы хотите добавить, в тот файл, к которому вы хотите его добавить (для этого введите имя последнего в диалоге копирования в поле для ввода каталога назначения). Total Commander выведет диалог подтверждения перезаписи, в котором вы можете нажать кнопку "Дописать".

Кнопка Опции позволяет установить параметры для автоматического копирования. По умолчанию Total Commander выводит запрос о перезаписи файлов. Эта кнопка позволяет установить по умолчанию "Заменить все", "Пропустить все" или "Заменить все старые". Она также позволяет игнорировать атрибуты только для чтения, скрытый и системный при перезаписи или перемещении файлов.

В поле "Только файлы типа", вы можете указать, какие файлы копировать, причём это распространяется также на файлы из подкаталогов.

Примеры:

\*.txt \*.doc будут копироваться только файлы .doc и .txt.

\*.\* | \*.bak \*.old будет копироваться всё, кроме файлов .bak и .old.

\*.\* | папка1\ папка2\ не будут копироваться файлы из указанных каталогов.

Если исходный каталог показывает содержимое архива, выводится диалог распаковки файлов. Вы теперь можете также распаковать файлы непосредственно из архива на FTP-сервер. В этом случае файлы будут сначала распакованы во временный каталог, а затем переданы на удаленный сервер. Обратное направление (с FTP в архив) не поддерживается.

Если вы хотите создать новый архив и упаковать в него выделенные файлы, просто нажмите ALT+F5. Откроется диалоговое окно упаковки файлов. При использовании сочетания ALT+SHIFT+F5 файлы после упаковки будут удалены.

Если вы хотите распаковать архив под курсором (или выделенные архивы), нажмите ALT+F9. После указания каталога

назначения (и при необходимости – маски файлов), все файлы из архива будут распакованы.

В 32-битной версии операции копирования, распаковки (только ZIP) и упаковки могут быть переведены в фоновый режим нажатием кнопки 'В фоне' во время самой операции. Это позволяет во время этой операции выполнять в Total Commander другие задачи. После того, как фоновая операция завершится, нужно нажать F2 или CTRL+R, чтобы обновить каталог. Иначе изменённые файлы не будут показаны.

Эта команда позволяет выполнять переименование файлов и целых каталогов в исходном каталоге, она же может использоваться для перемещения их в другие каталоги или даже на другие диски. Вы можете также переместить (упаковать и затем удалить) файлы в архив. Перемещать файлы из архивов нельзя, необходимо последовательно использовать функции распаковки и удаления.

Выберите файлы и/или каталоги, которые вы хотите переименовать или переместить, затем нажмите F6. Если вы хотите только переименовать файл, оставив его на месте, нажмите вместо этого SHIFT+F6. При этом имя файла, подлежащего переименованию, будет открыто в небольшом окне редактирования прямо в файловой панели. Повторное нажатие F6 или SHIFT+F6 служит для циклического переключения между способами выделения (имя без расширения/имя+расширение). Закончив правку, вы должны нажать ENTER для подтверждения переименования файла. Операция отменяется щелчком за пределами окна редактирования или клавишей ESC. При переименовании ZIP-файла в EXE-файл Total Commander спросит, хотите ли вы создать самораспаковывающийся ZIP-архив. Если вы выберете "Да", архив будет преобразован в EXE-файл.

По F6 открывается диалоговое окно, где вы можете ввести каталог назначения и маску файла. По умолчанию предлагается путь к каталогу, открытому в другой панели. Если вы не указываете каталог назначения, таковым считается исходный каталог, т.е. файлы будут просто переименованы. Если вы не указываете маску файла, имя файла не изменяется.

Если выделено несколько файлов, комбинацией SHIFT+F6 открывается диалоговое окно, которое позволяет указать симво-

лы подстановки (\* и ?) для переименования нескольких файлов. Вы можете выбрать опцию Переименовать каждый файл отдельно, тогда в ходе операции вас запросят о новом имени для каждого файла по отдельности.

Если исходный каталог отображает содержимое архива, эта функция недоступна. Исключение: при помощи SHIFT+F6 вы можете переименовать одиночный файл/каталог внутри ZIP-архивов.

Чтобы переименовать большое число файлов согласно определённой схеме, вы можете использовать отдельный Инструмент группового переименования (Ctrl+M).

#### *Создание каталога*

Эта команда создаёт новый подкаталог в исходном каталоге. После нажатия F7 просто введите желаемое имя каталога. Теперь вы можете также создавать и несколько подкаталогов за одну операцию. Просто отделите подкаталоги обратной косой чертой (обратный слэш), например: каталог1\каталог2\каталог3. Кроме того, вы можете создавать несколько подкаталогов в одних и тех же или в разных каталогах. Синтаксис для использования: каталог1|каталог2|каталог3 или c:\каталог1|c:\каталог2|c:\каталог3.

Если исходный каталог отображает содержимое архива (кроме ZIP), эта функция недоступна.

#### *Удаление*

Выделите файлы и/или каталоги, которые хотите удалить, и нажмите F8. После подтверждения файлы удаляются. Процесс может быть прерван в любой момент кнопкой 'Отмена'. Для каждого непустого каталога будет запрашиваться подтверждение в дополнительном диалоговом окне.

В Windows файлы автоматически перемещаются в Корзину. Вы можете дважды щёлкнуть по значку Корзины на Рабочем столе, чтобы восстановить эти файлы или удалить их окончательно. Вы можете также выбрать, сколько места в Корзине могут занимать удалённые файлы, а также вообще отказаться от её использования. Если вы хотите удалить файлы, минуя Корзину, вы можете удерживать клавишу SHIFT при нажатии F8 или DEL.

Если исходный каталог отображает содержимое архива, для удаления файлов запускается соответствующий упаковщик. В этом случае файлы удаляются сразу, минуя Корзину.

В 32-битной версии операция удаления может быть переведена в фоновый режим нажатием кнопки 'В фоне' во время операции удаления.

### *Настройка программы*

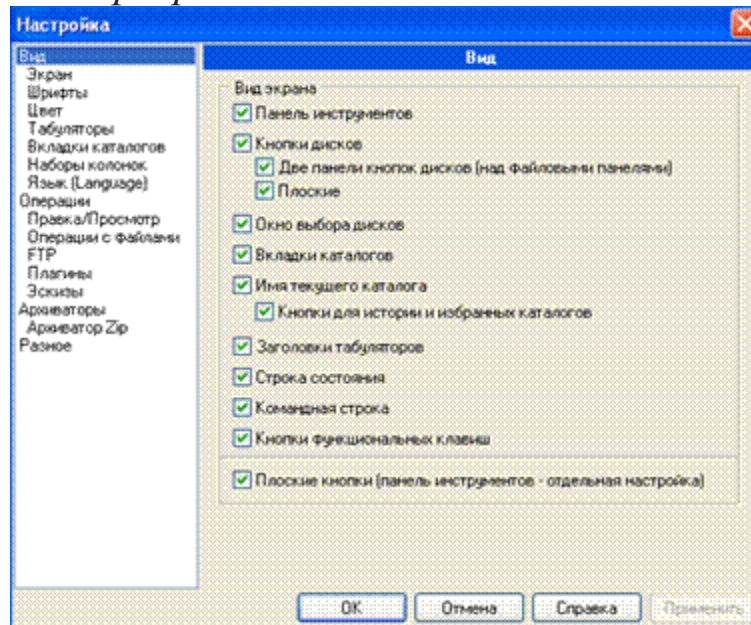


Рисунок 28 – Диалоговое окно настройки параметров программы Total Commander

Настройка программы осуществляется в диалоге Настройка (см. рисунок 28), открываемом командами меню Конфигурация F Настройка. Большинство параметров может быть изменено на одной из страниц этого диалога.

Следующие кнопки доступны на всех страницах диалога:

**ОК** Сохраняет изменения, которые вы сделали, в файле wincmd.ini;

**Отмена** Игнорирует изменения и возвращает в Total Commander;

**Справка** Открывает эту страницу справки;

**Применить** Применяет текущие параметры настройки к обеим файловым панелям без закрытия диалогового окна.

Для настройки определенных элементов программы щелкните соответствующую ссылку в левой панели, а затем установите нужные флажки и переключатели.



### 5.2.4 Выводы

1. Программа Проводник – служебная программа, относящаяся к категории *файловых менеджеров* и предназначенная для навигации по файловой структуре компьютера и ее обслуживания. Проводник глубоко интегрирован в ОС. Программа Проводник управляет файлами и обеспечивает один из способов просмотра файловой системы. Она позволяет открывать папки, копировать, перемещать, удалять, переименовывать и упорядочивать папки и файлы, запускать программы на выполнение, открывать документы и многое другое.

2. Основным средством выполнения операций с объектами в окне программы Проводник является мышь, для которой существует ряд приемов работы: наведение указателя, щелчок, двойной щелчок, перетаскивание, протягивание, правый щелчок, специальное перетаскивание.

3. Total Commander – файловый менеджер, предназначенный для работы в среде Windows и обладающий удобным пользовательским интерфейсом. Придерживаясь традиции, идущей от Norton Commander, в нем реализована концепция двухпанельного окна. Утилита реализует ряд дополнительных функций, в том числе, поддержку работы со многими форматами архиваторов.

4. Far Manager – программа управления файлами и архивами в ОС семейства Windows. Она работает в текстовом режиме и позволяет просто и наглядно выполнять большинство необходимых действий: просматривать файлы и каталоги, редактировать, копировать и переименовывать файлы, а также многое другое. Far Manager имеет многоязычный, легко настраиваемый интерфейс, обеспечивает обработку файлов с длинными именами. Простую навигацию по файловой системе обеспечивают цветовое выделение и группы сортировки файлов. Функциональность менеджера существенно расширяется за счет внешних подключаемых DLL-модулей – плагинов.

## 5.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;

- выполнить выданное задание;
- ответить на контрольные вопросы.

