Министерство образования и науки Российской Федерации
Федеральное агентство по образованию
Государственное образовательное учреждение высшего
профессионального образования

РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Шуранов Е.В. Петров Г.А. Левин И.И.

Руководство к выполнению лабораторных работ по дисциплине «Сетевое администрирование»

РГГМУ

Санкт-Петербург
2009

УДК 004.65

Шуранов Е. В., Петров Г. А., Левин И.И. Руководство к выполнению лабораторных работ по дисциплине «Сетевое администрирование» - СПб.: РГГМУ, 2009.- 36 с.

Руководство к выполнению лабораторных работ по дисциплине «Сетевое администрирование» включает шесть лабораторных работ, входящих в программу этой дисциплины. Лабораторные работы ориентированы на приобретение студентами практических навыков в работе с системами контроля версий и освоение принципов командной работы над проектом.

Содержащиеся в руководстве сведения по теории управления проектами создания программного обеспечения, и рекомендации по выполнению лабораторных работ позволяют использовать их в качестве дополнительного учебного пособия.

Руководство предназначено для студентов, обучающихся по специальности «Информационная безопасность телекоммуникационных систем» и «Морские информационные системы и оборудование»

Рецензент Матвеев С. И., канд. техн. наук, сотр. филиала корпорации ЛГ электрикс инк.

ВВЕДЕНИЕ

В наши дни не редко возникает ситуация, когда специалисты получившие образование в области компьютерных наук, успешно справляются с задачами по программированию, но оказываются абсолютно не приспособленными быстро влиться в рабочий процесс на месте работы. Одна из причин этого заключается в необходимости освоения многочисленных вспомогательных инструментов используемых программистами для оптимизации рабочего процесса. Основная цель данных лабораторных работ – привить студентам базовые навыки работы с некоторыми из этих инструментов.

Лабораторные работы рассчитаны на студентов владеющих навыками программирования. Изначально курс ориентирован на студентов специальности информационная безопасность, поэтому в качестве типовых программ (приложение 1) предлагается реализация криптографических алгоритмов, изученных студентами ранее. В случае адаптации курса под другие специальности рекомендуется заменить типовые программы приложения 1 на другие, не вызывающие у студентов сложностей с реализацией.

В теоретической части лабораторных работ встречается текст, *выделенный курсивом*, это полезная информация, которая не используется в предлагаемых работах, но может пригодиться при работе с реальными проектами.

ЛАБОРАТОРНАЯ РАБОТА № 1

Изучение базовых консольных команд ОС Windows.

Цель работы

Приобрести практические навыки в работе с командами переадресации, конвейерами и командами фильтров: **FIND**, **MORE**, **SORT**.

Краткие теоретические сведения

Консольные команды ОС Windows – команды, обрабатываемые приложениями COMMAND.COM или CMD.EXE, встроен-

ными в ОС Windows. Ввод таких команд можно осуществить выбрав меню “Пуск”» (“Start”) -> “Выполнить…” (“Run…”). В появившемся окне набрать “cmd” и нажать кнопку “Ok”. На экране появится консольное окно (рис. 1), в котором вводятся команды.

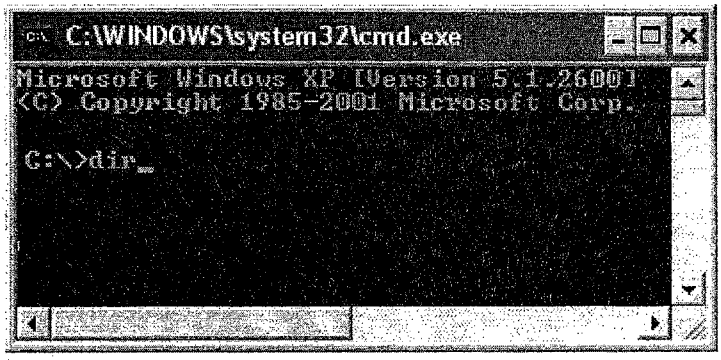

Рис. 1. Пример ввода команды dir.

По умолчанию результат выполнения команды выводится на экран. Использование символов переадресации потоков ввода-вывода позволяет переназначить источник или получатель информации. Для переадресации используют символы:

- ">" – переадресовать поток вывода. Для записи результатов выполнения команды в файл после символа “>” должно следовать полное имя файла. При применении данной команды запись результатов осуществляется в новый файл. Если файл с введенным именем уже существовал на диске, все его содержимое будет потеряно. При работе с файлами возможн ввод только имени файла (без указания пути), в этом случае файл создается в текущем каталоге. Кроме того, может быть указано имя порта ввода-вывода для передачи данных внешнему устройству. Например, переадресация вида “> COM1” позволяет перенаправить поток вывода устройству, подключенному к последовательному порту компьютера.
- ">>" – результат аналогичен предыдущему, но для существующего файла, новая информация записывается в конец файла. Таким образом, файл будет не перезаписан, а дополнен новыми данными.
- "<" – переадресовать поток ввода. Данная команда используется в случае, если входные данные не вводятся пользователем с клавиа-

туры, а поступают от внешнего периферийного устройства или записаны в файл.

Например, перечень содержимого каталога C:\temp в файл с именем DIRFILE.TXT можно осуществить следующей командой:

C:\>dir c:\temp>DIRFILE.TXT

Текстовый файл создается в текущем каталоге, в данном примере на диске C.

Команда поиска указанных данных **FIND** имеет формат:

FIND [/C][/N][/V] "text"[[PATH]FILENAME]...[...], где

- **"text"** – искомая символьная строка;
- **[/C][/N][/V]** спецификаторы команды:

/C – служит для вывода только количества вхождений искомой строки. Содержимое строкфайла, содержащее искомую комбинацию символов, не выводится;

/N – выводится количество вхождений строки "text" в файл с указанием номеров строк в файле;

/V – исключающий поиск, т.е. вывод строк, не содержащих образец "text";

/I – для игнорирования регистра искомой строки.

- **[[PATH]FILENAME]** – полное имя файла в котором осуществляется поиск.

Команда может применяться только для текстовых файлов.

Например, если необходимо найти в текстовом файле EXAMPLE.TXT строки, содержащие сообщение “FILENAME”, достаточно ввести команду

C:\>FIND "FILENAME" D:\EXAMPLE.TXT

В примере каталогом по умолчанию (из которого производится запуск команды) является корневой каталог диска **C:**. Поиск строки символов осуществляется в файле, расположенном в корневом каталоге диска **D:**. Результат выполнения команды выводится на экран.

