Отчёт по лабораторной работе №11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Грузинова Елизавета Константиновна. НКНбд-02-21

Содержание

# 1 Цель работы

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

# 2 Задание

1. Используя команды getopts grep, написать 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. Написать 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 в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды $?, выдать сообщение о том, какое число было введено.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp,4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
4. Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find).

# 3 Теоретическое введение

Циклы Bash - это циклические конструкции, используемые для итерационного выполнения (перебора) любого заданного количества задач до тех пор, пока не будут выполнены все пункты в указанном списке или же предопределенные условия. Циклы в Bash имеют три основных типа. Цикл for используется для повторения любого заданного кода для любого количества элементов в заданном списке. Следующий вид циклов в нашем списке - цикл while. Конкретно этот цикл действует по заданному условию. То есть он будет выполнять код, заключенный в рамки DO и DONE пока заданное условие истинно. Как только заданное условие станет ложным, выполнение цикла прекратится. Последний цикл, который мы рассмотрим в этой статье по написанию скриптов - это цикл until. Цикл until действует прямо противоположно циклу while. Цикл until также действует по заданному условию. Однако код, заключенный между DO и DONE, будет выполняться только до тех пор, пока это условие не изменится с ложного на истинное. [1]

# 4 Выполнение лабораторной работы

1. Используя команды getopts grep, написать 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. 1-3)

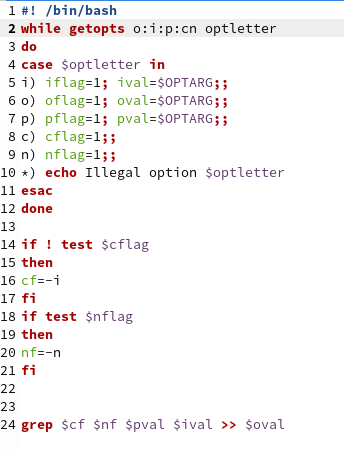


Рис. 1: Код командного файла

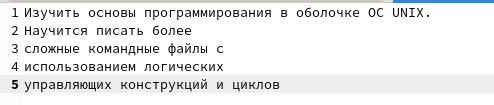


Рис. 2: Текст файла 1.txt

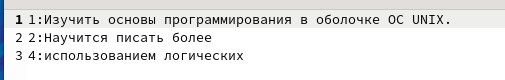


Рис. 3: Результаты поиска файла 1.txt в 2.txt

1. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp,4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют). (рис. 4-6)

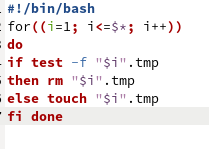


Рис. 4: Код командного файла

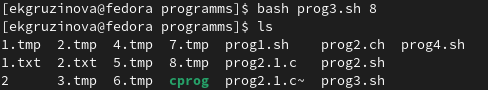


Рис. 5: Создание файло 1.tmp, 2.tmp и т. д.

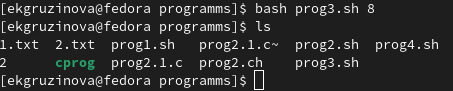


Рис. 6: Удаление этих файлов этой же программой

1. Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find). рис. 7-10)

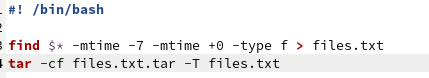


Рис. 7: Код командного файла

Рис. 8: Работа в терминале

Рис. 8: Работа в терминале



Рис. 9: Результат в виде архива tar

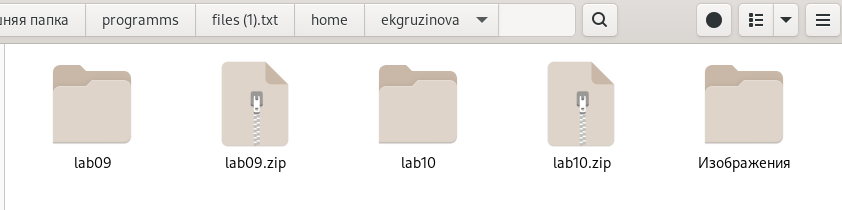


Рис. 10: Содержимое этого архива

# 5 Контрольные вопросы.

1. Каково предназначение команды getopts?

Getopts – это POSIX-совместимая встроенная функция bash, которая принимает короткие аргументы, такие как -h, -v, -b и т.д.

1. Какое отношение метасимволы имеют к генерации имён файлов?

После всех подстановок в каждом слове команды ищутся символы \*,?, и [. Если находится хотя бы один из них, то это слово рассматривается как шаблон имен файлов и заменяется именами файлов, удовлетворяющих данному шаблону (в алфавитном порядке). Если ни одно имя файла не удовлетворяет шаблону, то он остается неизменным. Значения указанных символов:

* любая строка, включая и пустую ? один любой символ […] любой из указанных между ними символов. Пара символов, разделенных знаком -, означает любой символ, который находится между ними, включая и их самих. Если первым символом после “[” идет “!”, то указанные символы не должны входить в имя файла

1. Какие операторы управления действиями вы знаете?

Точка с запятой (;), амперсанд (&), символ доллара со знаком вопроса ($?), двойной амперсанд (&&) и Знак фунта (#).

1. Какие операторы используются для прерывания цикла?

Break, return.

1. Для чего нужны команды false и true?

Если мы выполняем команды, они действительно “ничего не делают”, за исключением того, что команда true возвращает 0, а команда false возвращает 1. Команда false необходима для возращения кода сбоя, а true позволяет создавать бесконечные циклы.

1. Что означает строка if test -f mani.$s, встреченная в командном файле?

Если mani.$s существует и является обычным файлом, то…

1. Объясните различия между конструкциями while и until

While действует до тех пор, пока истинное в условии не станет ложным. Until же, наоборот, действует до тех пор, пока ложное в условии не станет истинным.

# 6 Выводы

В процессе выполнения лабораторной работы изучила основы программмирования в оболочке OC UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций.

# Список литературы

1. Программирование Bash. Часть третья [Электронный ресурс]. URL: <https://blog.yodo.im/programmirovanie-bash-chast-tretya/>.