Операционные системы

Лабораторная работа № 2

Командный язык ОС Unix - shell оперирует с командами. Shell фактически есть язык программирования очень высокого уровня. На этом языке пользователь осуществляет управление компьютером. Обычно, после входа в систему вы начинаете взаимодействовать с командной оболочкой. Признаком того, что оболочка (shell) готова к приему команд, служит выдаваемое ею на экран приглашение. В простейшем случае это один доллар ("$"). Shell не является необходимым и единственным командным языком (хотя именно он стандартизован в рамках POSIX [POSIX 1003.2] - стандарта мобильных систем). Например, немалой популярностью пользуется язык cshell, есть также kshell, bashell (из наиболее популярных в последнее время) и другие. Более того, каждый пользователь может создать свой командный язык. Может одновременно на одном экземпляре операционной системы работать с разными командными языками.

Shell - это одна из многих команд UNIX. То есть в набор команд оболочки (интерпретатора) "shell" входит команда "sh" - вызов интерпретатора "shell". Первый "shell" вызывается автоматически при вашем входе в систему и выдает на экран приглашение. После этого вы можете вызывать на выполнение любые команды, в том числе и снова сам "shell", который вам создаст новую оболочку внутри прежней.

Основные команды shell

Приведем основные команды shell. Следует помнить, что в UNIX различаются регистры символов, то есть имена File, file и FILE будут являться тремя разными именами файла.

cd <путь> – изменение текущего каталога.

ls [путь/] [имя файла] [параметры] – просмотр каталога.

mkdir <путь> – создание каталога.

rmdir <путь> – уничтожение каталога.

cat> файл – создание пустого текстового файла.

rm <имя файла> – удаление файлов.

mv <имя файла1> <имя файла2> – переименование файлов.

cp <имя файла1> … <имя файла N> <имя каталога> – копирование файлов.

rm -r <имя каталога> – удаление каталога со всем содержимым.

cat <имя файла1> … <имя файла N> – вывод файлов на экран.

clear – очистка экрана монитора

grep <шаблон> <имя файла1> … <имя файла N> – выдает все строки в названном файле (файлах), которые содержат заданный шаблон.

vi – вызов текстового редактора vi.

ключ –h (--help) – вывод справки о команде - команда – help.

man – справочная информация - man команда

Можно применять перенаправление вывода, например ls>1.txt выведет содержимое текущего каталога в файл 1.txt.

Пакетные файлы shell

При помощи языка shell, можно создавать пакетные файлы, их еще называют скрипты (script). Исполнение командного файла мало отличается от исполнения команд в интерактивном режиме. Если в строке встречается "#', то с этого места до конца строки все символы являются комментарием.

" ;" – означает перевод строки, служит для разделения команд. Можно писать несколько команд в одной строке, разделяя их " ; ". Если в начале файла стоит "#!", то за ним должно стоять имя оболочки shell, под которым будет работать этот командный файл. #!/bin/sh - каким интерпретатором хотим исполнять этот файл.

Возможная альтернатива: передать командный файл в качестве аргумента оболочки shell: sh myfile.

Для исполнения командного файла myfile можно сделать его выполняемым с помощью команды chmod 755 myfile. Его также можно выполнить, вызвав явно команду "sh" ("shell") sh myfile или sh < myfile. Файл можно выполнить и в текущем экземпляре "shell". Для этого существует специфическая команда "." (точка), то есть .myfile.

Метасимволы

Иногда возникают задачи, для которых значение отдельных параметров неизвестно, или требуется выполнить схожие действия для большого числа операндов. Например, необходимо найти файл, первый символ имени которого неизвестен, или необходимо выполнить копирование всех файлов с расширением .txt. Для этих задач удобно использовать специальные символы (метасимволы) посредством которых упрощается задача указания файлов или групп файлов как аргументов команды.

Метасимвол «*» осуществляет поиск любой строки символов, включая нулевую (пустую) строку. Можно использовать «*» для обозначения полного или частичного имени файла. Просто символ «*» ищет все имена файлов и справочников в текущем справочнике, за исключением тех, которые начинаются с точки.

Метасимвол «?» осуществляет поиск любого одного символа в имени файла за исключением лидирующей точки. Предположим, вы имеете книгу, в которой 12 глав и хотите получить список глав до 9-ой главы. Если ваш справочник содержит следующие файлы:

Chapter1

Chapter2

Chapter5

Chapter9

Chapter11

то введите команду ls с метасимволом ? для получения всех глав, которые начинаются со строки "Chapter" и заканчиваются одним символом:

$ ls Chapter?