Конвейеры позволяют организовать последовательность команд с передачей выходных данных от предыдущей команды в качестве входных параметров к последующей и т.д. Символ конвейера "|".

При использовании команды **FIND** можно организовать фильтр в конвейере. Например, для поиска на диске D:\ файлов, в имени которых содержится "DISK" нужно выполнить команду:

C:\>DIR D: | FIND "DISK"

При этом результат работы будет выведен на экран в виде списка имен файлов и каталогов, содержащих в имени искомую комбинацию символов.

Некоторые команды предусматривают возможность постраничного вывода на экран. Например, команда **DIR/P** позволяет постранично вывести содержимое текущего каталога. Другие команды для этого используют команду **MORE**.

Например, можно записать команду постраничного вывода текстового файла:

C:\>TYPE D:\MYFILE.TXT | MORE

Если вывод содержимого файла на один экран невозможен, то он будет осуществляться постранично, внизу экрана появиться сообщение:

----------MORE----------

после чего для продолжения вывода достаточно нажать любую клавишу.

При необходимости сортировки в текстовых файлах по алфавиту или числам в любой колонке текстовых строк можно воспользоваться командой **SORT**.

Формат команды:

SORT [/R] [/+N] [[drive:][path]filename] [/O [drive:][path]filename]

где **/R** – задает реверсированную сортировку (по убыванию);

/O – позволяет задать имя получателя (файл, консоль, т.д.);

/+N – указывает номер колонки в строке, с которой начинается сортировка.

Например, файл OLD.TXT необходимо отсортировать по возрастанию и по алфавиту и записать под именем NEW.TXT, для этого можно использовать команду:

D:\>SORT < D:\OLD.TXT > D:\NEW.TXT

Команда сортировки может использоваться в конвейере. Например, при просмотре каталога с упорядочением файлов по возрастанию кода первой буквы:

D:\>DIR D: | SORT

Полный перечень команд приводится в приложении 2.

Задание

1. Открыть консольное окно.

2. Создать в своей папке текстовый файл, используя команды просмотра каталогов и символов переадресации. Файл должен содердать не менее 40 строк.
3. С помощью команд **TYPE** и **FIND** просмотреть созданный текстовый файл и найти строки, содержащие образцовый текст, например, расширение **COM** или **EXE**. Повторить команду, но с записью результата исполнения команды в файл в свою папку.
4. Проверить работу ключей **/N** и **/I** команды **FIND**. Объяснить полученные результаты.
5. Осуществить постраничный вывод текстового файла на экран, используя команду **MORE**, символы конвейера.
6. Выполнить поиск по образцу в созданном в п.1.2 текстовом файле с выводом информации постранично на экран. В качестве образцового текста использовать результат выполнения команды **DIR**.
7. Осуществить сортировку текстового файла с выводом информации на экран и в новый текстовый файл.
8. Проверить работу ключа **/R** команды **SORT**. Объяснить полученный результат.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Объяснить отличие между символами переадресации ">" и ">>".
2. Как осуществить реверсивную сортировку по пятой позиции текста в строке?
3. Приведите пример совместного использования команд MORE и SORT.
4. Возможно ли совместное использование команд FIND и SORT?
5. Как осуществить сортировку каталога по алфавиту с созданием файла?
6. Можно ли изменить с помощью изученных команд каталог дискеты, чтобы, например, при ее просмотре он был отсортирован по алфавиту.

Требования к содержанию отчета

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen) с краткими пояснениями.

ЛАБОРАТОРНАЯ РАБОТА № 2

Файлы пакетной обработки.

Цель работы

Приобрести практические навыки работы с консольными командами ОС Windows. Получить навыки создания и применения файлов пакетной обработки.

Краткие теоретические сведения

Пакетный файл (англ. batch file) — текстовый файл в MS-DOS, OS/2 или Windows, содержащий последовательность команд, предназначенных для исполнения командным интерпретатором. После запуска пакетного файла программа-интерпретатор (как правило COMMAND.COM или CMD.EXE) читает его строка за строкой и последовательно исполняет команды. Пакетный файл это аналог *shell script* в *Unix*-подобных операционных системах.

Пакетные файлы полезны для автоматического запуска приложений. Основная область применения — автоматизация рутинных операций, которые регулярно приходится совершать пользователю компьютера. Примерами таких операций могут служить: обработка текстовых файлов, копирование, перемещение, переименование, удаление файлов; работа с папками; архивация; создание резервных копий баз данных и т. п. Пакетные файлы поддерживают операторы **if** и **goto** (а в системах семейства Windows NT и расширенный оператор **for**), что позволяет обрабатывать результаты выполнения предыдущих команд или приложений и в зависимости от этого выполнять дальше тот или иной блок команд (как правило, в случае удачного завершения приложение возвращает 0 в переменной errorlevel, в случае неудачного — 1 или большее значение).

Пакетные файлы в DOS имеют расширение .bat; для других операционных систем они могут иметь другие расширения. Например, .CMD в Windows NT и OS/2, или .BTM в 4DOS или подобных оболочках.

Пакетные файлы могут содержать как внутренние команды, обрабатываемые непосредственно COMMAND.COM или CMD.EXE, так и обращения к внешним утилитам, существующим в виде от-

дельных программ (.EXE файлов). Данные программы значительно расширяют возможности пакетных файлов.

Важным свойством командных файлов является возможность использовать внутри них формальные параметры. При этом обращение к командному файлу приобретает вид:

C:\> имя_командного_файла параметр1 [параметр2...]

Параметры, значения которых будут заданы при обращении к командному файлу, внутри файла будут иметь ния **%1,%2,...,%N** (где N<10).

Рассмотрим операторы, которые используются только в командных файлах.

Оператор **ECHO** позволяет управлять потоком вывода на дисплей:

ECHO OFF блокирует выдачу на экран приглашений MS-DOS и текстов команд.

ECHO ON отменяет действие **ECHO OFF**.

ECHO <TEXT> позволяет вывести текст при заблокированной выдаче.

Оператор **REM** позволяет ввести комментарий в текст командного файла. Строка после **REM** не анализируется командным процессором.

Оператор **PAUSE** приостанавливает дальнейшую обработку пакетного файла до нажатия пользователем любой клавиши. При этом на экране появляется сообщение "Strike any key when ready".

Оператор **GOTO** позволяет передавать управление на метку и может использоваться самостоятельно либо совместно с оператором проверки условия **IF**. Метка в пакетных файлах занимает отдельную строку и отличается тем, что её первый символ - двоеточие (:).

