Лабораторная работа №11

Отчет

Форис Анастасия Дмитриевна

Содержание

Список иллюстраций

Список таблиц

Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Задание

1. Используя команды getopts grep , напишем командный файл, который анализирует командную строку с ключами: • -iinputfile — прочитать данные из указанного файла; • -ooutputfile — вывести данные в указанный файл; • -pшаблон — указать шаблон для поиска; • -C — различать большие и малые буквы; • -n — выдавать номера строк; а затем ищет в указанном файле нужные строки, определяемые ключом -p.
2. Мы написали на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции exit (n), передавая информацию о коде завершения в оболочку. Командный файл должен вызывать эту программу и проанализировав с помощью команды $?. Код возврата, выдает сообщение о том, какое число было введено.
3. Мы написали командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до ⊠ (например, 1.tmp, 2.tmp, 3.tmp,4.tmp и т.д.). Число файлов, которые необходимо создать, передается в аргумент командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
4. Мы написали командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировали его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (команда find).

Выполнение лабораторной работы

1. Используя команды getopts grep , напишем командный файл, который анализирует командную строку с ключами: • -iinputfile — прочитать данные из указанного файла; • -ooutputfile — вывести данные в указанный файл; • -pшаблон — указать шаблон для поиска; • -C — различать большие и малые буквы; • -n — выдавать номера строк; а затем ищет в указанном файле нужные строки, определяемые ключом -p. Рис.1 Анализ командной строки
2. Мы написали на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции exit (n), передавая информацию о коде завершения в оболочку. Командный файл должен вызывать эту программу и проанализировав с помощью команды $?. Код возврата, выдает сообщение о том, какое число было введено. Рис.2 Исполнение Рис.3 Командный файл
3. Мы написали командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до ⊠ (например, 1.tmp, 2.tmp, 3.tmp,4.tmp и т.д.). Число файлов, которые необходимо создать, передается в аргумент командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).Рис.4 Командный файл и исполнение
4. Мы написали командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировали его так, чтобы запаковывались только те файлы, которые были изменены ме-

нее недели тому назад (команда find).Рис.5 Командный файл и исполнение

Выводы

В ходе лабораторной работы мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Контрольные вопросы

1. Весьма необходимой при программировании является команда getopts, которая осущ

Флаги – это опции командной строки, обычно помеченные знаком минус; Например, -F является флагом для команды ls -F. Иногда эти флаги имеют аргументы, связанные с ними. Программы интерпретируют эти флаги, соответствующим образом изменяя свое поведение. Строка опций option-string — это список возможных букв и чисел соответствующего флага. Если ожидается, что некоторый флаг будет сопровождаться некоторым аргументом, то за этой буквой должно следовать двоеточие. Соответствующей переменной присваивается буква данной опции. Если команда getopts может распознать аргумент, она возвращает истину. Принято включать getopts в цикл while и анализировать введенные данные с помощью оператора case. Предположим, необходимо распознать командную строку следующего формата: testprog -ifile_in.txt -ofile_out.doc -L -t -r Вот как выглядит использование оператора getopts в этом случае: while getopts o:i:Ltr optletter do case $optletterino)oflag = 1; oval$ =OPTARG;; i) iflag=1; ival=$OPTARG;; L) Lflag=1;; t) tflag=1;; r) rflag=1;; *) echo Illegal option $optletter esac done Функция getopts включает две специальные переменные среды – OPTARG и OPTIND. Если ожидается дополнительное значение, то OPTARG устанавливается в значение этого аргумента (будет равна file_in.txt для опции i и file_out.doc для опции o) . OPTIND является числовым индексом на упомянутый аргумент. Функция getopts также понимает переменные типа массив, следовательно, можно использовать ее в функции не только для синтаксического анализа аргументов функций, но и*

для анализа введенных пользователем данных. 2. При перечислении имен файлов текущего каталога можно использовать следующие символы: • — соответствует произвольной, в том числе и пустой строке; • ? — соответствует любому одному символу; • [c1-c1] — соответствует любому символу, лексикографически на ходящемуся между символами c1 и c2. • echo * — выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды ls; • ls *.c — выведет все файлы с последними двумя символами, равными .c. • echo prog.? — выдаст все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются prog. . • [a-z]* — соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита. 3. Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости от результатов проверки некоторого условия. Для решения подобных задач язык программирования bash предоставляет Вам возможность использовать такие управляющие конструкции, как for, case, if и while. С точки зрения командного процессора эти управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути дела являются операторами языка программирования bash. Поэтому при описании языка программирования bash термин оператор будет использоваться наравне с термином команда. 4. Два несложных способа позволяют вам прерывать циклы в оболочке bash. Команда break завершает выполнение цикла, а команда continue завершает данную итерацию блока операторов. Команда break полезна для завершения цикла while в ситуациях, когда условие перестает быть правильным. Пример бесконечного цикла while, с прерыванием в момент, когда файл перестает существовать: while true do if [! -f $file] then break fi sleep 10 done 5. Команды ОС UNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях. Команда test, например, создана специально для использования в командных

файлах. Единственная функция этой команды заключается в выработке кода завершения. 6. Введенная строка означает условие существования файла man$s$/i.$s 7. Если речь идет о 2-х параллельных действиях, то это while. когда мы показываем, что сначала делается 1-е действие. потом оно заканчивается при наступлении 2-го действия, применяем until.