#### Задание

1. Запустите программу FAR Manager.
2. Создайте ярлык программы на Рабочем столе.
3. Изучите состав и назначение команд меню FAR Manager.
4. Ознакомьтесь с названиями кнопок панели инструментов.
5. Настройте окно программы FAR Manager под свои потребности.
6. Откройте диск C:. Расположите объекты окна по имени, типу, размеру или дате. Ознакомьтесь с содержанием логического диска C:, представив его на правой панели в виде дерева.
7. Создайте на диске D: папку Персональная-3.
8. Из папки D:\KITскопируйте в папку Персональная-3 файлы ПЗ\*.doc.
9. Переименуйте эти файлы в prakt\_#.doc.
10. Скопируйте папку Персональная-3 в папку Мои документы и переименуйте ее в Персональная-копия-3.
11. Удалите все файлы из папки Персональная-копия-3 и скопируйте в нее все файлы, найденные в предыдущем пункте..

### 5.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Определите назначение и возможности программы Проводник.
2. Объясните назначение элементов интерфейса программы Проводник.
3. Как осуществляется настройка интерфейса программы Проводник?
4. Как осуществляется поиск объектов файловой системы в программе Проводник?
5. Какие приемы работы с мышью используются при выполнении операций с объектами в окнах Windows?
6. Как осуществляется «тонкая настройка» Проводника?
7. Определите назначение и основные характеристики утилиты Total Commander.

8. Объясните назначение основных элементов интерфейса программы Total Commander.

9. Определите назначение основных клавиатурных команд программы Total Commander.

10. Как осуществляется выделение объектов в программе Total Commander?

11. Как просмотреть содержимое текстового файла в программе Total Commander и как можно отредактировать такой файл?

12. Как осуществляется копирование, перемещение, переименование и удаление файлов в программе Total Commander?

13. Как осуществляется поиск нужных объектов в программе Total Commander?

14. Как осуществляется настройка программы Total Commander?

15. Определите назначение и основные характеристики утилиты FAR manager.

16. Объясните назначение основных элементов интерфейса программы FAR manager.

17. Определите назначение основных клавиатурных команд программы FAR manager.

18. Как осуществляется выделение объектов в программе FAR manager?

19. Как просмотреть содержимое текстового файла в программе FAR manager и как можно отредактировать такой файл?

20. Как осуществляется копирование, перемещение, переименование и удаление файлов в программе FAR manager?

21. Как осуществляется поиск нужных объектов в программе FAR manager?

22. Как осуществляется настройка программы FAR manager?

## 6. РАБОТА С КОМАНДАМИ В ОПЕРАЦИОННОЙ СИСТЕМЕ. ИСПОЛЬЗОВАНИЕ КОМАНД РАБОТЫ С ФАЙЛАМИ И КАТАЛОГАМИ. РАБОТА С ДИСКАМИ

### 6.1 ЦЕЛЬ РАБОТЫ

Цель работы – изучение функционирования операционной системы и приобретение практических навыков работы в ней с файлами и директориями при помощи основных команд (на примере MS DOS).

### 6.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

#### *6.2.1 Характеристика MS DOS. Организация доступа к файлу*

Способ хранения файлов на диске и организацию доступа к ним можно сравнить соответственно с организацией хранения книг в библиотеке и процедурой поиска нужной книги по ее шифру из каталога.

**Доступ** – процедура установления связи с памятью и размещенным в ней файлом для записи и чтения данных.

**Директория (каталог)** – справочник (список) файлов с указанием месторасположения на диске.

Различают два состояния директории – активное (текущее) и пассивное. MS DOS помнит текущую директорию на каждом логическом диске.

**Текущая директория** – это директория, в которой работа пользователя производится в текущее машинное время.

**Пассивная директория** – это директория, с которой в данный момент времени не имеется связи.

В ОС MS DOS принята иерархическая структура организации директорий (каталогов). На каждом диске всегда имеется главная (корневая) директория. Она находится на нулевом (высшем) уровне иерархии и обозначается символом '\'. Корневая директория создается при форматировании (инициализации, разметке) диска и не может быть удалена средствами MS DOS. В корневую директорию могут входить

другие директории (директории первого уровня) и файлы, которые создаются и удаляются командами ОС. В свою очередь, в директории первого уровня могут входить поддиректории (директории второго уровня) и т.д.

**Родительская директория** – это директория, имеющая поддиректории.

**Поддиректория** – это директория, которая входит в другую директорию.

Как правило, употребляют термин "директория" ("каталог"), подразумевая поддиректорию (подкаталог) или родительскую директорию (родительский каталог) в зависимости от контекста.

Правила наименования директорий такие же, как и правила наименования файлов. Для формального отличия от файлов обычно директориям присваивают только имена.

Доступ к содержимому файла организован из главной директории через цепочку соподчиненных директорий  $n$ -го уровня. В директориях любого уровня могут храниться записи как о файлах, так и о директориях нижнего уровня.

Описанный принцип организации доступа к файлу через директорию является основой файловой системы.

**Файловая система** – часть ОС, управляющая размещением и доступом к файлам и директориям на диске.

С понятием файловой системы связано понятие *файловой структуры* диска, под которой понимают, как размещаются на диске директории, файлы, ОС, а также какие для них выделены объемы памяти.

Доступ к файлу можно организовать следующим образом:

- если имя файла зарегистрировано в текущей директории, то достаточно указать только его имя (полное имя);
- если имя файла зарегистрировано в пассивной директории, то, находясь в текущей директории, нужно указать еще и путь.

**Путь** – цепочка соподчиненных директорий, которую необходимо пройти по иерархической структуре к директории, где зарегистрирован искомый файл.

При задании пути имена директорий записываются в порядке следования и отделяются друг от друга символом '\'.

Взаимодействие пользователя с ОС осуществляется с помощью командной строки, индицируемой на экране дисплея. В начале командной строки всегда имеется *приглашение*, которое заканчивается символом '>'. В приглашении может быть отражено: имя текущего диска, имя текущей директории, символы-разделители, текущее время и дата, путь.

**Приглашение ОС** – индикация на экране дисплея информации, означающей готовность ОС к вводу команд пользователя.

Возможны три варианта организации пути доступа к файлу в зависимости от места его расположения. Файл находится в текущей директории (путь отсутствует). При организации доступа к файлу достаточно указать его полное имя.

1. Файл находится в пассивной директории одного из нижних уровней, подчиненного текущей директории. При организации доступа к файлу необходимо указать путь, в котором перечислены имена всех директорий нижнего уровня, лежащих на этом пути (включая директорию, в которой находится данный файл).

2. Файл находится в пассивной директории на другой ветке по отношению к местонахождению текущей директории. Здесь необходимо указать путь, начиная с корневой директории, то есть с символа '\'. Горизонтальные переходы из директории в директорию недопустимы.

### 6.2.2 Модульная структура MS DOS. Модули ОС MS DOS

Понятие модуля широко используется применительно как к аппаратной, так и к программной части компьютера.

**Модуль** – унифицированная самостоятельная функциональная часть системы, имеющая законченное оформление и средства сопряжения с другими функциональными узлами и модулями.

Структуру ОС MS DOS образуют следующие модули:

- 1) BIOS (Basic Input/Output System) – базовая система ввода-вывода;
- 2) модуль расширения – EM BIOS (Extension Module BIOS) в виде файла с именем IO.SYS;
- 3) системный загрузчик (SB – System Bootstrap);

4) внешние драйверы – файлы с расширением .COM, .EXE, .SYS;

5) базовый модуль (BM – Basic Module) в виде файла с именем MSDOS.SYS;

6) командный процессор или интерпретатор команд (CI – Command Interpreter) в виде файла с именем COMMAND.COM;

7) внешние команды, утилиты – файлы с расширением .COM, .EXE, .SYS;

8) инструментальные средства DOS: система программирования MS DOS QBASIC; текстовый редактор MS DOS EDITOR; отладчик DEBUG для тестирования и отладки исполняемых файлов.

Первые четыре модуля составляют *машинозависимую* часть ОС, а последние четыре модуля – *машинонезависимую* часть ОС.

### 6.2.3 Система прерываний

Основным механизмом функционирования MS DOS является система прерываний.

**Прерывания** – это процедуры, которые компьютер вызывает для выполнения определенной задачи.

Различают аппаратные, логические и программные прерывания.

Аппаратные прерывания инициируются аппаратурой, например, сигналом от принтера, нажатием клавиши на клавиатуре, сигналом от таймера и т.д.

Логические прерывания возникают при нестандартных ситуациях в работе микропроцессора, например, деление на ноль, переполнение регистров и т.д.

Программные прерывания инициируются программами, т.е. возникают, когда программа ждет получения сервиса со стороны другой программы, например, доступ к определенным аппаратным средствам.

#### *6.2.4 Функции и назначение модулей системы*

##### *Функции и назначение базовой системы ввода-вывода*

BIOS находится в постоянной памяти, которая входит в комплект поставки ПК. Тип ОС может изменяться, а BIOS остается постоянным.

BIOS устанавливает связь между техническими средствами и стандартизированным программным обеспечением (ПО), а именно ОС. BIOS содержит специальные программы (драйверы) по управлению работой стандартными внешними устройствами. Назначение BIOS состоит в выполнении наиболее простых и универсальных функций ОС, связанных с вводом-выводом. BIOS содержит также: тест функционирования ПК, проверяющий работу памяти и устройств после включения питания, программу загрузки ОС. BIOS – общая (неизменяемая) часть всех ОС для данной модели ПК. Системный загрузчик считывает в оперативную память модуль расширения BIOS и модуль обработки прерываний.

##### *Функции и назначение модуля расширения BIOS*

Модуль расширения BIOS придает гибкость ОС, позволяет управлять с ее помощью набором аппаратных средств ПК. Этот модуль можно модифицировать с учетом необходимых нужд конкретной версии MS DOS.

Модуль позволяет перекрыть функции BIOS в постоянном запоминающем устройстве и обеспечивает возможность подключения дополнительных драйверов (программ обслуживания внешних устройств). Основная функция этого модуля – это увеличение возможностей BIOS.

##### *Функции и назначение базового модуля*

Основная функция базового модуля – управление ресурсами ПК, файловой системой, работой программ при помощи системы прерываний. Функциями базового модуля на этапе загрузки являются: считывание в память и запуск командного процессора, инициализация векторов прерываний верхнего уровня.