Оператор **IF** позволяет проверить условие и выполнить команду в зависимости от результата его проверки. В качестве условия может выступать:

1. Проверка кода завершения программы, срабатывающей перед оператором **IF** (сформированного специальным прерыванием MS-DOS):

IF ERRORLEVEL <N> <команда MS-DOS>

Условие считается выполненным, если выработанный код завершения больше или равен N.

2. Проверка наличия файла:

IF EXIST <имя или шаблон файла> <команда MS-DOS>

Условие считается выполненным при обнаружении файла.

3. Сравнение двух строк, которые могут быть заданы и через формальные параметры:

IF %<N>==<текстовая строка> <команда MS-DOS>

При абсолютном совпадении двух строк условие считается выполненным.

Любое из этих условий может задаваться со знаком логического отрицания **NOT**.

Оператор **FOR** обеспечивает циклическое выполнение команд MS-DOS. При этом можно задать формальный параметр и список фактических параметров (обычно имён файлов), которые последовательно подставляются в текст исполняемой команды вместо формального параметра.

Формат команды:

FOR <формальный параметр> IN (<список фактических параметров>) DO <команда MS-DOS>

Например, команда

FOR %%A IN (PAS OBJ EXE) DO COPY PROG.%%A D:

вызывает копирование на диск D: трёх файлов с именем PROG и расширениями PAS, OBJ,EXE.

Оператор **SHIFT** вызывает сдвиг формальных параметров относительно фактических. Так, после него первым (%1) формальным параметром становится фактический второй и т.д. Команда SHIFT может применяться для последовательной обработки заранее не определённого количества параметров либо для обработки более 9 фактических параметров (так как максимальный номер формального параметра %9).

Оператор **COMMAND** позволяет вызывать новую копию командного процессора. При помощи этого оператора возможно выполнение рекурсивных вызовов командных файлов с возвратом. Так, команда **COMMAND/C V** вызовет обработку командного файла V.BAT до тех пор, пока в нем не закончатся операторы или не встретится оператор **EXIT**.

По этому оператору продолжится обработка вызывающего командного файла.

Пример batch файла, для автоматического запуска программ.

Файл autoexec.bat представляет собой командный файл, обращение к которому осуществляется автоматически при начальной загрузке операционной системы. При этом сам файл должен находиться в корневом каталоге системного диска. Обработка его происходит вслед за обработкой содержимого файла config.sys.

Основное назначение файла autoexec.bat состоит в формировании удобной индивидуальной операционной обстановки. Для этого могут быть использованы команды установки пути поиска (path), приглашения MS-DOS (prompt), загружены определённые программы и т.д. Вообще в файле autoexec.bat могут быть использованы любые команды MS-DOS и вызваны любые программы пользователя.

Как и любой пакетный файл, autoexec.bat обрабатывается командным процессором последовательно, команда за командой. Выполнение его можно прервать при помощи комбинации клавиш Ctrl-C. После выполнения очередной команды на экране появится надпись "Terminate batch job ? (y/n)", и после нажатия на клавишу Y выполнение файла прекращается.

Рассмотрим, как может выглядеть файл autoexec.bat.

```
echo off
dosedit
path b:;c:\;c:\work
prompt $p$g
ver
echo Система загружена
pause
date
time
cls
```

Первая строка определяет, что текст всех команд командного файла не будет выводиться на дисплей. Вторая строка загружает программу, запоминающую и позволяющую вызвать и редактировать все команды пользователя на протяжении сеанса работы. Третья строка указывает, что файлы будут отыскиваться после поиска в текущем каталоге в установленном каталоге диска B:, корневом каталоге диска C: и подкаталоге WORK диска C:. Четвёртая строка определяет, что приглашение MS-DOS будет содержать имя теку-

щего каталога. Пятая строка вызовет указание номера версии MS-DOS. Шестая строка выводит сообщение "Система загружена". Седьмая строка вызывает появление на экране сообщения "Strike any key when ready...", в результате чего отработка следующей команды будет производиться лишь после нажатия любой клавиши. Команды **date** и **time** вызывают инициализацию и задание пользователем даты и времени. Команда **cls** в последней строке командного файла очищает экран дисплея.

```
rem echo off отключает режим отображения команд на экране.
@echo off
rem echo. печатает пустую строку.
echo.
echo Здравствуй, мир! Нажмите любую клавишу для запуска программы Program.exe!
pause > nul
rem Запуск программы с аргументом, переданным при запуске пакетного файла
Program.exe %1
rem Обработка возможной ошибки
if errorlevel 1 goto error
echo.
echo Программа завершила свою работу!
goto end
:error
echo.
echo Произошла ошибка при работе программы
:end
```

Выбор команд, включаемых в файл autoexec.bat в общем случае определяется пользователем.

Задание

Вариант 1.

Исходные данные:

- наличие диска C:\
- наличие на диске C: каталога C:\ALPHA\BETTA\GAMMA

Создать командный файл BATCH1.BAT, выполняющий действия:

1. Создание на диске C: каталога DIR1 и в нем подкаталога DIR2.

2. Создание в каталоге C:\ALPHA\BETTA\GAMMA файла TEXT1.TXT и запись в него строки "Hello, world!".
3. Копирование файла TEXT1.TXT из каталога C:\ALPHA\BETTA\GAMMA в каталог C:\DIR1\DIR2 под именем TEXTNEW.TXT.
4. Удаление исходного файла.

В создаваемом командном файле должно быть предусмотрено:
1. Отключение режима отображения на экране выполняемой команды.
2. Вывод на экран: "Копирование и удаление файла".
3. Вывод на экран: "Файл скопирован и удален".
5. Пауза по окончании работы до нажатия пользователем любой клавиши.

Вариант 2

Исходные данные:
- наличие диска C:\
- наличие на диске C: каталогов C:\D1 и C:\D2
- наличие в каталоге C:\D1 файлов ANEW.PAS и BNEW.PAS с любым текстом внутри.

Создать командный файл с именем SUMMA.BAT, выполняющий действия:
1. Вывод на экран "Объединение и переименование файлов".
2. Объединение содержимого файлов ANEW.PAS и BNEW.PAS, находящихся в каталоге C:\D1, в файл CNEW.PAS в каталоге C:\D2.
3. Вывод содержимого файла CNEW.PAS на экран.
4. Ожидание нажатия клавиши.
5. Переименование файлов ANEW.PAS и BNEW.PAS в AOLD.PAS и BOLD.PAS соответственно.
6. Вывод на экран: "Задание выполнено".

