Лабораторная работа № 12

Программирование в командном процессоре ОС UNIX. Командные файлы

Мальянц Виктория Кареновна

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 архиваторов 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 команды 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 командной строки.
5. Контрольные вопросы.

# 3 Выполнение лабораторной работы

Пишу скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в моем домашнем каталоге. При этом файл должен архивироваться архиватором tar. Способ использования команд архивации узнаю, изучив справку (рис. 1) (рис. 2) (рис. 3) (рис. 4).

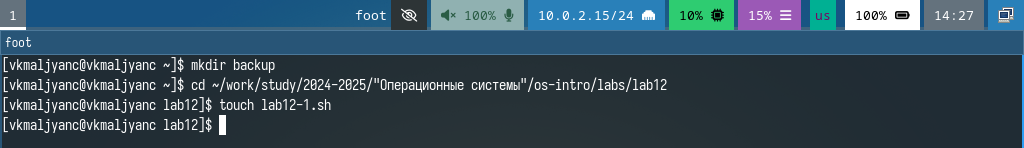


Рис. 1: Создание каталога backup и создание файла lab12-1.sh

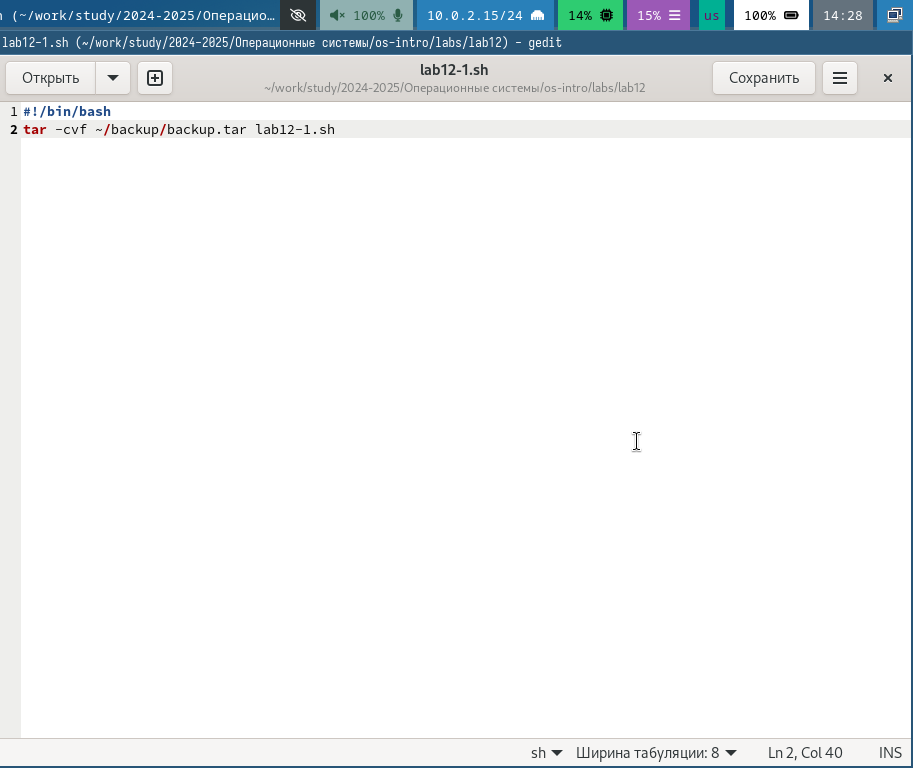


Рис. 2: Редактирование файла

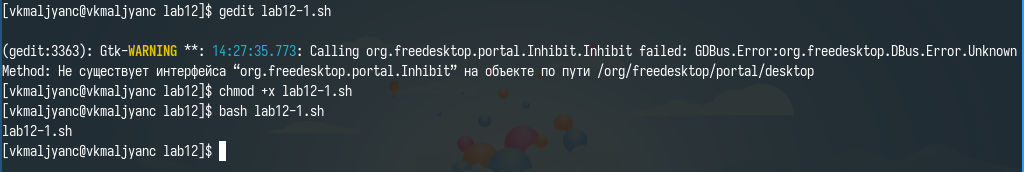


Рис. 3: Добавление права на исполнение и исполнение файла

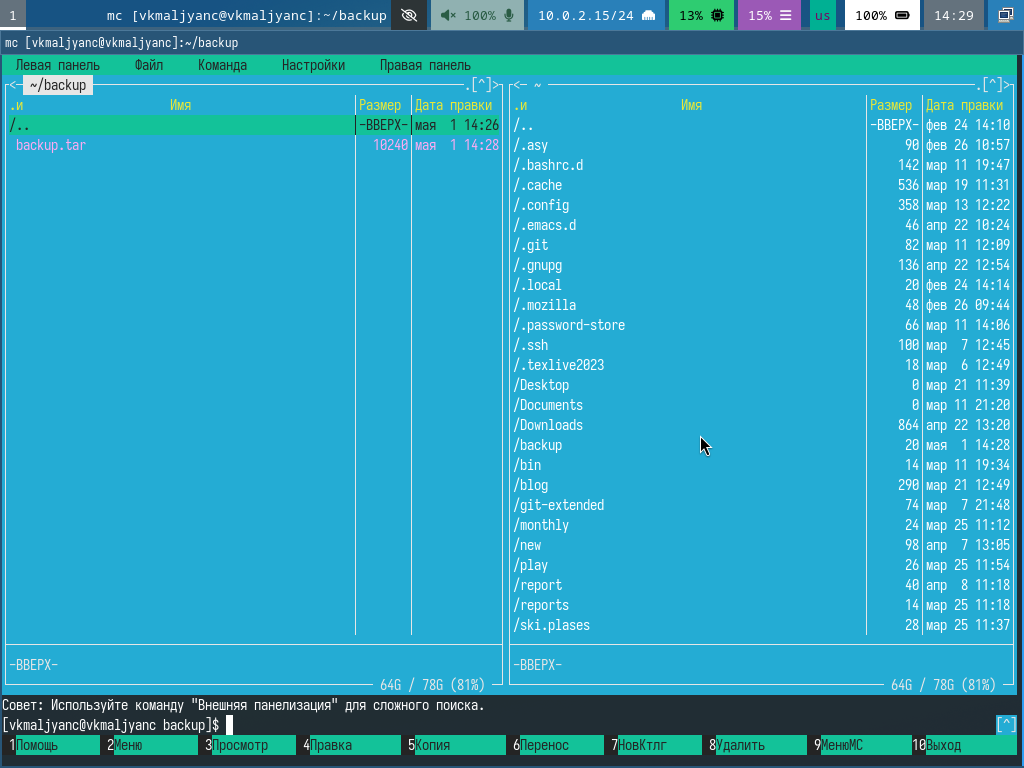


Рис. 4: Проверка корректности работы исполяемого файла

Пишу пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов (рис. 5) (рис. 6) (рис. 7).

