Лабораторная работа №12

Программирование в командном процессоре ОС UNIX. Командные файлы

Полякова Юлия Александровна

Содержание

# 1 Цель работы

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

# 2 Задание

1. Ознакомиться с теоретическим материалом.
2. Написать 4 командных файла по заданию.
3. Ответить на контрольные вопросы.

# 3 Выполнение лабораторной работы

1. Для начала командой touch создаем 4 файла с расширением .sh Рассмотрим первый скрипт. При запуске будет делает резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup. Архивируется архиватором zip. (рис. 1).

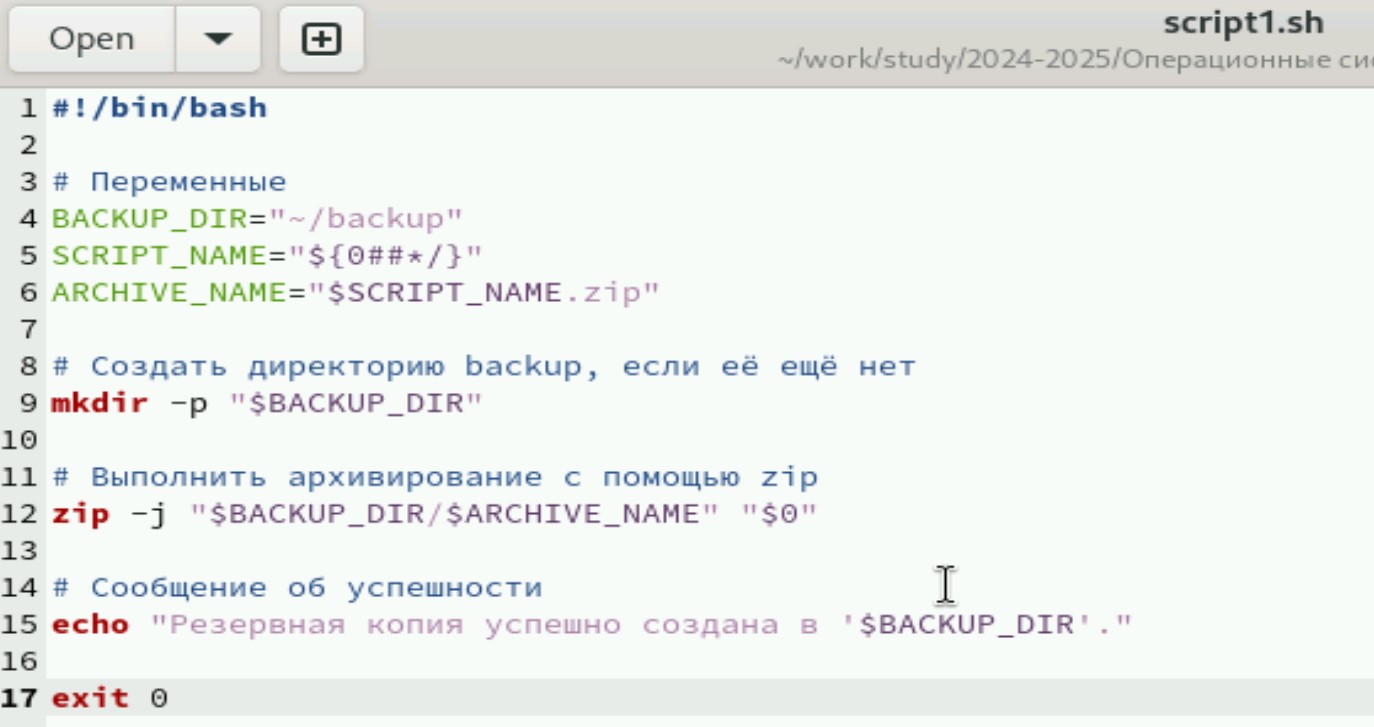


Рис. 1: Листинг script1.sh

1. Чтобы запустить файл нужно дать ему доступ на исполняемость. Как видно по сообщению, резервная копия успешно создалась. (рис. 2)