##### *Функции и назначение командного процессора*

Командный процессор на диске может занимать любое место и, по сути, представляет собой выполняемую программу. Командный процессор выполняет в ПК следующие функции:



- прием и разбор команд с клавиатуры или из командного файла;
- выполнение команд MS DOS, находящихся внутри файла COMMAND.COM;
- загрузка и выполнение внешних команд MS DOS (утилит) и прикладных программ, хранящихся в виде файлов с расширением COM и EXE.

Программы с расширением COM не требуют настройки адресов после их загрузки в оперативную память, а с расширением EXE – настраиваются по месту размещения (для них задаются соответствующие адреса сегментов).

При загрузке в оперативную память командный процессор распадается на две части:

- резидентную часть, постоянно размещаемую в оперативной памяти;
- нерезидентную (транзитную) часть, периодически изменяемую путем передачи данных между оперативной памятью и диском.

Резидентная часть содержит подпрограммы стандартной обработки прерываний. Здесь же находятся: программа подзагрузки нерезидентной части в оперативную память и подпрограмма, обрабатывающая файл AUTOEXEC.BAT при запуске ПК.

#### *Назначение загрузчика*

Загрузчик BOOT RECORD (модуль начальной загрузки) всегда размещается на диске в нулевом секторе. Основное назначение загрузчика – поиск и перезапись (загрузка) с диска в оперативную память двух файлов IO.SYS и MSDOS.SYS, а также запуск модуля расширения базовой системы ввода-вывода.

#### *Утилиты, внешние команды и драйверы*

**Утилиты** – обслуживающие программы, поставляемые вместе с ОС в виде файлов и предоставляющие пользователю сервисные услуги (форматирование дискет, проверку дисков и т.д.).

**Внешней командой** принято считать программу, выдающую пользователю ряд простых запросов или выполняющуюся автоматически без специально организованного интерфейса пользователя. MS DOS имеет определенный перечень внешних команд.

**Внешние драйверы** – программы, дополняющие систему ввода-вывода и обеспечивающие обслуживание новых устройств или нестандартное использование имеющихся устройств. Драйверы загружаются в оперативное запоминающее устройство при загрузке ОС, а их имена указываются в файле конфигурации CONFIG.SYS.

*Загрузка MS DOS в оперативную память с диска*

ОС хранится во внешней памяти на жестком или (реже) на гибком диске. Для работы ПК необходимо, чтобы основные модули ОС находились в оперативной памяти. Поэтому после включения ПК организована автоматическая перезапись (загрузка) ОС с диска в оперативную память.

Запуск ПК и подготовка ОС к работе включает следующие шаги:

1. При включении ПК управление передается базовой системе ввода-вывода BIOS. BIOS выполняет тестирование памяти, проверку состояния аппаратуры и инициализирует устройства. Параметры конфигурации ПК извлекаются из так называемой энергонезависимой памяти. При нажатии клавиши <Del> перед инициализацией устройств можно передать управление программе изменения параметров конфигурации.

2. Управление конфигурацией ПК (задание параметров жесткого диска, указание системного диска, задание пароля) выполняется с помощью программы Setup.

3. Вызов загрузчика (BOOT RECORD) и загрузка с его помощью в оперативную память модуля расширения IO.SYS и базового модуля MSDOS.SYS.

4. Загрузка командного процессора COMMAND.COM.

5. Обработка файла конфигурации CONFIG.SYS, содержащего команды подключения необходимых драйверов.

6. Обработка командного файла AUTOEXEC.BAT. С помощью этого файла можно произвести настройку параметров ОС. Например, создать виртуальный диск, обеспечить смену режимов печати, загрузить вспомогательные программы и т.д.

### 6.2.5 Технология работы в MS DOS. Общие сведения о командах

Работа в ОС MS DOS организуется *командами*. Они вызывают определенное действие: организуют передачу информации, вырабатывают необходимый управляющий сигнал, подключают внешнее устройство для организации процесса ввода-вывода информации и т.д.

Команда технически реализована программой в машинных кодах и хранится либо в файле на диске (внешняя команда), либо входит в состав командного процессора COMMAND.COM (внутренняя команда). По порядку запуска внутренние и внешние команды не различаются. При запуске внешних команд необходимо удостовериться, что файлы, в которых они находятся, существуют на диске и находятся на "видимой" (компьютеру) директории. Как и любая другая программа, команда имеет уникальное имя и всегда имеет тип COM или EXE.

Ввод команды осуществляется в *командной строке* в соответствии с определенными правилами, заданными в виде *формата*.

**Командная строка** – строка экрана дисплея, начинающаяся с приглашения ОС. Командная строка состоит из информации подсказки, указателя ввода и курсора. Обычная информация подсказки указывает на диск и директорию, где в это время производится работа.

**Формат команды** – правило формирования команды пользователем с клавиатуры.

При формировании команды в соответствии с установленным форматом необходимо соблюдать следующие правила:

- 1) формат команды состоит из имени команды (латинскими буквами без указания типа) и отделенных от него одним пробелом параметров, уточняющих действие команды;
- 2) в большинстве случаев параметры между собой пробелом не разделяются, а в качестве разделителя часто используется символ '/';
- 3) параметрами могут быть: имя логического диска, путь, имя файла, тип файла, латинские буквы, символы, цифры;

4) параметры в формате могут и отсутствовать, что указывается с помощью квадратных скобок '[' и ']'.  
 Обобщенный формат команды можно представить в следующем виде: <имя команды> [<параметры>].

**Пример:** C:\>DIR D:\USER\\*.TXT/P

Здесь:

C:\> – приглашение ОС MS DOS;

DIR – имя команды;

D:\USER\\*.TXT/P – параметры.

Эта команда вызывает с помощью параметра D:\USER\\*.TXT/P на экран записи обо всех файлах типа 'TXT' из директории первого уровня 'USER' логического диска 'D'. Вызов записей производится постранично, на что указывает параметр '/P'.

Процедура ввода команды состоит в следующем:

1) в соответствии с форматом в командной строке набирают имя команды и необходимые параметры;

2) нажимают клавишу ввода, что служит сигналом начала анализа структуры набранной команды. При отсутствии ошибок в формате команды она будет выполнена, иначе на экран выдается сообщение: *Bad command or filename* (Неверная команда или имя файла);

3) при невыполнении команды просматривают вводимую конструкцию и вновь вводят ее, но уже в откорректированном варианте.

#### *Порядок действий при выполнении команды MS DOS*

После ввода команды с клавиатуры MS DOS выполняет следующие действия:

1. MS DOS анализирует первое слово командной строки (последовательность символов до первого пробела) с целью выяснить – задано ли просто имя, неполный адрес или точный адрес. Основным ключ к анализу – наличие символов ':' и '\'.  
 2. Если задано просто имя, MS DOS ищет его сначала в файле COMMAND.COM, затем в текущей директории, затем в директориях, перечисленных в команде PATH, записанной в файле автозапуска AUTOEXEC.BAT. Если расширение в имени опущено, поиск ведется по собственному имени с подстановкой расширения в следующем порядке: COM, EXE, BAT.

3. Если задан неполный адрес, MS DOS ищет программу либо на текущем диске, либо начиная с текущей директории. Например, если указано: \SIMP\REM.EXE, то поиск файла REM.EXE ведется в каталоге SIMP текущего диска. Если указано: C\SIMP\REM.EXE, то MS DOS ищет файл REM.EXE в поддиректории 'C' текущей директории.

4. Если задан точный адрес, то MS DOS просто следует по указанному пути, не обращая внимания ни на текущий каталог, ни на директории, перечисленные в команде PATH.

5. Найдя программу, MS DOS загружает ее и передает ей в качестве параметров все, что набрано в командной строке.

6. После завершения программы на экран вновь выводится приглашение MS DOS.

7. Если программа не найдена, на дисплей поступает сообщение: *Bad command or filename* (имя команды или файла указано неверно), и выдается приглашение MS DOS.

#### *Команды MS DOS общего назначения*

По мере необходимости пользователь может использовать следующие команды, называемые командами общего назначения:

1. CLS – очистка экрана от выведенной до этого информации.

2. ECHO <сообщение> – печать сообщения на экране. Команды ECHO OFF и ECHO ON соответственно запрещают и разрешают печать сообщения других команд.

3. DATE – вывод на экран или установка текущей даты в формате "мм-дд-гг".

4. TIME – вывод на экран или установка системного времени в формате "чч:мм".

5. PROMPT \$<тип информации>\$<вид указателя> – определение системной подсказки. Тип информации задается символами: D – текущая дата, P – текущий диск и путь, N – только текущий диск, T – текущее время. Вид указателя задается символами: G (на экране появится символ '>') или L (на экране появится символ '<'). Обычно эту команду используют в следующем виде: PROMPT \$P\$G.

6. VER – вывод на экран номера версии ОС на этом ПК.

### 6.2.6 Основные команды для работы с директориями

#### Команда DIR – просмотр директории

Работа на ПК, как правило, начинается с просмотра директории, например, чтобы убедиться в том, что нужный вам файл существует. Часто необходимо просмотреть содержимое пассивной директории.

В зависимости от параметров, допустимых в структуре команды, можно просмотреть записи директории в стандартной или усеченной форме с выводом только полных имен файлов, а также при большом содержании директории выводить ее постранично.

Формат команды:

DIR [Имя дискового:] [Путь\] [Имя файла] [Параметры]

Если имя дискового и/или путь отсутствуют в команде, то подразумевается текущий дисковод и текущая директория. Параметры (ключи) задают порядок вывода списка файлов и директорий.

Назначение основных параметров (ключей):

/P – постраничный вывод содержимого директории на экран. Для продолжения вывода следует нажать любую клавишу;

/W – вывод только полных имен файлов и директорий;

/A – индикация содержимого директорий с атрибутами;

/O – задание порядка сортировки выводимых сведений.

Примеры:

C:\>DIR	Вывод содержимого корневой текущей директории на экран
C:\USER1>DIR *.BAK	Вывод на экран всех имен файлов типа BAK из текущей директории первого уровня USER1
C:\>DIR A:	Вывод на экран содержимого пассивного дисковода A
C:\B1>DIR B2	Вывод на экран содержимого пассивной директории второго уровня B2, находящейся в директории первого уровня B1
C:\USER1>DIR \B1\B2/P	Вывод на экран постранично содержимого пассивной директории B2. Эта директория находится в другой

	ветке иерархической структуры директории, чем текущая директория USER1
C:\B1>DIR /W	Вывод на экран записей текущей директории B1 в усеченном формате (только полные имена файлов и директорий)

Команда MD – создание директории

Новую директорию можно создать командой MD в текущей директории или, если указан путь, в пассивной директории.