Cоздание файла lab12-2.sh

Рис. 5: Cоздание файла lab12-2.sh

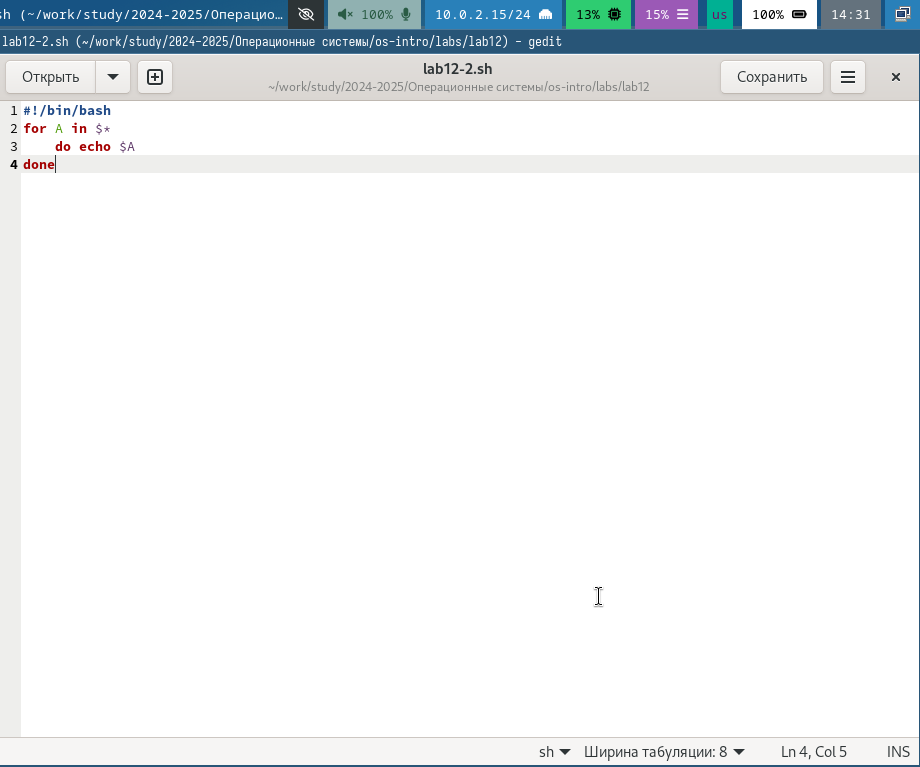


Рис. 6: Редактирование файла

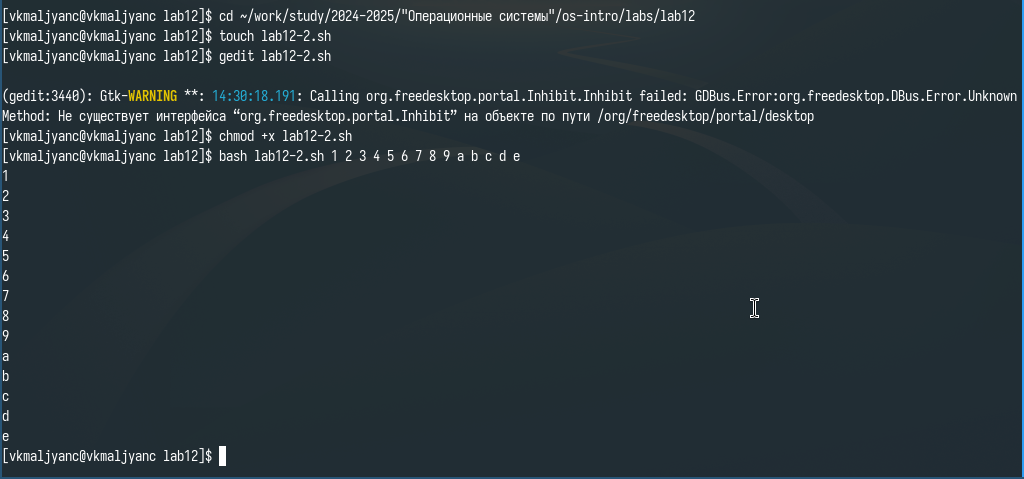


Рис. 7: Добавление права на исполнение и исполнение файла

Пишу командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога (рис. 8) (рис. 9) (рис. 10).