Chapter1

Chapter2

Chapter5

Chapter9

Чтобы получить список всех глав в текущем справочнике, используйте метасимвол «*»:

ls Chapter*

Если вы хотите найти любой символ из перечисленных вами символов, то заключите эти символы в квадратные скобки. Предположим, ваш справочник содержит следующие файлы: cat, fat, mat, rat. Если вы воспользуетесь в качестве части имени файла шаблоном [crf], то будут найдены имена файлов, в которые входят либо буква "c", либо буква "r", либо буква "f" в указанной позиции. Пример:

$ ls [crf]at

cat

fat

rat

$

Символы, которые могут быть сгруппированы в скобки, называются классом символов.

Скобки могут также использоваться для обозначения диапазона символов, цифр или букв. Предположим в вашем справочнике содержатся следующие файлы: chapter1, chapter2, chapter3, chapter4, chapter5, chapter6.

Если вы укажете:

chapter[1-5]

то будут найдены файлы с chapter1 по chapter5.

Класс символов можно также указать с помощью диапазона букв. Если вы укажете [A-Z], то будут найдены только большие буквы, если [a-z] - то малые буквы.

Переменные

Все переменные в языке shell - текстовые. Их имена должны начинаться с буквы и состоять из латинских букв, цифр и знака подчеркивания. Чтобы воспользоваться значением переменной, надо перед ней поставить символ «$». Использование значения переменной называется подстановкой. Начальные значения переменным с именем могут быть установлены следующим образом:

<имя>=<значение> [<имя>=<значение>] ...

Не может быть одновременно функции и переменной с одинаковыми именами. Для подстановки значений переменных возможны также следующие конструкции:

${<переменная>}

если значение <переменной> определено, то оно подставляется. Скобки применяются лишь если за <переменной> следует символ, который без скобок приклеится к имени.

Для проведения арифметических вычислений можно воспользоваться командой let. В большинстве случаев, let можно считать упрощенным вариантом команды expr. expr - универсальный обработчик выражений: вычисляет заданное выражение (аргументы должны отделяться пробелами).

Выражения могут быть арифметическими, логическими или строковыми.

Пример. Сложение двух чисел

```
a=1
b=2
let c=$a+$b ‘ c=`expr $a+$b`
echo $c
echo нажмите любую клавишу
read
```

Для организации ввода переменных с клавиатуры используется команда read

read [<переменная> ...]

Читается из стандартного ввода одна строка; первое ее слово присваивается первой переменной, второе - второй и т.д., причем все оставшиеся слова присваиваются последней переменной.

Пример. Ввод значений переменных из командной строки

```
echo введите a
read a
echo введите b
read b
c=`expr $a+$b`
echo $c
echo нажмите любую клавишу
read
```

Параметры командной строки

Аргументы командной строки в shell-скриптах задаются точно также как в пакетных файлах, только при обращении к ним используется знак «$».

Пример. Параметры командной строки.

```
let c=$1+$2
echo $c
echo нажмите любую клавишу
Read
```

Ветвления в shell

1. Оператор if

if <список1>

then

<список2>

[elif <список3>

then

<список4>]

. . .

[else

<список5>]

fi

Выполняется <список1> и, если код его завершения 0, то выполняется <список2>, иначе - <список3> и, если и его код завершения 0, то выполняется <список4>. Если же это не так, то выполняется <список5>.

Части elif и else могут отсутствовать.

2. Оператор case

case $<переменная> in

<шаблон> | <шаблон>...) <список> ;;

. . .

esac

Оператор выбора выполняет <список>, соответствующий первому <шаблону>, которому удовлетворяет <переменная>. Форма шаблона та же, что и используемая для генерации имен файлов. Часть | шаблон... может отсутствовать.

Циклы в shell

1. Цикл for

for <переменная> [in <набор>]

do

<список>

done

Если часть in <набор> опущена, то это означает in "$@" (то есть in $1 $2 ... $n).

Пример. Вывести на экран содержимое всех текстовых файлов текущего каталога:

for f in *.txt

do

cat $f

done

2. Цикл while (until)

while <список1>

do

<список2>

done

До тех пор, пока код завершения последней команды <списка1> есть 0, выполняются команды <списка2>. При замене служебного слова while на until условие выхода из цикла меняется на противоположное.