Формат команды:

MD [Имя дискового:] [Путь\]Имя директории

Примеры:

C:\>MD USER1	Создание (в текущей корневой директории) директории первого уровня USER1
C:\>MD USER1\USER2	Создание (в директории первого уровня USER1) директории второго уровня USER2
C:\T1\T2>MD USER1\USER2	Создание директории второго уровня USER2, если ОС находится в другом каталоге второго уровня T2

Команда RD – уничтожение директории

Эта команда уничтожает только пустую директорию. Предварительно необходимо удалить из нее командой DEL все файлы, а затем командой DIR убедиться в том, что она пустая.

Формат команды:

RD [Имя дискового:] [Путь\]Имя директории

Примеры:

C:\>RD USER1	Удаление директории USER1 из корневой директории
--------------	--

C:\B1>RD  
 \USER1\USER2

Удаление пассивной директо-  
 рии второго уровня USER2,  
 если ОС находится в текущей  
 директории B1

Команда CD – переход в другую директорию

Иногда необходимо перейти в другую директорию и сделать ее текущей. В этом случае следует воспользоваться командой CD.

Формат команды:

CD [Имя дисководы:][Путь\]Имя директории

Для перехода в родительскую директорию достаточно вместо имени директории задать '..' (две точки). Для перехода в корневую директорию задают символ '\'.  
 Примеры:

C:\>CD USER1	Переход в директорию USER1 из корневой директории. После ввода команды приглашение примет вид: C:\USER1>
C:\B1\B2>CD \USER1	Переход из директории второго уровня B2 в директорию первого уровня USER1, находящуюся в другой ветви иерархической структуры. После ввода команды приглашение примет вид: C:\USER1>
A:\>CD C:\USER1	Переход из корневой директории диска A в директорию первого уровня диска C. После ввода команды приглашение примет вид: C:\USER1>
C:\M1\M2\M3>CD ..	Переход в родительскую директорию. После ввода команды приглашение примет вид: C:\M1\M2>



C:\F1\F2>CD \

Переход в корневую директорию. После ввода команды приглашение примет вид: C:\>

### 6.2.7 Основные команды для работы с файлами

Команда TYPE – просмотр текстового файла

Командой TYPE удобно пользоваться для просмотра содержимого текстового файла на экране дисплея или на принтере. После запуска команды текст (содержимое файла) выводится непрерывным потоком, причем скорость смены кадров с текстом на экране настолько велика, что прочесть его практически невозможно. Для приостановки вывода текста нажимают одновременно две клавиши: <CTRL> и <S>. Нажатие затем любой клавиши возобновит вывод текста.

Формат команды для вывода на экран:

TYPE [Имя дисковода:][Путь\]Полное имя файла

Формат команды для печати на принтере:

TYPE [Имя дисковода:][Путь\]Полное имя файла>PRN

Примеры:

C:\>TYPE ROK.TXT

Вывод на экран содержимого файла ROK.TXT, расположенного в корневой директории активного дисковода

C:\>TYPE A:\RED\LOT.TXT

Вывод на экран содержимого файла LOT.TXT, расположенного в директории RED первого уровня пассивного дисковода A

C:\>TYPE \B1\BOOK.TXT>PRN

Печать на принтере содержимого файла BOOK.TXT, расположенного в директории B1 первого уровня текущего дисковода C

Команда DEL – удаление файлов

Можно удалять как один файл, так и группу файлов, указывая для группы в шаблоне имени файла символы '\*' или '?'.

Формат команды:

DEL [Имя дискового:] [Путь\] Полное имя файла> [/P]

Параметр [/P] служит для вывода на экран запроса на подтверждение удаления.

При вводе команды удаления всех файлов (DEL \*.\* ) ОС задает вопрос *Are You sure (Y/N)? (Вы уверены?)*. Если вы не передумали, нажмите клавишу <Y>, в противном случае – клавишу <N>.

Примеры:

C:\>DEL TOST.TXT	Удаление файла TOST.TXT из корневой директории текущего дискового C
C:\>DEL A:\AR\B.TXT	Удаление файла B.TXT из директории первого уровня AR пассивного дискового A
C:\>DEL \A1\A2*.BAS	Удаление всех файлов типа BAS из директории второго уровня A2 текущего дискового C
C:\F1>DEL F2\*.* /P	Удаление с подтверждением всех файлов из директории второго уровня F2 текущего дискового C, подчиненной текущей директории F1

Команда COPY – копирование файлов

Команда используется для создания копий существующих файлов, вывода содержимого файла на внешнее устройство, объединения содержимого нескольких файлов.

Команда COPY допускает выполнение функций над группой файлов, и тогда в шаблоне имени файла используются символы '\*' или '?'.

Формат команды для копирования файлов:

COPY [Имя дискового-источника:] [Путь\] Полное имя файла-источника [Имя дискового-приемника:] [Путь\] [Полное имя файла-приемника] [N]

Обязательным параметром является только полное имя файла-источника. Если копируется файл (группа файлов) с тем же именем (именами), то достаточно указать только полное имя

файла-источника. Имя дисководов и путь нужно указывать при работе с пассивным дисководом и директорией.

Примеры:

C:\>COPY ROK.PAS A:	Копирование файла ROK.PAS из корневой директории текущего дисковода C на диск A с тем же именем
C:\>COPY A1\A2\P.TXT \B1\B2\B3	Копирование файла P.TXT из директории A2 второго уровня в директорию третьего уровня B3 с тем же именем
C:\>COPY A:ROST.BAS	Копирование файла ROST.BAS с пассивного дисковода A в корневую директорию текущего дисковода C с тем же именем
C:\>COPY A:T.TXT A1\A2\S.TXT	Копирование файла T.TXT с диска пассивного дисковода в директорию второго уровня A2 текущего дисковода. Полученной копии файла присваивается новое имя S.TXT
C:\A1>COPY *.BAS B:/V	Копирование всех файлов типа BAS из текущей директории первого уровня A1 дисковода C на диск пассивного дисковода B с одновременным контролем процесса копирования

Формат команды объединения нескольких файлов:

COPY [Имя дисковода:][Путь\]Полное имя файла + [Имя дисковода:][Путь\]Полное имя файла + ... [Имя дисковода:][Путь\]Полное имя файла-приемника

Имена объединяемых файлов перечисляются в команде COPY через знак '+'.  
Имя результирующего файла записывается последним и отделяется от имен объединяемых файлов пробелом.

Содержимое результирующего файла представляет собой подсоединенное друг за другом содержимое исходных файлов в соответствии с порядком следования их имен.

Примеры:

C:\>COPY M1.TXT+ M2.TXT \K1\SUM.TXT	Объединение двух текстовых файлов M1.TXT и M2.TXT в файл SUM.TXT, который будет записан в директорию первого уровня K1
C:\>COPY T1.TXT+T2.TXT	Объединение двух текстовых файлов T1.TXT и T2.TXT. К содержимому файла T1.TXT добавляется содержимое файла T2.TXT, и результат объединения будет храниться в файле с именем T1.TXT

Форматы команд для обмена данными между внешним устройством и файлом, хранящимся на диске:

COPY Имя внешнего устройства (откуда) [Имя дисководов:] [Путь\] Полное имя файла (куда)

COPY [Имя дисководов:] [Путь\] Полное имя файла (откуда) Имя внешнего устройства (куда)

COPY Имя внешнего устройства (откуда) Имя внешнего устройства (куда)

Под внешним устройством здесь понимается любое устройство, кроме системного блока и дисководов. В ОС приняты соглашения относительно имен внешних устройств. К наиболее употребительным относятся:

- CON – клавиатура и дисплей (консоль);
- PRN или LPT1 – основной принтер.

Примеры:

C:\>COPY T.TXT PRN	Печать содержимого текстового файла на принтере
--------------------	---

C:\>COPY CON S.TXT	Заполнение файла S.TXT поступающими с клавиатуры символами
C:\>COPY CON PRN	Все символы, набираемые с клавиатуры, печатаются, минуя центральную часть компьютера, т.е. компьютер используется как пишущая машинка. Одновременно компьютер может обрабатывать информацию в соответствии с программой, где не требуется обращение к принтеру

### 6.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

Образец задания:

1. На диске С создать директорию ТЕСТ. В этой директории создать директорию КАТЕР, а в ней создать директорию КОРВЕТ. Внутри последней директории создать две поддиректории КОМ и РЕМАРКА.

2. Скопировать в поддиректорию КОМ все файлы с расширением TRU из директории C:\ТР или ТР7\BIN. Скопировать в поддиректорию РЕМАРКА файлы с расширением PAS, имеющие 7 и менее символов в своем имени и находящиеся в директории ТР или ТР7 или ее поддиректориях.

3. Соединить три файла в поддиректории РЕМАРКА в один файл с именем СТАКАН.TXT.

4. Переименовать файл с наименьшим размером в поддиректории КОМ в файл с именем СОРТ.TXT.

5. Удалить поддиректорию КОМ.

## 6.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое файл, характеристики файла?
2. Понятие имени файла и полного имени файла.
3. Каковы правила при задании имени файла в MS DOS?
4. Какие символы используются в шаблоне имени файла?
5. Доступ и три способа организации доступа к файлу.

## **7. КОНФИГУРИРОВАНИЕ ФАЙЛОВ. УПРАВЛЕНИЕ ПРОЦЕССАМИ В ОПЕРАЦИОННОЙ СИСТЕМЕ. РЕЗЕРВНОЕ ХРАНЕНИЕ, КОМАНДНЫЕ ФАЙЛЫ**

### **7.1 ЦЕЛЬ РАБОТЫ**

Цель работы – получение практических навыков резервного копирования данных, программирования командных файлов, резервного копирования, программирования командных файлов (на примере MS DOS).