Вариант 3

Исходные данные:
- наличие диска C:\,
- наличие на диске C: каталога C:\FOR и файла SIMP.FOR в этом каталоге.

Создать командный файл с именем _EXIST.BAT, выполняющий действия:
1. Вывод на экран "Копирование файла в случае его отсутствия на диске".

2. В случае отсутствия на диске C:\ файла SIMP.FOR скопировать его туда из каталога C:\FOR и вывести на экран: "Файл simp.for скопирован на диск C:\".
3. Если файл SIMP.FOR уже есть на диске, вывести на экран: "Файл simp.for уже есть на диске".

Вариант 4

Создать командный файл с именем BATCH4.BAT, выполняющий следующие действия в зависимости от переданного параметра в строке вызова:
1. Создание каталога C:\MYDIR и копирование в него всех .com и .exe файлов из каталога указаного в качестве параметра.
2. Создание в каталоге C:\MYDIR каталога \NEWDIR и копирование в него всех .com файлов из каталога, указанного в качестве параметра.
3. Вывод на экран содержимого каталога C:\MYDIR.
4. Удаление каталога C:\MYDIR\NEWDIR.
5. Удаление каталога C:\MYDIR.

Вариант 5

Создать командный файл, , выполняющий следующие действия:
1. Установить переменные окружения: ENV1, равное пути вашей рабочей папки, ENV2, содержащей пути переменной PATH и переменной ENV1 в стандартном формате переменной PATH.
2. В папке, указанной в переменной окружения ENV1, удалить файлы с заданными расширениями (*.OBJ, *.EXE). После создания файла запустить его на выполнение.

Требования к содержанию отчета

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen), с краткими пояснениями.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. При сравнении текстовых констант отличается ли прописная буква от строчной?
2. Как осуществить рекурсивный вызов командных файлов?
3. Как и зачем используется команда If?
4. Как и зачем используется команда Goto?
5. Как и зачем используется команда For?

6. Зачем используется команда pause?

ЛАБОРАТОРНАЯ РАБОТА№3

Система версионного контроля. Базовые понятия.

Цель работы

Приобрести практические навыки по внедрению системы версионного контроля. Получить навык создания нового проекта под системой версионного контроля, настройки клиентской части. Освоить основные опирации по работе с системой версионного контроля SVN.

Краткие теоретические сведения

Управление версиями это искусство работы с изменениями информации. Долгое время оно было жизненно важным инструментом программистов, которым необходимо произвести небольшие изменения в программе, или же сделать "откат" изменений, возвращаясь к предыдущей версии. Однако полезность систем управления версиями выходит далеко за пределы мира разработчиков программного обеспечения. Управление версиями требуется повсюду, где можно встретить людей, использующих компьютер для работы с постоянно изменяющейся информацией.

Subversion (SVN) это свободная система управления версиями с открытым исходным кодом. Subversion позволяет управлять файлами и каталогами во времени. Дерево файлов помещается в центральное **хранилище**, которое похоже на обычный сервер файлов с тем отличием, что запоминает каждое изменение, внесённое в файл или каталог. Это позволяет восстановить ранние версии данных, исследовать историю изменений данных.

Общий взгляд на устройство Subversion показан на рис. 2.

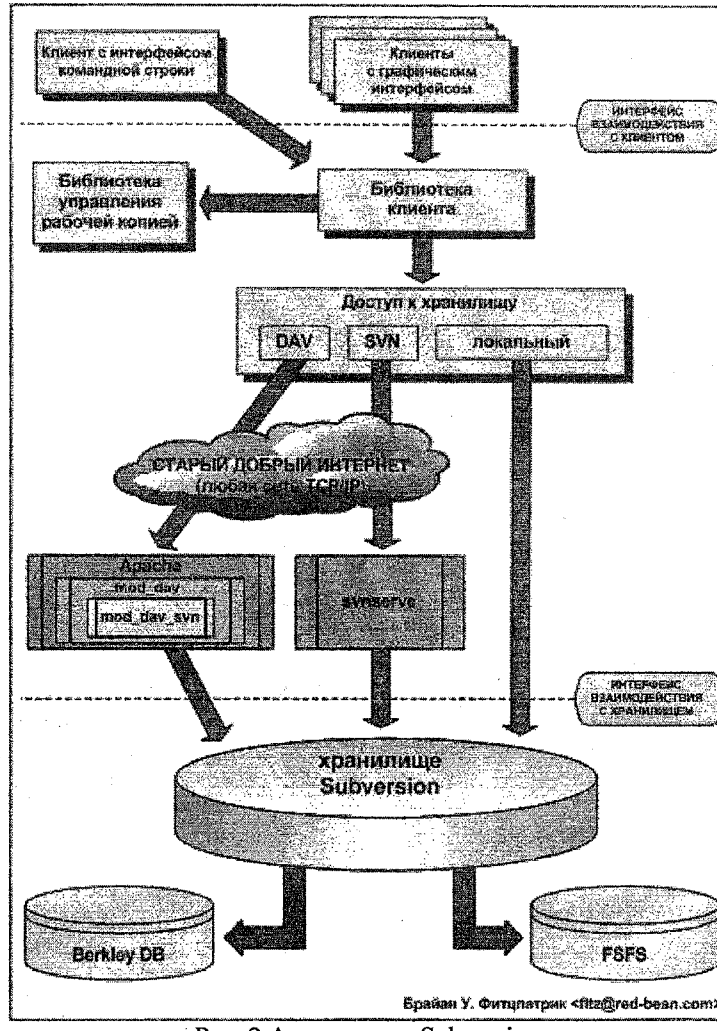

Рис. 2 Архитектура Subversion.

В нижней части схемы изображено хранилище Subversion, в котором хранится информация с версиями. В верхней части схемы показана программа-клиент Subversion, которая управляет локальными отражениями различных фрагментов этих данных (также называемыми «рабочими копиями»). Между этими сторонами проложены различные маршруты, проходящие через разные слои доступа

к хранилищу. Некоторые из этих маршрутов используют компьютерные сети и сетевые сервера, чтобы достичь хранилища, в то время как другие маршруты в сети не нуждаются и ведут к хранилищу напрямую.

Установка Subversion.

Для установки системы версионного контроля SVN необходимо скачать устоновочные файлы, находящиеся в свободном доступе на сайте изготовителя:

http://subversion.tigris.org/.