Cоздание файла lab12-3.sh

Рис. 8: Cоздание файла lab12-3.sh

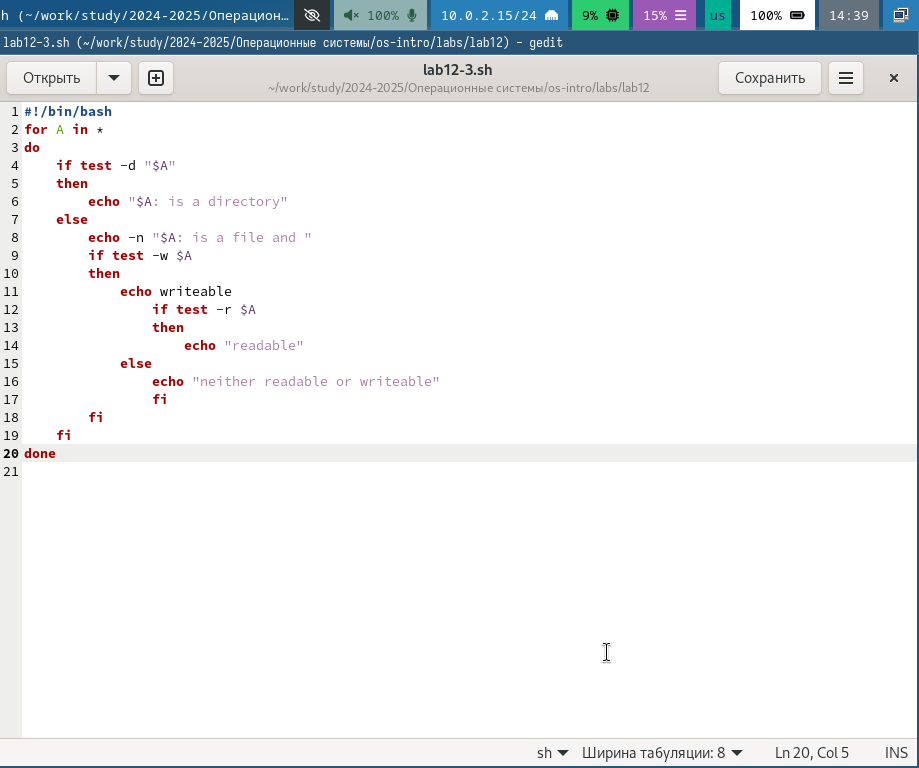


Рис. 9: Редактирование файла

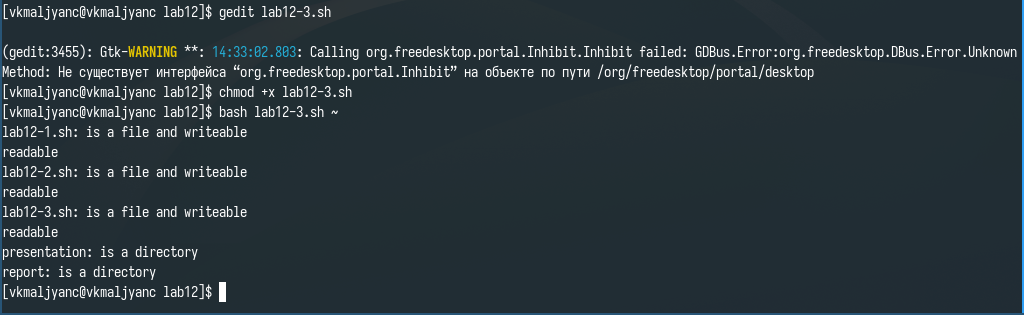


Рис. 10: Добавление права на исполнение и исполнение файла

Пишу 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 командной строки (рис. 11) (рис. 12) (рис. 13) [1].