### **7.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

#### *7.2.1 Файл конфигурации CONFIG.SYS*

Удобная рабочая обстановка для пользователя ПЭВМ может быть создана в результате продуманного конфигурирования и начальной настройки системы. DOS дает возможность при запуске машины автоматически задавать определенные начальные условия, которые влияют на дальнейшую работу пользователя. Осуществляется это с помощью содержимого двух особых файлов: CONFIG.SYS и AUTOEXEC.BAT. Указанные файлы, если они имеются в корневом каталоге системного диска, обрабатываются при любом включении или перезапуске машины.

С помощью файла конфигурации CONFIG.SYS можно расширять операционную систему и изменять некоторые параметры, влияющие на работу внешних устройств. Одно из важнейших свойств DOS состоит в возможности добавления новых внешних устройств и подключения специальных программ, обеспечивающих управление их работой.

Эти программы, называемые драйверами внешних устройств, можно включить в систему, перечислив их в файле CONFIG.SYS. Помимо этого в файле конфигурации можно указать, сколько файлов в системе может быть открыто одновременно, задать количество буферов для обменов с внешними накопителями и некоторые другие параметры.

Пример. Рассмотрим типичный вид файла CONFIG.SYS:

```
break=on  
files=30
```

```
device=C:\sys\rk.com
device=C:\sys\vdisk.sys
device=C:\sys\ansi.sys
device=C:\sys\mouse.sys
```

В первой строке с помощью выражения `BREAK=ON` устанавливается режим, при котором пользователь будет иметь возможность прервать любую работающую программу при определенных условиях. Для этого дается команда прерывания, вызываемая одновременным нажатием управляющих клавиш `Ctrl` и `Break`. Работающая программа будет прервана, если эти клавиши оказываются нажатыми в момент выполнения операций ввода/вывода (включая печать на принтере, обмен с дисками и др.).

Во второй строке командой `FILES=30` устанавливается, что количество одновременно открытых файлов может достигать 30.

Четыре следующие строки имеют одинаковый вид и содержат команды подключения к DOS дополнительных внешних устройств. В правой части каждого выражения, после знака равенства, задается имя файла, являющегося драйвером нового устройства:

`RK.COM` – драйвер, обеспечивающий прием с клавиатуры и выдачу на дисплей букв русского алфавита;

`VDISK.SYS` – драйвер виртуального диска, создаваемого в оперативной памяти;

`ANSI.SYS` – драйвер расширенного управления клавиатурой и дисплеем; `MOUSE.SYS` – драйвер манипулятора "мышь".

Кроме указанных в примере команд, в файле конфигурирования можно установить нестандартное количество буферов для обмена информацией с дисковыми накопителями. Задание числа буферов делается с помощью выражения вида;

```
BUFFERS=(количество_буферов)
```

Еще одна возможность позволяет указывать имя файла, который будет играть роль нестандартного командного процессора (вместо стандартного файла `COMMAND.COM`). Такое указание осуществляется с помощью выражения вида:

```
SHELL=(имя_файла_с_новым_командным_процессором)
```

Таким образом, с помощью файла конфигурирования `CONFIG.SYS` пользователь может задать собственные, нестан-



дартные возможности, которые действуют в течение текущего сеанса работы.

### 7.2.2 Командные файлы

Командные файлы принадлежат к категории исполняемых файлов. Такие файлы снабжаются типом .BAT (от английского слова *batch* – пачка). Запуск командного файла осуществляется точно так же, как запуск файла типа COM или EXE: указывается имя файла без расширения и, если требуется, параметры.

Роль командных файлов особая. Они могут содержать целую группу команд DOS или обращений к прикладным программам, которые выполняются последовательно или в более сложном порядке. Командный файл, в отличие от исполняемых файлов остальных двух типов, содержит не машинный код программы, а текст, который интерпретируется командным процессором DOS. Таким образом, по форме это обычный текстовый файл. Его строки могут представлять собой: встроенные команды DOS, обращения к исполняемым программам, вызовы других командных файлов, специальные команды для управления выводом на экран, специальные команды для организации ветвлений и циклов, метки.

Все эти конструкции (за исключением меток) могут не только фигурировать в командных файлах, но и просто вводиться пользователем с клавиатуры. Однако в практической работе лишь команды первых трех типов вводятся пользователем, а остальные конструкции, включая метки, служат главным образом для создания нетривиальных командных файлов.

Рассмотрим несколько простых примеров применения командных файлов.

Пример 1. Допустим, наша регулярная работа осуществляется в каталоге WORK, но время от времени мы хотим переходить в каталог GRAPH, запускать там графический пакет VICONT, а по окончании работы с этой программой возвращаться в каталог WORK. Последовательность команд при этом должна быть следующей:

```
cd \GRAPH
VICONT
```

```
cd \WORK
```

Эту последовательность можно разместить с помощью текстового редактора в командном файле. Каждая команда должна следовать на отдельной строке. Дадим этому файлу имя V.BAT. Достаточно теперь задать в качестве одной команды имя этого файла — V, и содержащиеся в нем несколько команд начнут интерпретироваться одна за другой, избавляя нас от необходимости вводить их текст с клавиатуры. Первая команда сменит каталог, а вторая погрузит нас в операционную среду системы VICON.T. Когда после выполнения необходимых действий мы, наконец, закончим работу с VICON.T, управление вернется в командный файл. Выполнение третьей команды вернет нас в каталог WORK. Заметим, что при выполнении каждой команды в данном примере на экране печатается приглашение, затем текст самой команды (эхо) и, наконец, результат ее работы.

Пример 2. Допустим, мы хотим предельно сократить число манипуляций для создания необходимой рабочей обстановки при запуске системы. Мы можем заставить DOS автоматически провести всю необходимую подготовку операционной среды с помощью командного файла следующего вида;

```
echo off
mode co80
path C:\;C:\EXE;C:\TURBO
cd \WORK
turbo
```

Присвоив этому файлу какое-либо имя, например, START.BAT, мы избавим себя от систематического набора указанных команд: достаточно набирать лишь слово START. Первая команда этого файла ECHO OFF отключает "эхо", т. е. блокирует выдачу на экран приглашений DOS и текстов последующих команд. Три следующие команды осуществляют настройку необходимых параметров (установка режима дисплея, задание альтернативных маршрутов и рабочего каталога), а последняя команда производит запуск системы TURBO Pascal.

Пример 3. Для подготовки принтера к печати русского текста мы можем воспользоваться специальной программой загрузки шрифта, обращение к которой имеет вид: LFONT 0. Перед запуском этой программы нужно сделать подготовительные дей-

ствия – включить принтер, вставить бумагу. Можно создать небольшой командный файл, который будет выдавать на экран соответствующее уведомление и затем запускать программу LFONT с указанным параметром. Вид такого файла:

```
echo off
echo Включите принтер, вставьте бумагу
echo Будет загружен русский шрифт
pause
Ifont 0
```

Дадим этому файлу имя LF.BAT. Для загрузки русского шрифта достаточно набрать команду LF. Исполнение этого командного файла начинается, как и в предыдущем примере, с команды ECHO OFF. Две следующие команды ECHO выводят на экран сообщения, заданные в файле в качестве текстовых строк – аргументов команд: "Включите принтер, вставьте бумагу" и "Будет загружен русский шрифт". Четвертая команда, PAUSE (пауза), приостанавливает дальнейшую обработку файла LF в ожидании, пока пользователь не нажмет какую-либо клавишу. Пауза нужна для того, чтобы можно было не спеша включить принтер и заправить бумагу. Наконец, последняя команда запускает программу LFONT.

В последнем примере были использованы команды ECHO и PAUSE, управляющие выдачей информации на экран. К этой же категории можно отнести команду REM, которая служит для внесения в текст командного файла комментария. Строка, следующая за символом REM, никак не интерпретируется командным процессором – она используется лишь для пояснений пользователю. С помощью команды REM удобно блокировать исполнение некоторых команд, не удаляя их из текста командного файла.

Использование команд ECHO, PAUSE и REM полезно лишь в командных файлах; вводить их с клавиатуры для непосредственного исполнения не имеет особого смысла. Есть и другие команды, предназначенные главным образом для использования в командных файлах. К ним относятся:

```
GOTO – безусловный переход (на метку);
IF – проверка условия и ветвление;
FOR – управление повторным выполнением команд;
SHIFT – сдвиг списка формальных параметров.
```

Отметим еще одно важное свойство командных файлов – возможность использования внутри них формальных параметров.

Для пояснения этой возможности обратимся еще раз к первому примеру. Допустим, нам хотелось бы применять одну и ту же последовательность команд, приведенную в примере 1, для запуска разных подсистем. Тогда можно видоизменить приведенный выше текст файла V.BAT, заменив имя каталога GRAPH и имя вызываемой программы VICONT формальными параметрами. Командный файл при этом приобретает вид:

```
cd \%1
%2
cd \WORK
```

Символы %1 и %2 обозначают формальные параметры, вместо которых при обращении к файлу V.BAT будут подставлены в текстовом виде фактические параметры, указанные в командной строке. Обращение к V.BAT может иметь вид:

```
V graph vicont
```

Строки graph и vicont займут место соответственно параметров %1 и %2, в результате чего получится точно такой же текст, как и в примере 1. Однако новый командный файл, в отличие от прежнего, можно использовать для запуска других подсистем, например:

```
V ss symph
или V ww word
```

В первом случае из каталога SS будет вызвана программа symph, во втором случае из каталога WW вызывается система word.

### *7.2.3 Развитые командные файлы*

Рассмотрим действие специальных команд, позволяющих управлять интерпретацией командных файлов. К ним относятся команды GOTO, IF, FOR, SHIFT, а также команда EXIT, играющая особую роль при рекурсивном вызове командных файлов.

#### **Команда GOTO.**

Позволяет передавать управление на метку и тем самым осуществлять повторное исполнение участков командного файла или, наоборот, обходить некоторые участки (что обычно имеет смысл в сочетании с командой ветвления IF).