Для работы необходимо установить SVN запустив:

svn-1.4.3-setup.exe [3].

Для работы установки Tartoise запустить:

TortoiseSVN-1.4.4.9706-win32-svn-1.4.4.msi [4] Tartoise – утилита, обеспечивающая удобный визуальный интерфейс в ОС windows для эфективной работы с SVN.

Полезным будет скачть в интернете руководство- **svnbook** [2].(http://svnbook.red-bean.com/nightly/ru/index.html)

Команды SVN обеспечивают пользователя всем необходимым набором инструментов. Тем не менее использование Tortoise значительно упрощает выполнение основных манипуляций в системе версионного контроля.

На рис. 3 и 4 представлены окна выбора каталогов и параметров установки, появляющиеся в процессе установки SVN и TortoiseSVN соответственно.

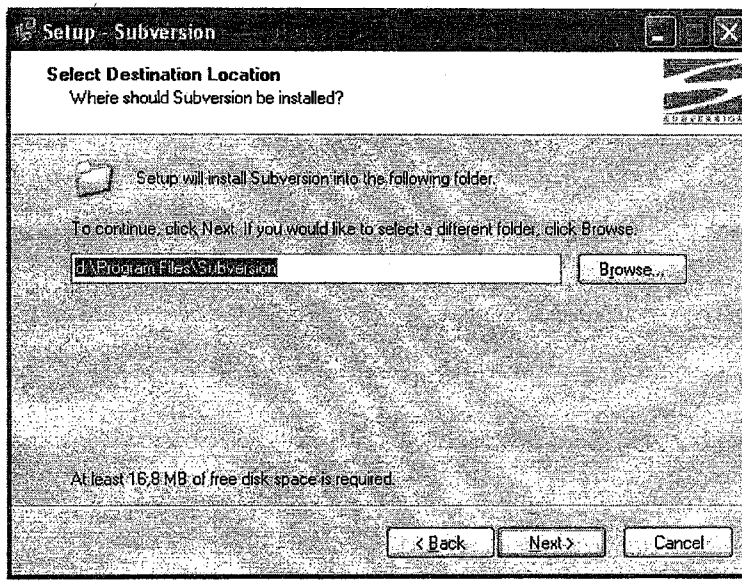

Рис. 3 Установка SVN

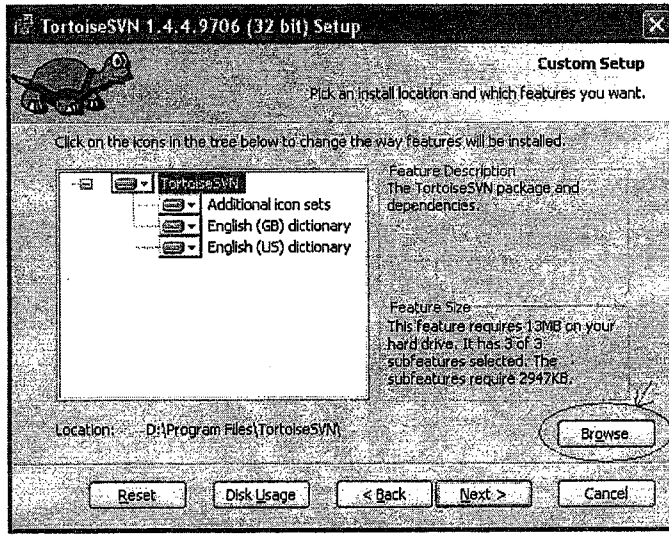

Рис. 4 Установка TortoiseSVN

Установленная Subversion имеет определенное количество компонентов.

- Svn - Клиент командной строки.
- Svnversion – Программа, показывающая состояние (в пределах ревизий существующих элементов) рабочей копии.
- Svnlook - Инстумент для контроля Subversion хранилища.
- Svnadmin - Инструмент для создания, настройки или востановления Subversion хранилища.
- Svndumpfilter - Программа для фильтрации дамповых потоков Subversion хранилища.
- mod_dav_svn - Подключаемый модуль для HTTP сервера Apache, использующийся для предоставления сетевого доступа к вашему хранилищу.
- Svnserve - Собственный отдельный сервер, запускается как процесс-демон и доступен посредством SSH; это еще один способ для предоставления сетевого доступа к хранилищу.

Для выполнения лабораторных работ, из перечисленных компонентов, будет активно использоваться только компонент svn и изредка svnadmin и svnversion. Для подробного ознакомления с перечисленными компонентами рекомендется использовать руководство по Subversion - **svnbook** [2] о котором говорилось рание, так как в данном руководстве будет говорится только о тех командах, которые могут пригодиться при выполнении лабораторных работ.

После установки SVN в командной строке станет доступен набор команд SVN, список которых можно получить с помощью команды **svn help**.

После установки TortoiseSVN в окне проводника windows при нажатии правой кнопкой мыши на любом каталоге или файле станут доступны пункты меню, выполняющие управление SVN. Файлы, внесённые под SVN, станут отмечаться в нижнем левом углу специальными значками в зависимости от их статуса.

Версию SVN можно определить с помощью команды: **svn –version**

Для создания нового проекта неолбходимо создать хранилище называемое так же репозиторием.

Хранилище является разновидностью файл-сервера, однако не совсем обычного. Что делает хранилище Subversion особенным

— это то, что он запоминает каждое внесенное изменение: любое изменение любого файла, равно как изменения в самом дереве директорий такие как добавление, удаление и реорганизация файлов и директорий.

При чтении данных из хранилища клиент обычно видит только последнюю версию дерева файлов. Но клиент также имеет возможность просмотреть предыдущие состояния файловой системы. Например, клиент может запросить такие данные, как «Что содержала эта директория в прошлую среду?», или «Кто был последним изменявшим этот файл, и какие вносились изменения?» Вопросы подобного типа основные для любой системы контроля версий: системы, разработанной для записи и отслеживания изменений информации во времени.

Для создания репозитория используется команда
svnadmin create REPOS_PATH

Данная команда создаёт в текущем (или в указанном) каталоге новое хранилище.

Внимание, несмотря на сжатие данных в репозитории, размер данной папки может в несколько раз привысить размер проекта, так как хранит все изменения. Для размещения репозитория необходимо выбирать диск с достаточным объёмом, а в случае больших проектов следует размещать эту папку на специальном сервере.