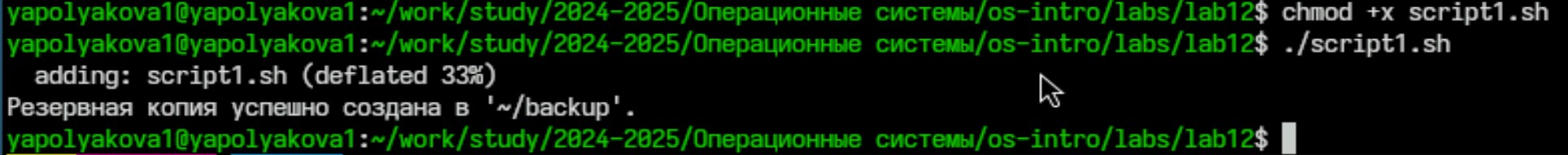


Рис. 2: Запуск script1.sh

1. Файл сархивировался в этот каталог. (рис. 3)

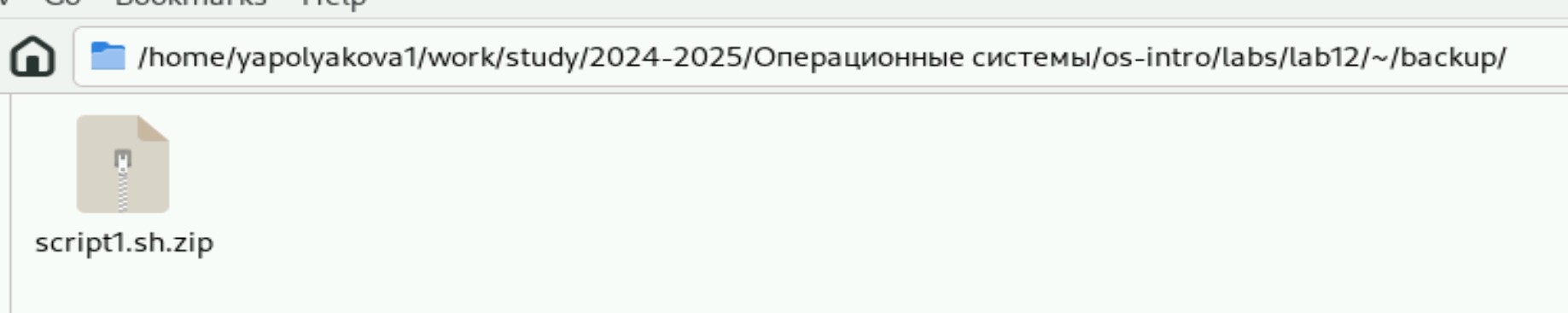


Рис. 3: Архив в backup

1. Рассмотрим второй скрипт. Обрабатывает любое произвольное число аргументов командной строки, в том числе превышающее десять. Последовательно распечатывает значения всех переданных аргументов в цикле (рис. 4)