Пример 1. Пусть командный файл PR.BAT имеет вид

```
echo off
:m1
echo Вывод на принтер файла %1
echo Для остановки нажмите Ctrl-C
copy %1 prn
pause
goto m1
```

В данном примере вторая строка содержит метку m1 (признаком метки является двоеточие в начале строки), а последняя строка – команду перехода на эту метку. Четыре команды внутри повторяемого участка командного файла служат для выдачи на экран поясняющих сообщений (команды ECHO) и копирования на принтер (т. е. печати) файла, имя которого подставляется вместо формального параметра %1. Команда PAUSE приостанавливает исполнение, чтобы пользователь имел возможность осмотреться и либо прервать работу, либо продолжить её дальше. Допустим, пользователь дает с терминала команду:

```
pr spectr.doc
```

Тогда начинается циклическое исполнение файла PR. BAT с фактическим параметром SPECTR.DOC. На принтер начнут выдаваться одна за другой копии этого текстового файла, прерываемые паузами после каждой очередной копии. Пользователь может прервать исполнение этого командного файла, нажав одновременно клавиши Ctrl и C.

### **Команда IF.**

Позволяет проверять условие и выполнять команду в зависимости от результата его проверки. Что можно задавать в качестве условия? Для этого имеется три возможности:

1. Проверка кода завершения программы, сработавшей перед оператором IF. При этом в начале оператора IF используется конструкция следующего вида:

```
IF ERRORLEVEL "N"
```

Любая программа может с помощью специального прерывания DOS выработать в момент своего окончания так называемый код завершения. Этот код сравнивается с числом N. Условие

считается выполненным (истинным), если выработанный код завершения равен или больше указанного числа N.

2. Проверка наличия файла в каталоге. В этом случае начало оператора IF имеет вид;

IF EXIST (имя\_файла)

В данном случае команда IF проверяет, действительно ли существует файл с заданным именем в указанном или в текущем каталоге. При обнаружении файла условие считается выполненным.

3. Сравнение двух строк, которые, в частности, могут задаваться через формальные параметры. Соответствующая конструкция может иметь вид;

IF %(N)=(текстовая\_строка)

Здесь значение формального параметра % (N) сопоставляется с конкретной строкой. При абсолютном совпадении двух строк условие считается выполненным.

### **Команда FOR.**

Обеспечивает циклическое выполнение команд DOS. При этом можно задать формальный параметр и список фактических параметров (обычно – имен файлов), которые последовательно подставляются вместо формального параметра в текст исполняемой команды.

Пример 2. Пусть имеется необходимость систематически копировать файлы PROG.PAS, PROG.OBJ и PROG.EXE из рабочего каталога на диск D: . С этой целью можно в соответствующем командном файле дать команду:

for %%A in (PAS OBJ EXE) do copy PROG.%%A D:

Здесь формальный параметр %%A последовательно сопоставляется со списком фактических параметров в круглых скобках и используется командой COPY, которая в данном случае выполняется 3 раза подряд.

Подставив в качестве второго параметра команды COPY вместо имени D: еще один формальный параметр %1, можно будет менять назначение копирования, задавая его с терминала.

### **Команда SHIFT.**

Вызывает сдвиг списка формальных параметров относительно списка фактических параметров. Так, если в командном файле фигурируют формальные параметры %1 и %2, а в обра-  
боте

нии к командному файлу – фактические параметры A, B, C, D, то сначала соответствие формальных и фактических параметров выглядит следующим образом:

```
%1=A %2=B
```

Однократное применения команды SHIFT дает следующее соответствие:

```
%1=B %2=C
```

Двукратное применение вызывает дальнейший сдвиг:

```
%1=C %2=D
```

Пример 3. Рассмотрим задачу, обратную задаче примера 2, а именно, – нужно копировать на диск D: файлы, имена которых пользователь будет вводить с терминала. Эту задачу позволит решить командный файл следующего вида:

```
echo off
:loop
copy %1 D:
shift
goto loop
```

Если присвоить этому файлу имя D.BAT и вызвать его с произвольным числом аргументов:

```
d f.1 f.2 f.3 f.4 f.5
```

то будет происходить последовательное присваивание формальному параметру %1 значений f.1, f.2, f.3, f.4, f.5 и циклическое исполнение команд COPY и SHIFT. После исчерпания фактических параметров команда COPY выдаст сообщение об ошибке, поскольку ее первый аргумент будет отсутствовать. В этот момент пользователь сможет прервать работу командного файла, нажав клавиши Ctrl и C.

Команда SHIFT используется также в тех случаях, когда число параметров командного файла превышает 9, так как в командном файле можно адресоваться к формальным параметрам только от %0 до %9.

#### 7.2.4 Файл автозапуска AUTOEXEC.BAT

С точки зрения содержащейся в файле AUTOEXEC.BAT информации он является обычным командным файлом. Однако он играет особую роль, потому что при начальном запуске и инициализации системы, вслед за обработкой CONFIG.SYS, опера-

ционная система пытается найти в корневом каталоге системного диска файл AUTOEXEC.BAT и начинает его обработку автоматически, без какого-либо побуждения со стороны пользователя. В файл AUTOEXEC.BAT удобно занести различные команды, которые осуществляют всю необходимую настройку системы.

Пример.

Рассмотрим следующий файл AUTOEXEC.BAT:

```
echo off
path C;\;C:\EXE
prompt $p$g
set ABC=C:\ABC
rkvga.com
ver
fkeys.bat
```

Начиная со второй строки этого файла, стоят команды, обеспечивающие создание определенной операционной обстановки. Рассмотрим их по порядку.

1. Команда PATH устанавливает альтернативные маршруты для поиска исполняемых файлов. В примере один из маршрутов указывает на системный подкаталог с именем EXE. Кроме того, как возможное место размещения используемых программ указан корневой каталог диска C.

В ряде случаев прикладные программы используют каталоги с именем BIN. В этом случае именно его имя полезно указать в команде PATH.

Для пользователей, часто обращающихся к какой-либо одной системе программирования или прикладной системе (например, Turbo Pascal), может оказаться полезным включение в число альтернативных маршрутов имени соответствующего подкаталога.

2. Команда PROMPT задает формат приглашения DOS. Задание формата приглашения осуществляется параметром команды – строкой, состоящей из обычных текстовых и специальных управляющих символов. Управляющие символы снабжаются префиксом – знаком \$, отличающим их от обычных символов.

В рассматриваемом примере строка \$p\$g в качестве параметра команды PROMPT обеспечивает выдачу приглашения, ко-



торое используется наиболее часто. Оно содержит имя текущего каталога и имеет, например, такой вид:

C:\WORK>

Если бы строка имела вид \$t \$d \$b \$p\$g, то в приглашение были бы включены довольно длинные сообщения о времени и дате:

13:55:34.05 Mon 11 – 26 – 98 | C:\WORK)

Командой PROMPT можно выдавать и другие информационные параметры, например, имя пользователя, название организации и др.

3. Команда SET позволяет вводить в операционную обстановку системы различные имена с параметрами, которые затем могут использоваться прикладными программами. Типичное использование таких параметров – задание имен каталогов, где должны находиться все рабочие файлы для данной прикладной системы (например, базы данных или текстового редактора).

В рассматриваемом примере команда SET вводит имя ABC с параметром C:\ABC, которое является указанием текстовому процессору, откуда следует брать вспомогательные файлы (шрифты, драйверы и др.).

Рассмотренные выше команды PATH и PROMPT также кодифицируют обстановку, но в отношении фиксированных имен (а именно, устанавливают параметры для имен PATH и PROMPT). Команда SET позволяет делать то же самое для произвольных имен, и ее, таким образом, можно рассматривать как обобщение вышерассмотренных команд.

Команда SET может использоваться, например, для указания каталога, ориентированного на обслуживание конкретной прикладной программы.

4. Команда RKVGA.COM запускает русификатор шрифтов.

5. Команда VER в шестой строке файла AUTOEXEC.BAT является встроенной в DOS командой. Ее задача очень проста – выдать на экран сообщение о номере версии операционной системы.

6. Последняя команда в файле автозапуска – FKEYS.BAT – осуществляет вызов другого командного файла FKEYS.BAT, в котором происходит переопределение некоторых функциональ-

ных клавиш для предоставления пользователю возможности быстрого набора некоторых часто используемых команд.

Таким образом, в рассмотренном файле автозапуска указаны команды, с помощью которых создается определенная операционная обстановка, способствующая удобной дальнейшей работе пользователя. Часто в файл автозапуска вставляются также команды управления экраном, с тем, чтобы перед началом работы сделать цветные заставки, выдать сообщения и др. Применяется также команда MODE для установки параметров экрана, принтера и коммуникационного канала, а также команда ASSIGN для переназначения логических имен накопителей.

#### *7.2.5 Управление внешними устройствами*

Некоторые команды DOS позволяют управлять характером работы компьютера и периферийных устройств. К числу таких команд относится команда Mode. Команда Mode – многоцелевая команда, но все ее цели имеют с небольшими вариациями один смысл: изменение режима работы аппаратуры. Обычно эту команду применяют в файле автозапуска Autoexec.bat, чтобы реконфигурация работы периферийных устройств выполнялась автоматически каждый раз при загрузке операционной системы.

Команда Mode используется в следующих целях:

- установка режимов работы принтеров;
- установка режимов экрана монитора;
- установка режимов работы последовательного порта ПК;
- подготовка к работе последовательного принтера;
- установка или замена кодовой страницы (при работе на альтернативных языках);
- получение информации о текущих настройках режимов аппаратуры;
- установка частоты повторения ввода нажатой клавиши.

Для получения подробных сведений о методике применения команды Mode в конкретных случаях следует обратиться к справочной документации по DOS.

Управлять экраном монитора можно также с помощью драйвера ANSI.SYS – специальной программы, подключаемой к DOS через файл конфигурации CONFIG.SYS. Этот драйвер объемом менее 2 К обеспечивает дополнительные функции управле-

ния дисплеем: задание цвета символов и фона, позиционирование курсора, переопределение символов, вводимых с клавиатуры и т.д. Реализация этих функций осуществляется с помощью следующего приема – в драйвер посылаются особые управляющие последовательности символов (так называемые Esc-последовательности), которые и заставляют его выполнять те или иные операции.

### 7.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

### 7.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Файл конфигурации CONFIG.SYS.
2. Командные файлы.
3. Развитые командные файлы.
4. Файл автозапуска AUTOEXEC.BAT.
5. Управление внешними устройствами.