Размещение репозетория на сервере является правильным решением ещё и потому, что это позволяет простыми способами организовать безопаснойть данных и бесперебойный доступ к нему всех участников проекта.

Для создания нового проекта на локальном компьютере пользователя создается пустая папка проекта. Далее необходимо указать на репозиторий с которым будет связана эта папка, нажав правой кнопкой мыши на папку и выбрав SVNCheckout. В папке можно создать новые файлы или скопировать их из других папок, чтобы внести их под SVN, т.е. чтобы в дальнейшем все изменения, сделанные с этими файлами, запоминались в созданом нами хранилище. Для внесения новых файлов под SVN необходимо по нажатию правой кнопки мыши выбрать команду TortoiseSVN->add. Новые добавляемые файлы помечаются синим плюсиком в нижнем

левом углу (рис. 5). Для завершения транзакции [1]добавления файлов используется команда SVN Commit.

При создании нового проекта, операцию SVNCheckout можно делать не для пустой папки, а для папки уже содержащей файлы проекта. Во всплывающем окне появится предупреждение о том, что данная папка не пуста. Необходимо нажать "Yes", после чего выбрать команду TortoiseSVN->add и т.д. по описанному выше алгоритму.

Рис. 5 Пример добавленных новых файлов

При редактировании файла, находящегося под версионным контролем, TortoiseSVN меняет зелёный флаг, в нижнем левом углу иконки, файла на красный (рис. 6).

[1] Транзакция (англ. transaction) — в информатике, группа последовательных операций, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще и тогда она не должна произвести никакого эффекта.

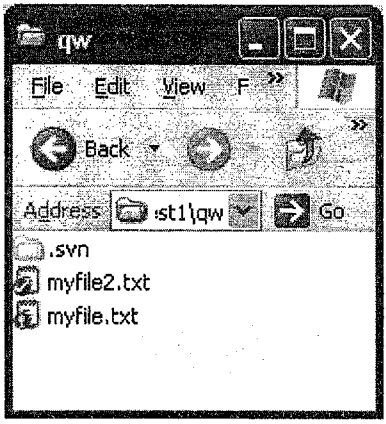

Рис. 6 Пример отображения файлов с изменениями (myfile.txt) и без изменений (myfile2.txt) .

Для внесения изменений под версионный контроль необходимо по нажатию правой мышки выбрать команду SVN Commit. В случае командной работы над проектом существует вероятность возникновения конфликтов, если один и тот же файл редактируется одновременно двумя или более участниками проекта. Для разрешения большинства подобных ситуаций помогает выполнение команды SVN Update перед выполнением SVN Commit. Команда SVN Update позволяет скачать с сервера последние изменения, сделанные другими пользователями, и автоматически объединить их с изменениями других участников проекта.

Если подобная операция невозможна в автоматическом режиме, вам будет предложено объединить конфликтующие участки вручную. Подобная операция "объединения" различных вариантов кода часто используется разработчиками программ. Для обозначения этой операции используется термин merge. Для удобства выполнения данной операции можно использовать встроенное в TortoiseSVN приложение, позволяющее сравнивать текстовые файлы. Подобные приложения, как правило, встроены в современные средства версионного контроля. Для этих целей так же широко используются коммерческие приложения такие как Araxis Merge, Beyond compare или свободнораспространяемый winmerge. В TortoiseSVN предусмотрена возможность замены встроеного приложе-

ния сравнения файлов на другое. Для этого используется пункт меню:
TortoiseSVN->Settings->External Programы->Diff Viewer.

Для просмотра всех изменений, сделанных с файлом, можно вывести лог[1] изменений нажав правую кнопку мыши и выбрав TortoiseSVN->show log. В появившемся окне есть возможность сравнить различные версии файлов, что позволяет совершать необходимые манипуляции с внесёнными изменениями.

Используемым в данной лабораторной работе визуальным командам TortoiseSVN соответствуют консольные команды SVN, приведённые ниже.

- *Команда создания рабочей копии:*

***svn checkout** URL[@REV]... [PATH]*
или
***svn co** URL[@REV]... [PATH]*
Параметры
--revision (-r) REV
--quiet (-q)
--non-recursive (-N)
--username USER
--password PASS
--no-auth-cache
--non-interactive
--ignore-externals
--config-dir DIR

- *Команда внесения изменений:*

***svn commit** [PATH...]*
или
***svn ci** [PATH...]*
Фиксирует сделанные пользователем изменения рабочей копии в хранилище. Каждое изменение сопровождается комментарием. Чтобы указать комментарий необходимо воспользоваться параметром –file *или* –message. *Если вы не воспользовались ни* --file, *ни* --message *параметром,* **svn** *запустит внешний редактор для составления комментария.*

[1] Лог — (англ. log) журнал событий.

Параметры
--message (-m) TEXT
--file (-F) FILE
--quiet (-q)
--no-unlock
--non-recursive (-N)
--targets FILENAME
--force-log
--username USER
--password PASS
--no-auth-cache
--non-interactive
--encoding ENC
--config-dir DIR

- *Команда загрузки обновлений:*

***svn update** [PATH...]*
позволяет скачать с сервера изменения в вашу рабочую копию, и автоматически объеденить их с вашеми изменениями.
Параметры
--revision (-r) REV
--non-recursive (-N)
--quiet (-q)
--diff3-cmd CMD
--username USER
--password PASS
--no-auth-cache
--non-interactive
--config-dir DIR
--ignore-externals

Задание

1. Проверить установлены ли SVN и TortoiseSVN на компьютере и в случае необходимости установть их.
2. Создать репозиторий для своего проекта.
3. Создать папку проекта и связать её с репозиторием.
4. Реализовать программу расшифровки текста по алгоритму в соответствии с вариантом. (см. Приложение 1)
5. Внести программу под SVN до исправления ошибок.

Под версионный контроль заносятся только исходные файлы.

6. Исправить ошибки, после каждого исправления внося соответствующие изменения под SVN. (Если ошибок нет, добавить тестовый комментарий в исходный код и внесите изменения под SVN)
7. Вывести на экран лог изменений файла, в котором было наибольшее количество изменений.
8. Вывести на экран сравнение фала до и после внесения одного из изменений.
9. Внести в отчёт скриншот выполнения пунктов 7 и 8.

Требования к содержанию отчета

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen) с краткими пояснениями. Сделать выводы по основным пунктам лабораторной работы.

ЛАБОРАТОРНАЯ РАБОТА № 4

Командная работа с Subversion.