В качестве одной из команд <списка1> может быть команда true (false). По этой команде не выполняется никаких действий, а код завершения устанавливается 0 (-1). Эти команды применяются для организации бесконечных циклов. Выход из такого цикла можно осуществить лишь по команде break.

Задания к лабораторной работе

Задание 1. Основы работы в shell

1. Откройте окно терминала (в зависимости от дистрибутива доступ к терминалу может осуществляться разными способами, чаще всего Главное меню-Стандартные-Терминал).
2. Создайте в домашнем каталоге каталог, названный вашей фамилией.
3. В своем каталоге создать каталоги First и Second.
4. Работа в каталоге First:
- Создать текстовый файл, содержащий рецепт приготовления кофе.
- Создать текстовый файл, содержащий описание команды FIND.
5. В каталоге Second:
- Скопировать все файлы из каталога First.
- Объединить файлы, скопированные из каталога First в один файл (используйте cat и перенаправление вывода в файл).
- Удалить файлы, скопированные из каталога First.
6. Предъявите работу преподавателю (записи в терминале).
7. Очистите экран.

Задание 2. Работа с пакетными файлами shell

1. Создайте командный файл DELCO.sh, выполняющий действия:
- Вывод на экран: "Копирование и удаление файла".
- Создание в домашнем каталоге каталога DIR1, и в нем создание каталога DIR2.
- Копирование файла с именем 1.TXT, предварительно созданного в папке группы, в файл с именем 1NEW.TXT в каталоге /DIR1/DIR2.
- Удаление исходного файла 1.TXT.
- Вывод на экран: "Файл скопирован и удален".
- Пауза до нажатия клавиши (для выполнения этого действия можно воспользоваться командой read).
2. Создайте командный файл с именем REN.sh, выполняющий действия:
- Вывод на экран “Объединение и переименование файлов”.
- Объединение содержимого файлов A.TXT и B.TXT, предварительно созданных в папке своей группы, под именем C.TXT.
- Вывод содержимого объединенного файла на экран.
- Ожидание нажатия клавиши.
- Переименование файлов A.TXT и B.TXT в FINA.TXT и FINB.TXT соответственно.
- Вывод на экран: "Задание выполнено".

Предъявите работу преподавателю, удалите все созданные вами файлы и каталоги.

Задание 3. Работа с переменными shell

Напишите скрипт для shell, сравнивающий два числа. Числа вводятся пользователем с клавиатуры. В результате сравнения выводится одно из следующих сообщений: «Первое число больше», «Второе число больше», «Числа равны». Для сравнения чисел используйте команду test.

Задание 4. Параметры командной строки в скриптах shell

Напишите скрипт для shell, определяющий наличие файла в директории с вводимым пользователем именем. Имя искомого файла передается как параметр командной строки. В случае если файл присутствует в директории, случае если файл присутствует в директории, выводится следующая информация о файле: доступен ли файл на чтение, запись, исполнение; иначе выводится сообщение «файл не найден». Для получения информации об атрибутах файла воспользуйтесь командой test, синтаксис команды разберите самостоятельно.

Задание 5. Организация цикла в скриптах shell

Создайте пакетный файл, выводящий имя и содержимое всех файлов с расширением .txt в задаваемой пользователем директории. Каталог задается из командной строки. Перед выводом содержимого каждого файла выводится полное имя файла (включая каталог).

Задание 6. Резервное копирование файлов

Создайте скрипт, выполняющий резервное копирование каталога. Имя резервного каталога задается пользователем, если пользователь не задает имя каталога, то копия помещается в том же разделе, где исходный каталог. В зависимости от введенного пользователем параметра командной строки, скрипт должен предусматривать следующие режимы работы:

1) замещение файлов (файлы из исходного каталога замещают файлы резервного);

2) добавление файлов (если в резервном каталоге файла не существует, то он добавляется из исходного, прочие файлы не заменяются);

3) удаление файлов (если в резервном каталоге существует файл, которого нет в исходном, то он удаляется);

4) синхронизация файлов (сравниваются даты последней модификации двух файлов, более новый файл копируется в резервный каталог).

Все режимы работы должны предусматривать рекурсивный обход вложенных каталогов. Предусмотрите возможность вызова справки по работе пакетного файла. Для сравнения времен модификации файлов используйте команду test.