Cоздание файла lab12-4.sh

Рис. 11: Cоздание файла lab12-4.sh

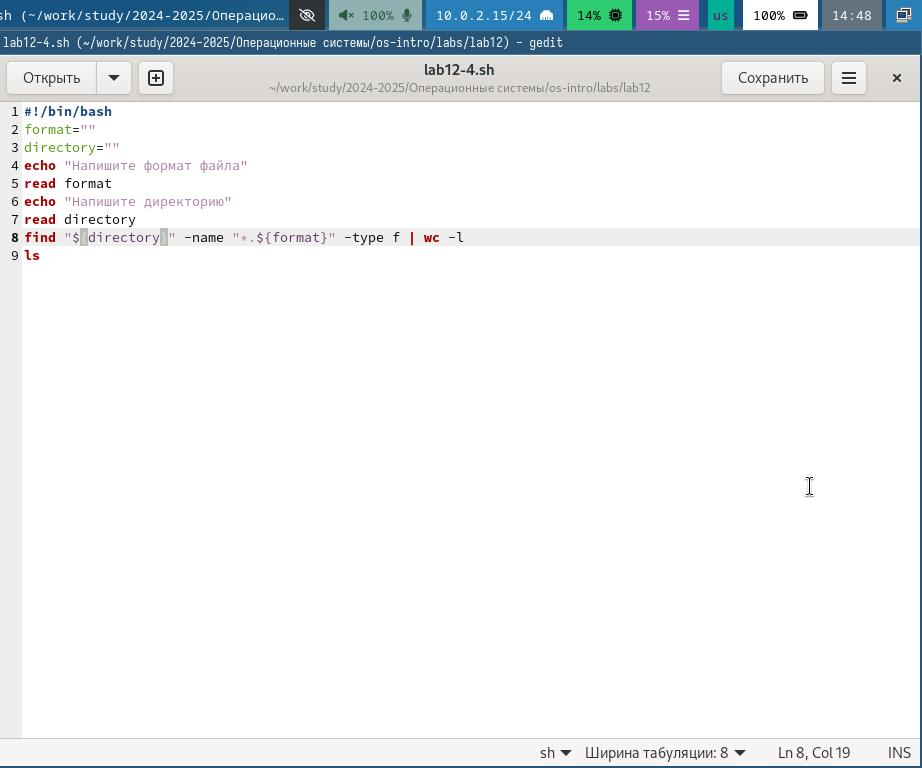


Рис. 12: Редактирование файла

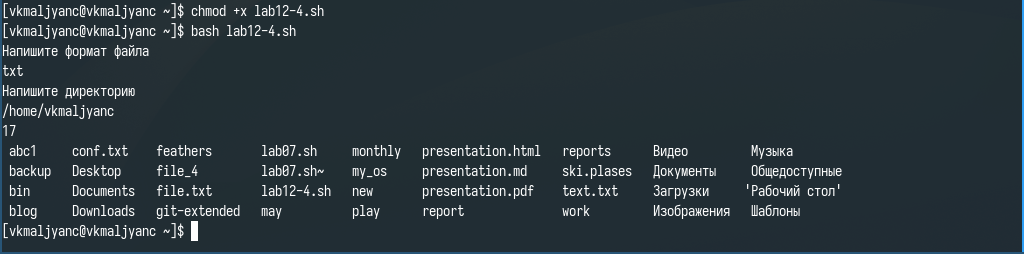


Рис. 13: Добавление права на исполнение и исполнение файла

# 4 Выводы

Я изучила основы программирования в оболочке ОС UNIX/Linux. Научилась писать небольшие командные файлы.

# 5 Контрольные вопросы

1. Командная оболочка - это программа, которая предоставляет интерфейс для взаимодействия пользователя с операционной системой. Пример командных оболочек: Bash, Zsh, Fish, Tcsh. Отличия: синтаксис и функциональность команд, некоторые оболочки имеют встроенные функции автозаполнения и подсказок, поддержка скриптов и расширяемость.
2. POSIX - это стандарт, разработанный для обеспечения совместимости операционных систем с UNIX.
3. Переменные определяются с помощью синтаксиса имя=значение. Массивы определяются с помощью синтаксиса имя=(значение1, значение2, …).
4. let используется для выполнения арифметических операций, read используется для чтения ввода от пользователя.
5. Сложение +, вычитание -, умножение \*, деление /, остаток от деления %.
6. Операция (( )) используется для выполнения арифметических операций и условий.
7. $HOME (домашний каталог пользователя), $USER (имя текущего пользователя), $PATH (переменная, содержащая пути к исполняемым файлам), $PWD (текущий рабочий каталог).
8. Метасимволы - это специальные символы, которые имеют особое значение в командной оболочке, например: \* - соответствует любому количеству символов, ? - соответствует любому одному символу, [] - соответствует любому символу из заданного набора.
9. Метасимволы можно экранизировать с помощью обратной косой черты .
10. Чтобы создать командный файл, нужно: открыть его в текстовом редакторе, начало файла должно начинаться с шебанга, затем для запуска скрипта сделать файл исполняемым.
11. f() {}.
12. Чтобы понять, является файл каталогом или обычным файлом, можно использовать команду test или [].
13. set - используется для установки параметров оболочки и переменных. typeset - используется для объявления переменных и их атрибутов. unset - используется для удаления переменной или функции.
14. Параметры передаются в командный файл через позиционные параметры. Внутри скрипта можно получить доступ к параметрам через $1, $2 и так далее.
15. Специальные переменные языка bash: %? - код завершения последней выполненной команды, $$ - PID (идентификатор процесса) текущего скрипта, $# - количество переданных параметров, $@ - все переданные параметры, $\* - все переданные параметры как одна строка.

# Список литературы

1. [Лабораторная работа № 12](https://esystem.rudn.ru/course/view.php?id=113).