Отчёт по лабораторной работе №11

Дисциплина: Операционные системы

Батова Ирина Сергеевна, НММбд-01-22

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.

1. 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.
2. 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).
3. 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 Выполнение лабораторной работы

1. Для начала работы создаем файл для написания скрипта и открываем его (рис. ??).

Создание файла ‘file21.sh’

Создание файла ‘file21.sh’

Нам необходимо, используя команды getopts grep, написать командный файл, который анализирует командную строку с ключами, а затем ищет в указанном файле нужные строки, определяемые ключом -p.

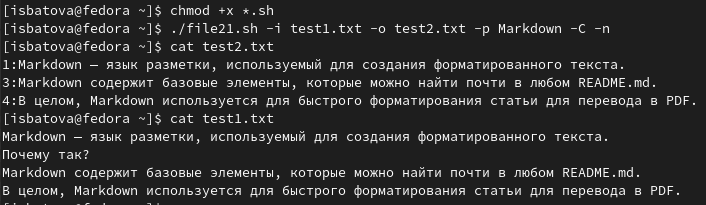
Вводим скрипт в наш файл (рис. ??).



Программа 1

В данном скрипте мы сначала вводим соответствующие опциям переменные и присваиваем этим переменным 0. Далее программа просматривает командную строку на наличие опций и присваивает 1 тем опциям (переменным), которые есть в командной строке. После этого мы используем команду if для проверки наличия различных опций. Сначала проверяем есть ли шаблон для поиска (слово, которое ищем), а потом есть ли файл, в котором будет искаться это слово. При невыполнении хотя бы одного из условий программа выводит ошибку. Далее мы перебираем различное сочетание опций и выводим соответствующие строки в файл.

Далее добавляем право на выполнение файла командой ’chmod +x \*.sh’, создаем файл с текстом (test1.txt) и пустой файл, в который выводятся данные (test2.txt) и выполняем скрипт командой ‘./file21.sh (аргументы)’. Для проверки корректности выполнения просматриваем содержимое файла командой cat (рис. ??).



Проверка корректности программы 1

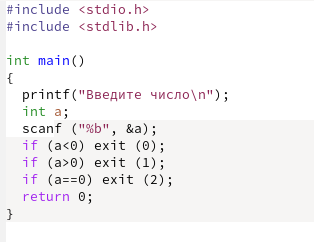
1. Для начала работы создаем файл для написания скрипта на языке Си и файл для написания скрипта на bash. Сначала открываем первый файл (рис. ??).

Создание файлов ‘file22.c’ и ‘file22.sh’

Создание файлов ‘file22.c’ и ‘file22.sh’

Нам необходимо, написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. После этого программа должна завершаться с помощью функции exit(n), передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав его, выдать сообщение о том, какое число было введено.

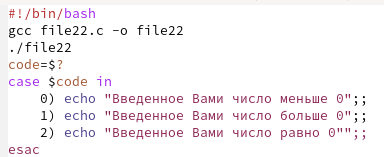
Вводим скрипт на языке си в файл ‘file22.c’ (рис. ??).



Программа 2 на языке Си

В данном скрипте мы запрашиваем у пользователя число, затем программа считывает его и определяет, число больше нуля, равно нулю или меньше нуля.

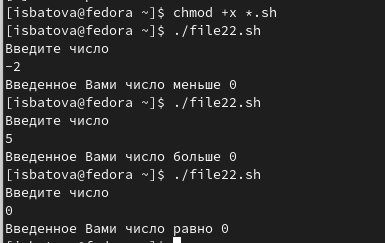
Далее открываем файл ‘file22.sh’ и вводим в него скрипт (рис. ??).



Программа 2

В данном скрипте мы сначала компилируем файл с программой на языке Си в объектный файл. Далее вводим команды запуска этого объектного файла, затем программа анализирует, какое число получилось на выходе. После этого с помощью команды “case” выводим соответствующее сообщение.

Далее добавляем право на выполнение файла командой ’chmod +x \*.sh’ и выполняем скрипт командой ‘./file22.sh’. Для проверки корректности выполнения я ввела несколько чисел из разных диапазонов (рис. ??).



Проверка корректности программы 2

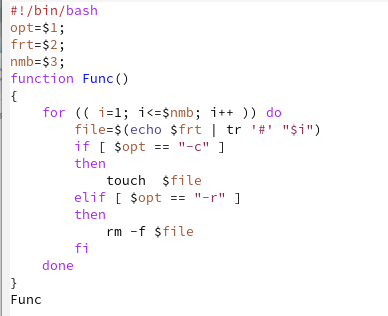
1. Для начала работы создаем файл для написания скрипта и открываем его (рис. ??).

Создание файла ‘file23.sh’

Создание файла ‘file23.sh’

Нам необходимо написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N, число файлов, которые необходимо создать, передаётся в аргументы командной строки. При этом этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).