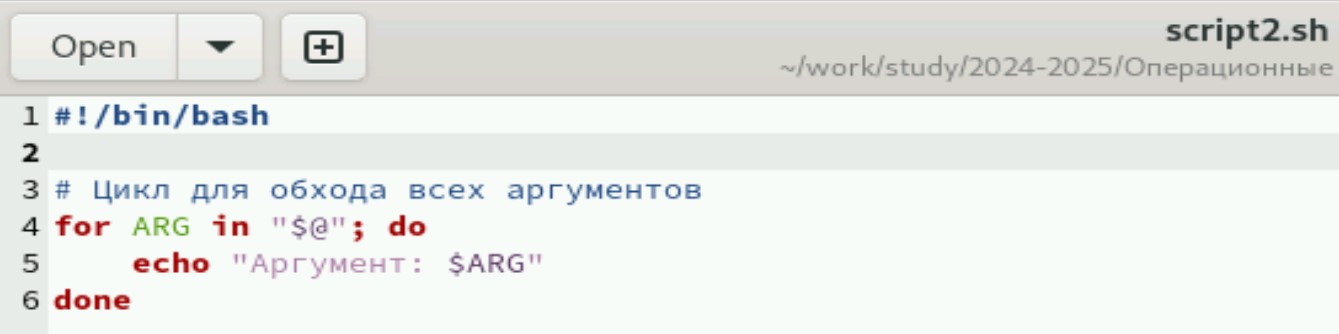


Рис. 4: Листинг script2.sh

1. Даем доступ на исполняемость и запускаем с большим кол-вом аргументов (рис. 5)



Рис. 5: Запуск script2.sh

1. Рассмотрим третий скрипт. Это командный файл — аналог команды ls (без использования самой этой команды и команды dir). Выдает информацию о нужном каталоге и выводит информацию о возможностях доступа к файлам этого каталога (рис. 6)

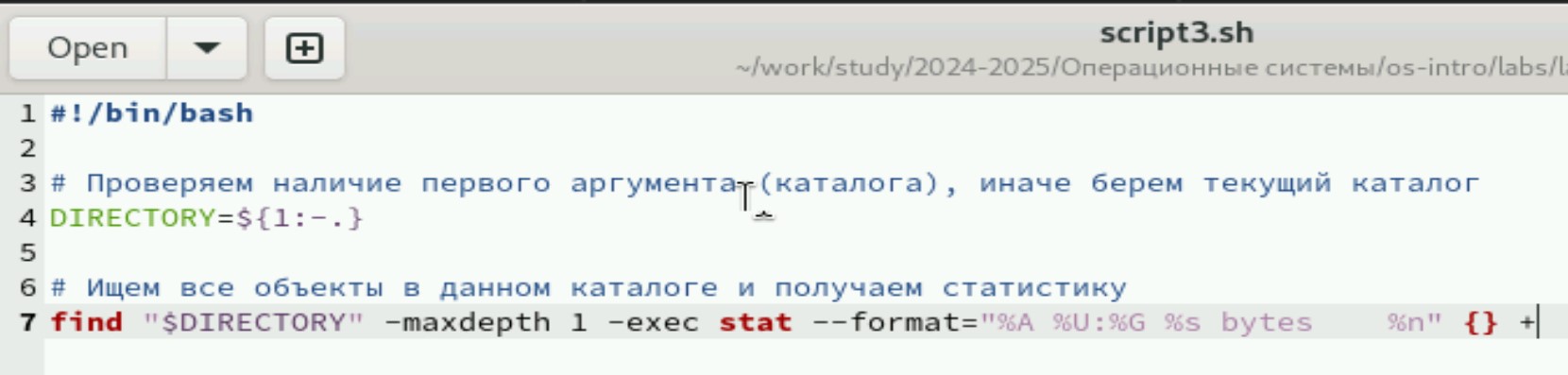


Рис. 6: Листинг script3.sh

1. Даем доступ к исполнению. Если не указать аргумент, то выведутся данные текущего каталога, если указать, то указанного (рис. 7)