Цель работы

Получение практических навыков в разработке программного продукта с использованием системы версионного контроля SVN. Получить практику командной работы над проектом. Создание патчей[1].

---

[1] **па́тч** (англ. patch — заплатка) — автоматизированное отдельно поставляемое программное средство, используемое для устранения проблем в программном обеспечении или изменения его функционала, а также сам процесс установки патча ("пропатчивание"). Исправление может применяться к уже установленной программе, либо к её исходным кодам. Сюда входит исправление ошибок, изменение внешнего вида, улучшение эргономичности или производительности программ, а также любые другие изменения, которые разработчик пожелал сделать. В качестве синонима может использоваться термин "обновление" (англ. update). Однако под словом "патч" чаще понимают исправление каких-то ошибок, в то время как под обновлением - улучшение функционала и добавление новых возможностей.

Краткие теоретические сведения

В SVN каждое изменение, сделанное в проекте, имеет свой номер - идентификатор. Версия проекта включающей все изменения до некоторого номера называется ревизией. В большинстве случаев происходит работа с последней ревизией (Head revision). Для создания новой копии проекта последней или выбранной ревизии используется операция SVNCheckout. Для перехода внесённой под SVN папки или файла к выбранной ревизии используется команда update to revision.

Для создания патча необходимо сделать две копии проекта на интересующих нас ревизиях и выбрать из них те файлы которые были изменены. Для определения необходимой ревизии и получения списка изменённых файлов следует воспользоваться командой show log.

Задание

Перед тем как группа приступит к выполнению лабораторной работы, необходимо создать на одном из компьютеров (или сервере) пустой проект так чтобы он был доступен всем выполняющим лабораторную работу. Если проект уже существует, следует узнать его адрес (URL) и приступить к выполнению пунктов 1-6.

1. Скачать проект из общего репозитория группы.
2. Добавить в проект свою папку с исходными файлами от предыдущей лабораторной работы.
3. Внести добавленную папку под SVN.
4. Изменить метод ввода/вывода, так чтобы ввод дешифруемого сообщения осуществлялся из файла input.txt, а вывод осуществлялся в файл output.txt каталога, в котором находиться программа.
5. Внести результаты под SVN.
6. Оформить внесённые изменения в виде патча, т.е. в каталоге patch создать два каталога: каталог содержащий только изменённые файлы, и каталог backup содержащий эти же файлы до изменений.

Требования к содержанию отчета

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen), с краткими пояснениями. Сделать выводы по основным пунктам лабораторной работы.

ЛАБОРАТОРНАЯ РАБОТА № 5

Контроль выполнения задач при помощи Subversion и файлов пакетной обработки.

Цель работы

Получение практических навыков в отладке программного продукта с использованием системы версионного контроля SVN. Получение навыков в использовании bat-файлов. Организация системы отчётности в проекте. Обработка событий в SVN.

Обработчик событий (hook) - специальная процедура, отслеживающая появление некоторого события.

Hook в операционной системе Windows – это механизм перехвата особой функцией событий (таких как сообщения, ввод с мыши или клавиатуры) до того, как они дойдут до приложения. Эта функция может реагировать на события и, в некоторых случаях, изменять или отменять их. Функции, получающие уведомления о событиях, называются фильтрующими функциями и различаются по типам перехватываемых событий.

В SVN hook-и встречается при обработке основных событий:

- post-commit.tmpl
- post-lock.tmpl
- post-revprop-change.tmpl
- post-unlock.tmpl
- pre-commit.tmpl
- pre-lock.tmpl
- pre-revprop-change.tmpl
- pre-unlock.tmpl
- start-commit.tmpl

Перечисленные файлы шаблонов находятся в репозитории папке hooks. Для написания собственного хука необходимо создать исполняемый, скриптовый или bat файл с именем совпадающем с

именем шаблона. Например, для обработки внесения изменений в проект, необходимо создать файл **pre-commit.cmd** с описанием действий которые необходимо сделать, и поместить его в папку hooks. Наиболее приемлемым является написание хука на скриптовом языке (например php), чтобы в случае необходимости его могли модифицировать другие пользователи.

```
echo off
"D:\Program Files\Subversion\bin\svnlook" log -t %2 %1>d:\1.txt
set /p mytmp="" <d:\1.txt
echo %mytmp% >d:\2.txt
if NOT GoodPatch==%mytmp% exit 1
```

Если в **pre-commit.cmd** будет реализован данный скрипт то опирация **commit** будет успешно выполняться только с коментарием "GoodPatch" с другими коментариями или без коментария операция **commit** будет прерываться с ошибкой 'pre-commit' hook failed.

Задание

Перед тем как группа приступит к выполнению лабораторной работы, необходимо создать в корне общего проекта пустой файл run.bat и group_result.txt и внести их под SVN, затем приступить к выполнению пунктов 1-5.

1. Выполнить команду Update на папке общего проекта. Убедиться в наличии файлов run.bat и group_result.txt.
2. Добавить в run.bat строчки, запускающие ваше приложение и записывающие результат работы из [вашего каталога]/output.txt в общий файл group_result.txt
3. Скорректировать hook отвечающий за обработку вносимых под SVN файлов таким образом, чтобы операция commit выполнялась только если в качестве комментария указанна фамилия студента выполняющего работу.
4. Внести результаты под SVN.
5. Получить лог изменений файла group_result.txt.

Требования к содержанию отчета

В отчёте зафиксировать все произведённые действия. Приложить копии экрана (Print Screen), с краткими пояснениями. Сделать выводы по основным пунктам лабораторной работы.

ПРИЛОЖЕНИЕ 1

Варианты заданий.

1. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного шифром Цезаря.
2. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного шифром простой замены.
3. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного шифром замены с использованием омофонов.
4. Реализовать алгоритм шифрования и дешифрования текста с помощью шифра Кардана.
5. Реализовать алгоритм шифрования и дешифрования текста с помощью шифра Вернама.
6. Реализовать алгоритм шифрования и дешифрования текста с помощью шифра Виженера
7. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного шифром Плейфейера.
8. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного шифром Хилла.
9. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного перестановочным шифром. (Например преобразование лесенки).
10. Реализовать алгоритм шифрования и дешифрования текста с помощью классической сети Файстеля
11. Реализовать алгоритм шифрования и дешифрования текста, зашифрованного перестановочным S-DES (упращённый DES)

ПРИЛОЖЕНИЕ 2.

Полный список консольных команд ОС Windows.

