Лабораторная работа 12

Операционные системы

Богданюк Анна Васильевна

Содержание

# 1 Цель работы

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

# 2 Задание

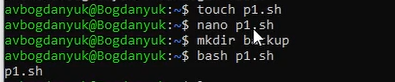
1. Написать скрипт, который при запуске будет делать резервную копию самого себя
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки
3. Написать командный файл — аналог команды ls
4. Написать командный файл, который получает в качестве аргумента командной строки формат файла

# 3 Теоретическое введение

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). 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. Рассмотрим основные элементы программирования в оболочке bash. В других оболочках большинство команд будет совпадать с описанными ниже.

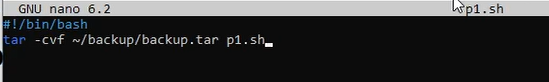
# 4 Выполнение лабораторной работы

Для начала создаю файл p1.sh и делаю его исполняемым. Выполняю его (рис. 1).



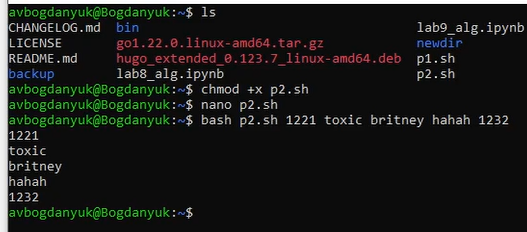
Создание файла и вывод

Пишу скрипт, который при запуске будет делать резервную копию самого себя в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar.(рис. 2).



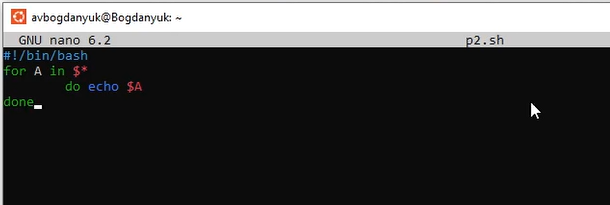
Скрипт

Для начала создаю файл p2.sh и делаю его исполняемым. Выполняю его (рис. 3).



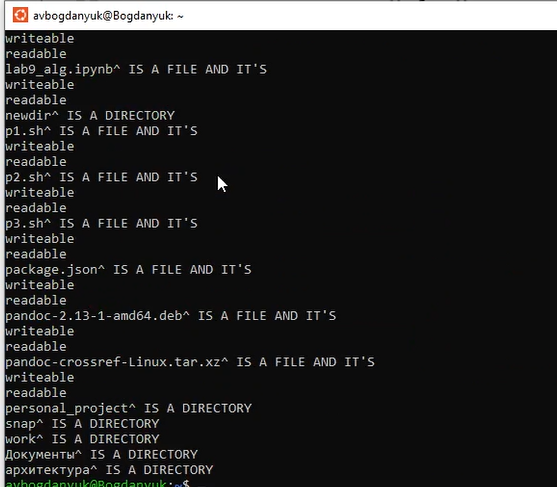
Создание файла и вывод

Пишу пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. (рис. 4).



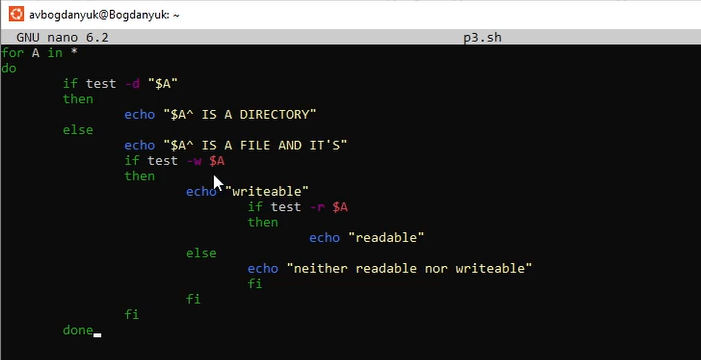
Скрипт

Для начала создаю файл p3.sh и делаю его исполняемым. Выполняю его (рис. 5).



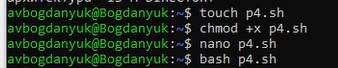
Создание файла и вывод

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. 6).



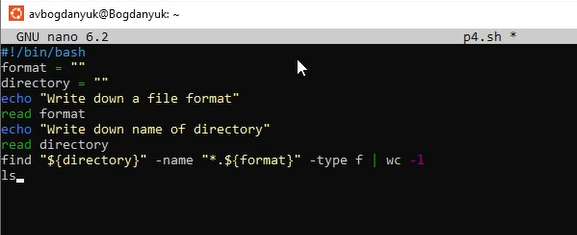
Скрипт

Для начала создаю файл p4.sh и делаю его исполняемым. Выполняю его (рис. 7).



Создание файла и вывод

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. 8).



Скрипт

# 5 Выводы

В ходе выполнения лабораторной работы были изучены основы программирования в оболочке ОС UNIX/Linux. Научилась писать небольшие командные файлы.