## **8. РАБОТА С ТЕКСТОВЫМ РЕДАКТОРОМ. РАБОТА С АРХИВАТОРОМ. РАБОТА С ОПЕРАЦИОННОЙ ОБОЛОЧКОЙ**

### **8.1 ЦЕЛЬ РАБОТЫ**

Цель работы – ознакомление с назначением архиватора и возможностями его настройки, изучение принципов работы с программами создания архивов.

### **8.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

#### *8.2.1 Технология работы:*

Программы архивирования данных позволяют за счет применения специальных методов “упаковки” информации сжимать информацию на дисках, т.е. создавать копии файлов меньшего размера, а также объединять копии нескольких файлов в один файл.

Для ознакомления с принципами и методами создания и обработки архивных файлов, в качестве примера, рассмотрим программу архиватор WinRAR. Для запуска программы

WinRAR необходимо найти на рабочем столе значоки запустить его.

#### *8.2.2 Интерфейс программы WinRAR*

Меню WinRAR содержит следующие пункты: "Файл", "Команды", "Операции", "Избранное", "Параметры" и "Справка". Щёлкните на интересующей вас ссылке, чтобы показать информацию о соответствующем меню.

Ещё один элемент интерфейса – панель инструментов. Она находится ниже меню и выше списка файлов. Кнопки на панели инструментов повторяют пункты из меню "Команды" (обратите внимание, что у всех пунктов в этом меню есть "горячие клавиши" для быстрого доступа). Во время просмотра содержимого архива некоторые кнопки могут быть отключены, если их функции неприменимы к архиву. При желании вы можете выбрать отображаемые кнопки, убрать текст с кнопок или уменьшить их

размер, вызвав диалог общих параметров программы или щёлкнув правой кнопкой мыши на панели инструментов.

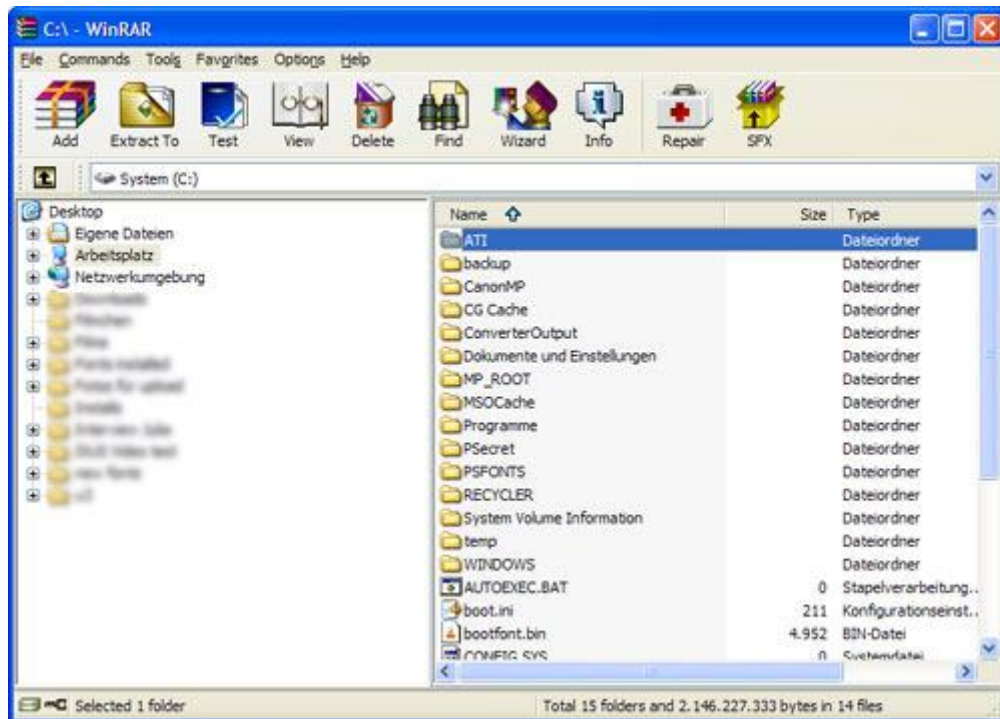


Рисунок 29 – Окно работы программы WinRAR

Под панелью инструментов находится маленькая кнопка со стрелкой вверх и строка списка дисков. При нажатии этой кнопки происходит переход в родительскую папку. Список дисков служит для выбора текущего диска или, скажем, сети. Этот список также можно открыть нажатием клавиши F4. При желании кнопку "Вверх" и список дисков можно перетащить в правый угол панели инструментов. Текущий диск также можно изменить нажатием сочетания клавиш Ctrl+D или щелчком мыши на маленьком значке диска в строке состояния.

Ниже панели инструментов расположено файловое окно. В нём отображается содержимое текущей папки или, если в WinRAR открыт архив, содержимое архива. Эти режимы называются режимом управления файлами и режимом управления архивами. Для каждого файла выводится следующая информация: имя, размер, тип и дата изменения. Для файлов в архиве показываются ещё два параметра — значение CRC32 и упакованный размер. CRC32 — это особая контрольная сумма, вычисляемая на

основании данных файла, с её помощью можно сразу определить, одинаковы ли упакованные в архиве файлы, не прибегая к их распаковке. Файлы с одинаковым содержимым всегда имеют одинаковые CRC32. Все параметры представлены в виде колонок. Порядок сортировки файлов можно поменять щелчком на заголовке колонки (там же синей стрелкой указывается направление сортировки). Кроме того, можно изменить ширину колонок, перетаскивая мышью разделители заголовков колонок. Несколько дополнительных параметров списка можно изменить в диалоге "Список файлов".

Если находящийся в архиве файл зашифрован, то после его имени будет стоять звёздочка ( \* ). Если файл продолжается в следующем томе, то после его имени будут стоять символы "--". Если файл продолжается из предыдущего тома, то после имени будут стоять символы "".

Перед обработкой файлов их необходимо выделить. Помимо стандартных способов выделения Windows в WinRAR для той же цели служат несколько дополнительных клавиш: , Insert, а также и на цифровой клавиатуре. Более подробно это описано в разделе "Выделение файлов".

Следующие комбинации клавиш можно использовать для навигации по списку файлов. Чтобы перейти в родительскую папку, нажмите клавиши Backspace, Ctrl+PgUp или дважды щёлкните мышью на папке ".." в списке файлов. Если вы сделаете это в корневой папке архива, то этим закроете архив и перейдёте в ту папку на диске, где он находится. Для перехода в другую папку можно нажать Enter, Ctrl+PgDn или дважды щёлкнуть левой кнопкой мыши на этой папке. То же действие на файле архива приведёт к открытию архива. Для перехода в корневую папку диска или архива служит комбинация клавиш Ctrl+\.

Если щёлкнуть правой кнопкой мыши на списке файлов, то появится меню с командами интерфейса и управления файлами. Эти команды доступны также из обычных меню WinRAR, с панели инструментов и с помощью сочетаний клавиш, поэтому вы можете использовать наиболее удобный для себя способ.

В левой части окна WinRAR может отображаться панель с деревом папок, если это включено в подменю "Дерево папок" в меню "Параметры". Дерево папок удобно использовать для быст-

рой навигации по папкам на диске и в архиве. Ширину панели с деревом можно изменять, перетаскивая мышью её правую границу.

Если включён параметр "Показывать комментарий" в диалоге общих параметров, а в открытом архиве есть комментарий, он будет показан в специальном окне справа от списка файлов. Ширину окна комментария можно изменять, перетаскивая мышью его левый край.

Внизу окна WinRAR (под списком файлов) находится строка состояния. В её левой части расположены два маленьких значка: "накопитель" и "ключ". Щелчком по значку "накопитель" можно изменить текущий диск, а щелчком по "ключу" — текущий пароль. Две соответствующие команды также есть в меню "Файл". По умолчанию значок "ключ" жёлтого цвета, но если введён пароль, то он становится красным. В средней части строки состояния выводится информация об общем размере выделенных файлов или о текущем состоянии. В правой части строки состояния отображаются общее количество файлов в текущей папке и их размер.

### 8.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

### 8.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение архивного файла?
2. Опишите технологию создания архивного файла с помощью WinRAR?
3. Опишите извлечение файлов в режиме графической оболочки WinRAR?
4. Опишите извлечение файлов в режиме командной строки?
5. В чем разница и преимущества форматов RAR и ZIP?

6. Какое действие выполняется при нажатии комбинации Alt+L.?

7. В чем смысл команды "Добавить информацию для восстановления"?

8. В каких случаях используется Команда "Восстановить архив" и её возможности?

9. Профили архивации – в чем заключается смысл данных настроек?

10. Объясните смысл создания самораспаковывающихся файлов?

11. Какие типы лицензий на использование WinRAR предусмотрены? Опишите кратко каждую.

12. Опишите процедуру лицензирования программы?



## **9. УСТАНОВКА И НАСТРОЙКА СИСТЕМЫ. УСТАНОВКА ПАРАМЕТРОВ АВТОМАТИЧЕСКОГО ОБНОВЛЕНИЯ СИСТЕМЫ. УСТАНОВКА НОВЫХ УСТРОЙСТВ. УПРАВЛЕНИЕ ДИСКОВЫМИ РЕСУРСАМИ**

### **9.1 ЦЕЛЬ РАБОТЫ**

Цель работы – изучить порядок установки и удаления устройств в операционной системе Windows.

### **9.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ**

#### *9.2.1 Базовые сведения*

Plug And Play – "вставь и играйся". Обозначает технологию, которая сводит к минимуму усилия по подключению новой аппаратуры. PnP-карты не имеют переключателей конфигурации или особых программ настройки; вместо этого общий для компьютера PnP-диспетчер (отдельная программа либо часть BIOS или ОС) сам находит каждую из них и настраивает на соответствующие адреса, линии IRQ, DMA, области памяти, предотвращая совпадения и конфликты.

PnP Manager записывает параметры конфигурации в ESCD (Extended System Configuration Data – данные расширенной системной конфигурации). Внешний PnP Manager использует для данных файл на диске, а PnP BIOS – собственное Flash-ПЗУ. Если в процессе конфигурации PnP-устройств обнаружены изменения – выдается сообщение "Updating ESCD..." и делается попытка записать изменения в ПЗУ. В случае успеха выдается сообщение "Success", отсутствие которого означает невозможность перепрограммирования Flash-ПЗУ (не установлена переключатель, стоит ПЗУ обычного типа или неисправны цепи программирования Flash-ПЗУ на системной плате).