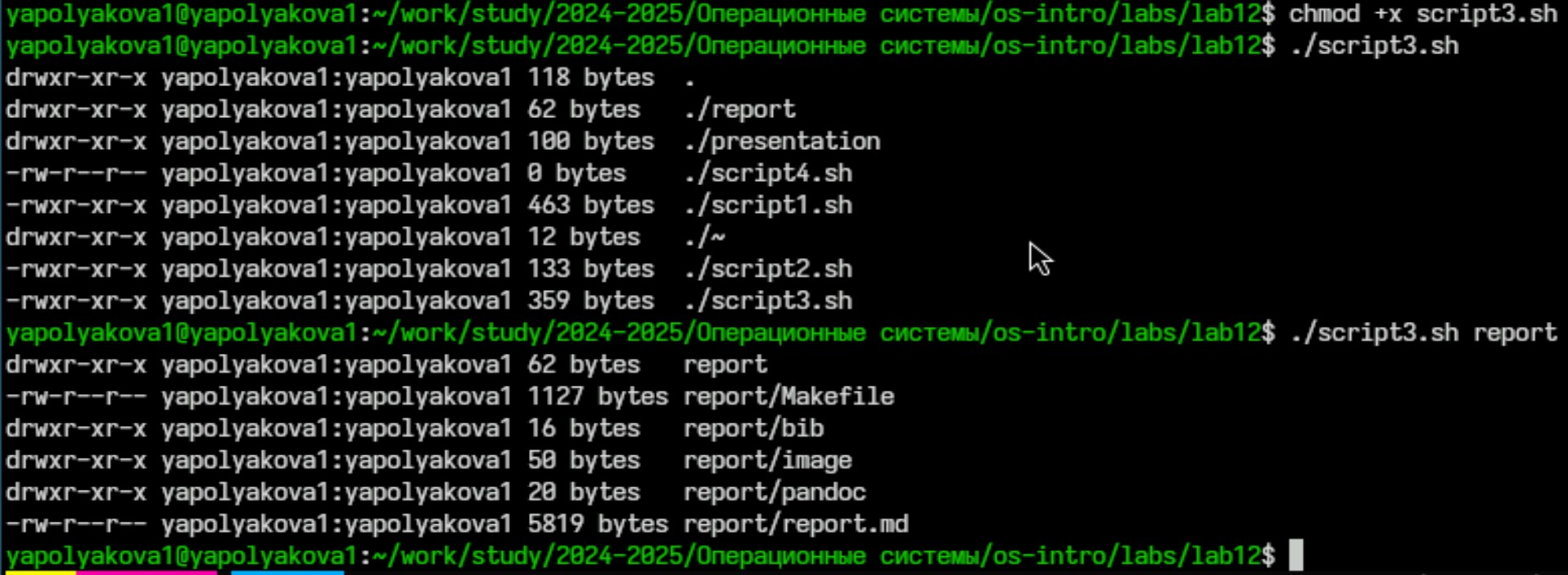


Рис. 7: Запуск script3.sh

1. Рассмотрим четвертый скрипт. Командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д., но сразу с точкой) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки. Также я решила добавить проверку на наличие аргументов. (рис. 8)

