Лабораторная работа №12

Отчет

Устинова Виктория Вадимовна

Содержание

# 1 Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

# 2 Задание

1. Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из ар- хиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.
3. Написать командный файл — аналог команды ls (без использования самой этой ко- манды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.
4. Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента ко- мандной строки.

# 3 Выполнение лабораторной работы

Написать скрипт, который будет делать резервную копию самого себя(рис. 1).

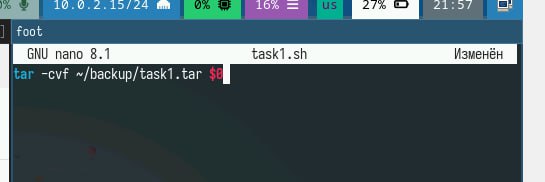


Рис. 1: Написали скрипт

Запускаем файл(рис. 2).

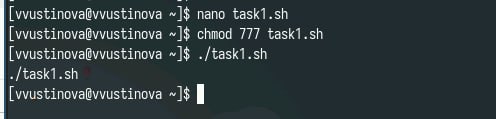


Рис. 2: Вводим команду ./task1.sh

При этом файл должен архивироваться одним из архиваторов на выбор(рис. 3).

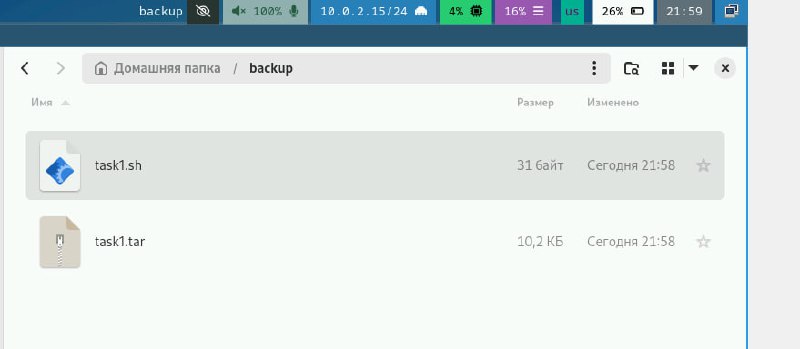


Рис. 3: Перешли с созданную папку backup и разархивировали файл

Создаем новый файл для второго задания и открываем редактор(рис. 4).

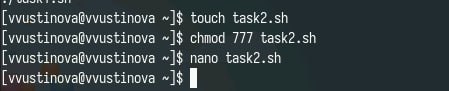


Рис. 4: Редактор nano

Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки(рис. 5).

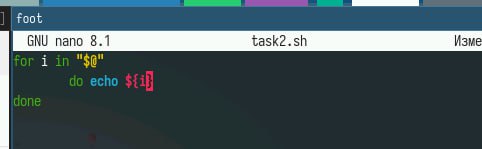


Рис. 5: Пишем код

Скрипт может последовательно распечатывать значения всех переданных аргументов(рис. 6).

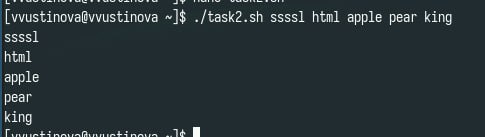


Рис. 6: У нас выводит последовательно, как и требуется

Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir)(рис. 7).



Рис. 7: Пишем код

Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога(рис. 8).

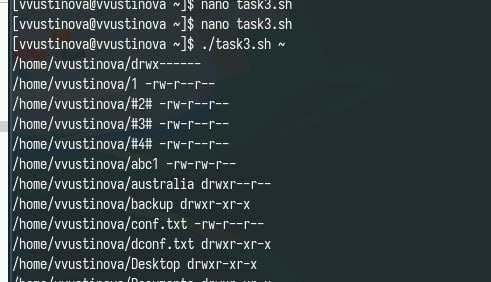


Рис. 8: Выводит необходимое

Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. (рис. 9).

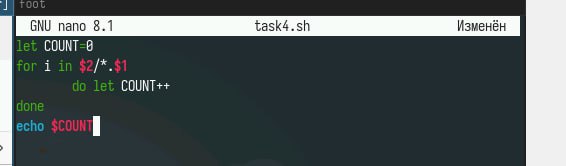


Рис. 9: Пишем код

Проверяем выполнение(рис. **¿fig:0010?**).

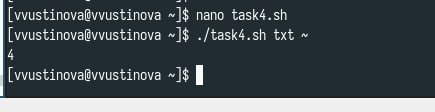


Рис. 10: Выводит необходимое, я проверила, ровно 4 файла txt находятся у меня в домашней директории

# 4 Выводы

Мы успешно изучили основы программирования в оболочке ОС UNIX/Linux. Научились писать небольшие командные файлы.

## 4.1 Ответы на контрольные вопросы

1. Командная оболочка: Интерфейс для взаимодействия с ОС через команды. Примеры: bash, zsh, fish. Отличаются синтаксисом, возможностями.
2. POSIX: Семейство стандартов, определяющих совместимость ОС Unix-типа.
3. Переменные: var=value, Массивы: array=(item1 item2).
4. let: Для арифметических операций. read: Для чтения ввода от пользователя.
5. Арифметика: +, -, , /, %, \\*.
6. (( )): Арифметические выражения и вычисления.
7. Стандартные переменные: PATH, HOME, USER, PWD, SHELL.
8. Метасимволы: Символы, имеющие специальное значение для оболочки (\*, ?, []).
9. Экранирование: Обратный слэш , кавычки (одинарные/двойные).
10. Командные файлы: Создать текстовый файл с командами, дать права на выполнение (chmod +x), запустить ./filename.
11. Функции: function\_name() { commands; }.
12. Тип файла: Команды test -d (каталог), test -f (обычный файл).
13. Назначение: set: Установить опции оболочки/переменные, typeset: Объявить атрибуты переменных, unset: Удалить переменную.
14. Параметры: Передаются при запуске скрипта: ./script arg1 arg2. Доступны как $1, $2 и т.д.
15. Специальные переменные: $?: Код возврата последней команды, $$: PID текущего процесса, $@: Все аргументы.