#### *9.2.2 Общий порядок установки и удаления периферийного оборудования*

Перед началом установки периферийного оборудования или плат необходимо выключить и обесточить компьютер. При необходимости отвинчивается и снимается защитный кожух систем-

ного блока. Выбранное устройство устанавливается в слот расширения, соответствующий интерфейсу устройства, или (если устройство внешнее), подключается к порту, соответствующему интерфейсу устройства. После этого компьютер включается и выполняется процесс установки драйверов. Если устройства поддерживают интерфейс Plug&Play, программа установки драйверов инициализируется во время запуска операционной системы. В противном случае запускается диалоговое окно Установка оборудования. После установки драйверов нового оборудования возможно возникнет необходимость перезагрузки компьютера, о чем операционная система предупредит при выходе из программы установки драйверов.

Удаление драйверов устройств производится вручную. После удаления драйвера устройства необходимо выключить системный блок, обесточить его, после чего выбранное устройство извлекается из слота, или (если устройство внешнее), отключается от порта.

В операционной системе Windows упрощены процессы установки и использования новых устройств благодаря поддержке новейших технологических стандартов современного оборудования. Пользователь может больше внимания уделять своей основной работе и затрачивать меньше усилий на настройку и устранение неполадок при установке нового оборудования. Функции Windows, упрощающие установку устройств:

Самонастройка устройств (Plug and Play), имеющаяся в операционной системе Windows, используется для поиска нового оборудования и установки соответствующего драйвера.

С помощью функции автозапуска Windows определяется вид содержимого, хранящегося на новых устройствах, и тип носителя. В зависимости от вида содержимого — графика, музыка или видео — функция автозапуска открывает программу, предназначенную для работы с соответствующим типом файлов.

Поддержка большего числа устройств и технологий в Windows позволяет:

Использовать расширенные клавиатуры и USB-устройства ручного ввода, имеющие дополнительные клавиши для функций работы с мультимедиа, просмотра веб-страниц, управления электропитанием и других функций;

- применять беспроводные сетевые устройства, избавляющие от необходимости создавать сети сложной конфигурации;
- работать с многофункциональными устройствами, объединяющими в себе возможности, например, сканера, факса и принтера.

Кроме того, в операционной системе Windows поддерживаются широко известные, надежные стандарты оборудования, такие как IrDA (Infrared Data Association), USB (Universal Serial Bus) и высокоскоростная шина IEEE 1394.

В отличие от Windows позволяют существенно сократить количество ситуаций, в которых после внесения конфигурационных изменений требуется выполнять перезагрузку компьютера. Ниже перечислены ситуации, в которых при работе с прежними версиями требовалась перезагрузка компьютера, а новая версия позволяет обойтись без перезагрузки: изменение сетевой конфигурации, например, модификация IP-адресов, добавление или удаление сетевых протоколов, добавление или удаление периферийных устройств, включая аудио- и видеоадаптеры и их драйверы, добавление, удаление и переконфигурирование многих периферийных устройств, включая жесткие диски, сетевые адаптеры и др. устройства.

### *9.2.3 Установка и конфигурирование аппаратных средств.*

Все задачи, связанные с настройкой аппаратных средств, выполняются полностью автоматизированы. В случае, когда вы используете нестандартное устройство, которое система самостоятельно распознать не может функционал «Диспетчера устройств». Для этой цели раскройте «Параметры» -> «Система» -> «О системе» -> «Диспетчер устройств». Устройство, для которого по каким-то причинам драйвер систем самостоятельно установить не удалось, обычно помечается специальным значком желтого цвета с восклицательным знаком. Далее необходимо выбрать помеченное устройство и в его свойствах выбрать пункт по установке и обновлению драйверов данного устройства.

### 9.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

### 9.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Возможности Диспетчера устройств (Device Manager) по упрощению установки новых устройств.
2. Быстрый доступ к Параметрам.
3. Изменения в Параметрах (панели управления) по сравнению с предыдущими версиями Windows.
4. Способы решения задач по конфигурированию системы в Windows.

## 10. ИЗУЧЕНИЕ ЭМУЛЯТОРОВ ОПЕРАЦИОННЫХ СИСТЕМ. УСТАНОВКА ОПЕРАЦИОННОЙ СИСТЕМЫ

### 10.1 ЦЕЛЬ РАБОТЫ

Цель работы – научиться создавать виртуальный жесткий диск и виртуальную машину с помощью MSVirtual PC, VirtualBox.

### 10.2 ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

#### *10.2.1 Понятие эмуляции*

Эмуляция (англ. emulation) – это совокупность логических и технических средств и ресурсов, направленных на полную имитацию технического устройства выбранной пользователем системы для максимально точного воспроизведения всех процессов, происходящих внутри эмулируемой системы.

#### *История*

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

#### *Различие процесса эмуляции и симуляции*

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

Симуляция, в свою очередь, это попытка симитировать конкретную функцию или часть устройства, не повторяя в точности

ее процессы и работу ее отдельных компонентов для данной системы. Например, программа, имитирующая игровой автомат, которая выполняет родную для него программу ROM, будет являться эмулятором, а эта же программа, написанная для другой системы, но при этом не являющейся файлом ROM, который был предназначен только работы с той системой, под которую он был изначально сделан, называется симулятор.

### *Виды эмуляций*

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

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

сто этого команды центрального процессора также как и оригинальная программа.

### *10.2.2 Структура систем эмуляций*

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

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

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

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

Эмуляция подсистемы памяти чаще всего представляет собой как минимум две процедуры – для чтения из памяти и для записи в нее, отвечающие за координаты нахождения правильного объекта. Такой подход связан с особенностью совместимости ло-

гического адреса и физической памятью при осуществлении процесса эмуляции.

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

- управление прерываниями посредством процедуры, которая устанавливает флаги, считываемые эмулятором CPU, когда объявлено прерывание, что позволяет виртуальному CPU «опрашивать прерывания»

- запись и чтение физической памяти посредством двух процедур, подобных обслуживающим логическую память (однако, в отличие от последнего, первые часто могут быть заменены простыми ссылками на массив памяти).

### *10.2.3 Отличительные особенности процессов виртуализации от эмуляции*

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

Эмулятор – полностью или почти полностью реализует для исполнения кода отдельную машину со своей архитектурой и своими ресурсами. Вплоть до того, что может быть процессор совершенно другой архитектуры, например ARM, Power или калькулятора МК-52. Так же необходимо отметить, что при эмуляции



осуществляется изоляция процессов эмулируемой системы, для максимально точного исполнения поведения данной системы в пределах хост-машины и для защиты корректного поведения систем этой машины

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

### 10.3 ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Данное занятие предполагает выполнение следующих этапов:

- изучить методические указания;
- выполнить выданное задание;
- ответить на контрольные вопросы.

Образец задания:

Работа с командами в среде MS Virtual PC и Virtual Box по установке и настройке различных ОС

Установить главный контроллер домена

Подсоединить компьютер к домену

Установить соединение с доменом в качестве пользователя

Изменить политику безопасности

Установка ОС Ubuntu на виртуальную машину.

При загрузке виртуальной машины появится первичный диалог, в нем выберите язык программы инсталляции и нажмите «Enter».

В следующем меню выберите «Установить Ubuntu».

Установка проходит в 6 шагов:

- выбор основного языка ОС;
- выбор местоположения;
- выбор раскладки клавиатуры;
- подготовка жесткого диска;
- указание параметров пользователя и компьютера, логина и пароля (указывать необходимо);

- проверка параметров установки.

При прохождении первых четырех шагов, параметры, указанные в них можно оставлять без изменений.

После окончания установки перезагрузите виртуальную машину.

#### 10.4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что понимается под процессом эмуляции?
2. Различие процесса эмуляции и симуляции.
3. Характеристика видов эмуляции.
4. Характеристика структуры систем эмуляций.
5. Отличительные особенности процессов виртуализации от эмуляции.

## **11. МЕТОДИЧЕСКИЕ УКАЗАНИЯ К САМОСТОЯТЕЛЬНОЙ РАБОТЕ ОБУЧАЮЩЕГОСЯ**

Самостоятельная работа обучающегося состоит из изучения теоретического материала, согласно рабочей программе дисциплины, подготовки к практическим и лабораторным занятиям.

На самостоятельное изучение вынесены следующие темы дисциплины:

1. Выполнение заданий по настройке интерфейса пользователя.
2. Подготовка сообщений по теме "Архитектура операционной системы".
3. Исследование состояния процессов.
4. Исследование способов распределения пространства дисковой памяти.

Проработка учебников, изучение конспекта лекций, ознакомление с методическими указаниями по выполнению практических и лабораторных работ позволит студенту всесторонне изучить разбираемую на занятиях тему. Это поможет студенту не только качественно выполнить практические работы, но и успешно пройти текущую и промежуточную аттестации.

## 12. СПИСОК ЛИТЕРАТУРЫ

1. Батаев, А. В. Операционные системы и среды : учебник для образовательных учреждений среднего профессионального образования по укрупненной группе специальностей "Информатика и вычислительная техника" / А. В. Батаев, Н. Ю. Налютин, С. В. Сеницын. – Москва : Академия, 2021. – 285 с. – URL: <https://academia-moscow.ru/reader/?id=539321> (дата обращения: 02.04.2024). – Текст : электронный.
2. Гостев, И. М. Операционные системы: учебник и практикум для СПО / Гостев И. М.. – Москва : Юрайт, 2020. – 164 с. – URL: <https://urait.ru/book/operacionnyye-sistemy-453469> (дата обращения: 31.03.2024). – Текст : электронный.
3. Партыка, Т. Л. Операционные системы, среды и оболочки : Учебное пособие / Т. Л. Партыка, И. И. Попов. – Москва : НИЦ ИНФРА-М, 2021. – 560 с. – URL: <https://znanium.com/catalog/document?id=364475> (дата обращения: 11.10.2023). – Текст : электронный.