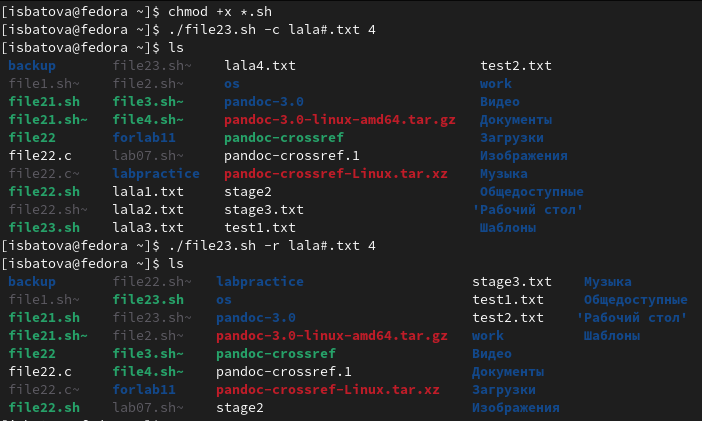
Вводим скрипт в наш файл (рис. ??).



Программа 3

В данном скрипте мы вводим переменные для опций, которые будут введены пользователям - опция -r или -с (remove - удалить или create - создать), формат файла и количество файлов, которые нужно создать. После этого задаем функцию Func, которая будет удалять и создавать нужное количество файлов нужного формата в зависимости от аргументов, и запускаем ее внутри программы.

Далее добавляем право на выполнение файла командой ’chmod +x \*.sh’ и выполняем скрипт командой ‘./file23.sh (аргументы)’. Для проверки корректности выполнения просматриваем содержимое каталога командой ls и после создания, и после удаления (рис. ??).



Проверка корректности программы 3

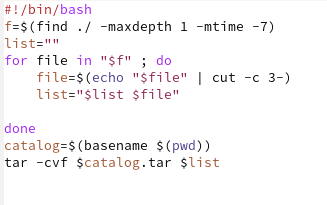
1. Для начала работы создаем файл для написания скрипта и открываем его (рис. ??).

Создание файла ‘file24.sh’

Создание файла ‘file24.sh’

Нам необходимо написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории, а затем модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад.

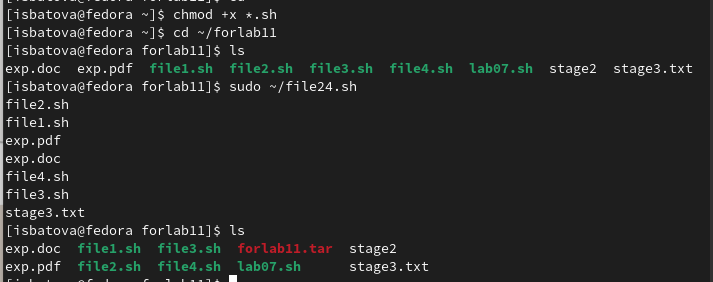
Вводим скрипт в наш файл (рис. ??).



Программа 4

В данном скрипте сначала с помощью команды find находим файлы, которые были изменены меньше недели назад и создаем переменную для списка файлов, которые будем архивировать. Далее с помощью цикла for анализируем каждый файл по времени создания и в зависимости от этого добавляем его в “список файлов” или нет. После окончания цикла архивируем все файлы, содержащиеся в списке.

Далее добавляем право на выполнение файла командой ’chmod +x \*.sh’, переходим в специально созданный каталог ‘forlab11’, в который помещены файлы различное давности, и выполняем скрипт командой ‘sudo ~/file24.sh’. Для проверки корректности выполнения просматриваем содержимое каталога командой ls и видим, что был создан архив файлов (рис. ??).



Проверка корректности программы 4

# 4 Выводы

В данной лабораторной работе мной были изучены основы программирования в оболочке ОС UNIX. Помимо этого, я научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 5 Контрольные вопросы

1. Команда getopts осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.
2. При перечислении имён файлов текущего каталога используются такие метасимволы, как “\*” (соответствует произвольной, в том числе и пустой строке) , “?” (соответствует любому одинарному символу), “[c1-c2]” — соответствует любому символу, лексикографически находящемуся между символами c1 и с2.
3. В языке bash есть следующие операторы действия: for (при каждом следующем выполнении оператора цикла for переменная принимает следующее значение из списка значений), case (реализует возможность ветвления на произвольное число ветвей), if (сначала выполняется последовательность команд (операторов), которую задаёт список команд в строке, содержащей служебное слово if, а затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), то будет выполнена последовательность команд (операторов), которую задаёт список команд в строке, содержащей служебное слово then) и while (cначала выполняется последовательность команд (операторов), которую задаёт список команд в строке, содержащей служебное слово while, а затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), выполняется последовательность команд (операторов), которую задаёт списоккоманд в строке, содержащей служебное слово do, после чего осуществляется безусловный переход на начало оператора цикла while).
4. Для прерывания цикла используются операторы break (завершает выполнение цикла) и оператор continue (завершает данную итерацию блока операторов).
5. Команда true всегда возвращает код завершения, равный нулю, и команда false, которая всегда возвращает код завершения, не равный нулю.
6. Строка if test -f mani.s/s и является ли он обычным файлом.
7. При замене в операторе цикла while служебного слова while на until условие, при выполнении которого осуществляется выход из цикла, меняется на противоположное.