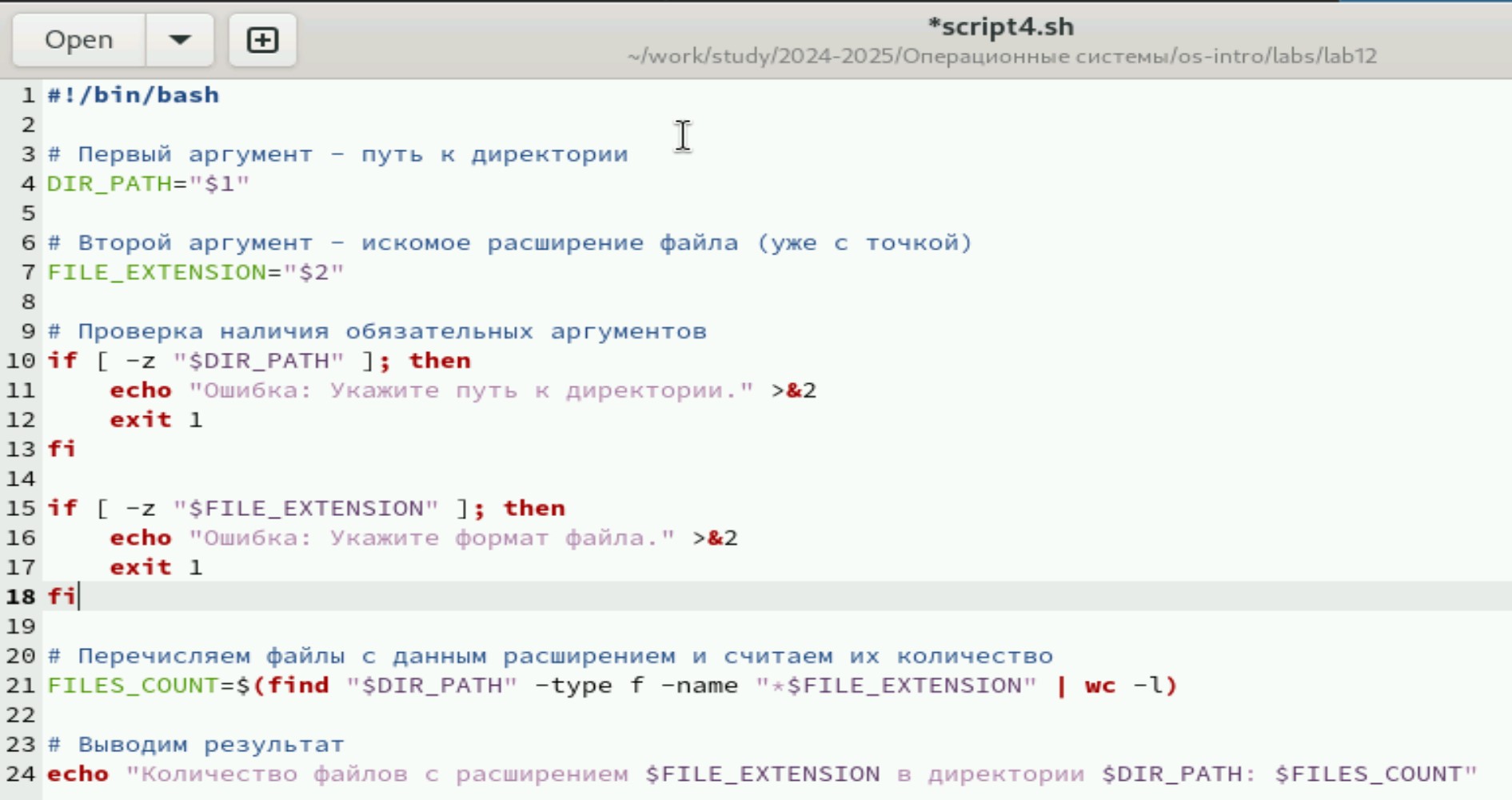


Рис. 8: Листинг script4.sh

1. Даем возможность исполнения, запускаем с различными форматами и каталогами. (рис. 9)

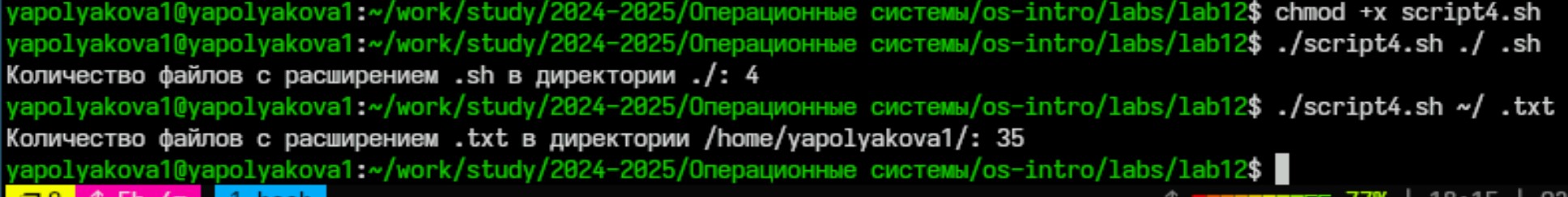


Рис. 9: Запуск script4.sh

# 4 Контрольные вопросы

1. Объясните понятие командной оболочки. Приведите примеры командных оболочек. Чем они отличаются?

Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек:

* оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
* С-оболочка (или csh) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
* оболочка Корна (или ksh) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна;
* BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек С и Корна (разработка компании Free Software Foundation).

Основные отличия:

* Синтаксис и особенности: Каждая оболочка имеет уникальный набор возможностей и особенностей, такие как поддержка встроенных функций, автозавершение команд, управление историей команд и другие.
* Совместимость: Некоторые оболочки строго следуют стандартам POSIX, тогда как другие добавляют собственные расширения.
* Производительность: Различные оболочки могут иметь разную производительность при выполнении определённых операций

1. Что такое POSIX?

POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна.

1. Как определяются переменные и массивы в языке программирования bash?

* Переменная определяется следующим образом: variable=value
* Массив объявляется следующим образом: array=(value1 value2 …)
* Доступ к элементам массива осуществляется через индекс: echo ${array[index]}

1. Каково назначение операторов let и read?

Оператор let используется для вычисления выражений и присваивания результата переменной:

let variable=expression или (( variable = expression ))

Оператор read предназначен для чтения строки из стандартного ввода и сохранения её в переменную: read variable

1. Какие арифметические операции можно применять в языке программирования bash?

* сложение (+)
* вычитание (-)
* умножение (\*)
* деление (/)
* остаток от деления (%)
* возведение в степень (\*\*)

Пример: result=$((a + b))

1. Что означает операция (( ))?

Операция (( )) позволяет выполнять арифметическое выражение внутри скобок и автоматически возвращает результат.

Например: (( result = a \* b ))

Это эквивалентно использованию оператора let, но удобнее и нагляднее.

1. Какие стандартные имена переменных Вам известны?

* $HOME: домашний каталог текущего пользователя.
* $PATH: список путей для поиска исполняемых файлов.
* $USER: имя текущего пользователя.
* $SHELL: используемая командная оболочка.
* $PWD: текущий рабочий каталог.
* $PS1: строка приглашения.

1. Что такое метасимволы?

Метасимволы — это символы, имеющие специальное значение в командной строке. Они используются для обозначения шаблонов имен файлов, перенаправлений ввода-вывода и специальных действий. Примеры метасимволов:

* \*
* ?
* [ ]
* < > | & ;

1. Как экранировать метасимволы?

Экранирование выполняется с использованием обратного слэша () или двойных кавычек (“). Пример:

ls file\\* # выводит все файлы начинающиеся с ‘file’

1. Как создавать и запускать командные файлы?

Сначала нужно создать файл с расширением .sh, затем добавить код и сделать файл исполняемым:

chmod +x script.sh

Чтобы запустить из текущего каталога: ./script.sh, если в другом, то полный путь.

1. Как определяются функции в языке программирования bash?

function\_name() {  
 commands  
}

Пример:

hello\_world() {  
 echo "Hello World!"  
}

1. Каким образом можно выяснить, является файл каталогом или обычным файлом?

Использовать команду test или оператор [[ ]]. Например:

if [[ -d "$filename" ]]; then  
 echo "Файл является каталогом"  
fi

1. Каково назначение команд set, typeset и unset?

* Команда set устанавливает значения глобальных переменных окружения.
* Команда typeset (аналогична declare) создаёт локальные переменные и задаёт атрибуты переменным.
* Команда unset удаляет переменные или функции.

1. Как передаются параметры в командные файлы?

Параметры передаются следующим образом:

./script.sh arg1 arg2 …

Доступ к параметрам осуществляется через позиционные переменные: $1, $2, …, $n

1. Назовите специальные переменные языка bash и их назначение.

* $#: количество аргументов.
* $@: аргументы команды как отдельные элементы.
* $\*: аргументы команды как одна строка.
* $$: PID процесса оболочки.
* $!: PID последнего запущенного фона.
* $?: статус завершения последней выполненной команды.

# 5 Вывод

Были изучены основы программирования в оболочке ОС UNIX/Linux. Также мы научились писать небольшие командные файлы.