ASSOC Вывод либо изменение сопоставлений по расширениям имен файлов.
ATTRIB Отображение и изменение атрибутов файлов.
BREAK Включение и выключение режима обработки комбинации клавиш CTRL+ C.

BCDEDIT Задает свойства в базе данных загрузки для управления начальной загрузкой.
CACLS Отображение и редактирование списков управления доступом (ACL) к файлам.
CALL Вызов одного пакетного файла из другого.
CD Вывод имени либо смена текущей папки.
CHCP Вывод либо установка активной кодовой страницы.
CHDIR Вывод имени либо смена текущей папки.
CHKDSK Проверка диска и вывод статистики.
CHKNTFS Отображение или изменение выполнения проверки диска во время загрузки.
CLS Очистка экрана.
CMD Запуск еще одного интерпретатора командных строк Windows.
COLOR Установка цвета текста и фона, используемых по умолчанию.
COMP Сравнение содержимого двух файлов или двух наборов файлов.
COMPACT Отображение и изменение сжатия файлов в разделах NTFS.
CONVERT Преобразование дисковых томов FAT в NTFS. Нельзя выполнить преобразование текущего активного диска.
COPY Копирование одного или нескольких файлов в другое место.
DATE Вывод либо установка текущей даты.
DEL Удаление одного или нескольких файлов.
DIR Вывод списка файлов и подпапок из указанной папки.
DISKCOMP Сравнение содержимого двух гибких дисков.
DISKCOPY Копирование содержимого одного гибкого диска на другой.
DISKPART Отображение и настройка свойств раздела диска.
DOSKEY Редактирование и повторный вызов командных строк; создание макросов.
DRIVERQUERY Отображение текущего состояния и свойств драйвера устройства.
ECHO Вывод сообщений и переключение режима отображения команд на экране.
ENDLOCAL Конец локальных изменений среды для пакетного файла.
ERASE Удаление одного или нескольких файлов.

EXIT Завершение работы программы CMD.EXE (интерпретатора командных строк).
FC Сравнение двух файлов или двух наборов файлов и вывод различий между ними.
FIND Поиск текстовой строки в одном или нескольких файлах.
FINDSTR Поиск строк в файлах.
FOR Запуск указанной команды для каждого из файлов в наборе.
FORMAT Форматирование диска для работы с Windows.
FSUTIL Отображение и настройка свойств файловой системы.
FTYPE Вывод либо изменение типов файлов, используемых при сопоставлении по расширениям имен файлов.
GOTO Передача управления в отмеченную строку пакетного файла.
GPRESULT Отображение информации о групповой политике для компьютера или пользователя.
GRAFTABL Позволяет Windows отображать расширенный набор символов в графическом режиме.
HELP Выводит справочную информацию о командах Windows.
ICACLS Отображение, изменение, резервное копирование или восстановление списков ACL для файлов и каталогов.
IF Оператор условного выполнения команд в пакетном файле.
LABEL Создание, изменение и удаление меток тома для дисков.
MD Создание папки.
MKDIR Создание папки.
MKLINK Создание символических и жестких ссылок
MODE Конфигурирование системных устройств.
MORE Последовательный вывод данных по частям размером в один экран.
MOVE Перемещение одного или нескольких файлов из одной папки в другую.
OPENFILES Отображение файлов, открытых на общей папке удаленным пользователем.
PATH Отображает или устанавливает путь поиска исполняемых файлов.
PAUSE Приостанавливает выполнение пакетного файла и выводит сообщение.
POPD Восстанавливает предыдущее значение активной папки, сохраненное с помощью команды PUSHD.
PRINT Выводит на печать содержимое текстового файла.

PROMPT Изменяет приглашение в командной строке Windows.
PUSHD Сохраняет значение активной папки и переходит к другой папке.
RD Удаляет папку.
RECOVER Восстанавливает данные, которые можно прочитать, с плохого или поврежденного диска.
REM Помещает комментарии в пакетные файлы и файл CONFIG.SYS.
REN Переименовывает файлы или папки.
RENAME Переименовывает файлы или папки.
REPLACE Замещает файлы.
RMDIR Удаление папки.
ROBOCOPY Улучшенное средство копирования файлов и деревьев каталогов
SET Показывает, устанавливает и удаляет переменные среды Windows.
SETLOCAL Начинает локализацию изменений среды в пакетном файле.
SC Отображает и настраивает службы (фоновые процессы).
SCHTASKS Выполняет команды и запускает программы по расписанию.
SHIFT Изменение положения (сдвиг) подставляемых параметров для пакетного файла.
SHUTDOWN Локальное или удаленное выключение компьютера.
SORT Сортировка ввода.
START Выполнение программы или команды в отдельном окне.
SUBST Назначение заданному пути имени диска.
SYSTEMINFO Вывод сведений о системе и конфигурации компьютера.
TASKLIST Отображение всех выполняемых задач, включая службы.
TASKKILL Прекращение или остановка процесса или приложения.
TIME Вывод и установка системного времени.
TITLE Назначение заголовка окна для текущего сеанса интерпретатора командных строк CMD.EXE.
TREE Графическое отображение структуры каталогов диска или папки.
TYPE Вывод на экран содержимого текстовых файлов.
VER Вывод сведений о версии Windows.

VERIFY Установка режима проверки правильности записи файлов на диск.
VOL Вывод метки и серийного номера тома для диска.
XCOPY Копирование файлов и деревьев каталогов.
WMIC Вывод сведений WMI в интерактивной среде.

Список используемых источников

1. Командная строка Microsoft Windows. Уильям Р. Станек. Изд. БХВ-Петербург, 2009 г. Санкт-Петербург.
2. http://svnbook.red-bean.com/nightly/ru/index.html
3. http://subversion.tigris.org/files/documents/15/36797/svn-1.4.3-setup.exe
4. http://downloads.sourceforge.net/tortoisesvn/TortoiseSVN-1.6.2.16344-win32-svn-1.6.2.msi?download

СОДЕРЖАНИЕ

Учебное издание

Руководство к выполнению лабораторных работ по дисциплине «Сетевое администрирование»

Авторы
Шуранов Евгений Витальевич
Петров Глеб Анатольевич
Левин Илья Иванович

Редактор
И.Г. Максимова

ЛР № 020309 от 30.12.96.

Подписано в печать 26.02.09. Формат $60 \times 90^{1}/_{16}$. Печать офсетная.
Печ. л. 2,3. Тираж 300. Зак. № 7.

195196, СПб, Малоохтинский пр. 98. РГГМУ