Отчёт по лабораторной работе №12

Операционные системы

Бережной Иван Александрович

Содержание

# 1 Цель работы

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

# 2 Задание

1. Написать скрипт, создающий резервную копию самого себя
2. Написать командный файл, выводящий в терминал введённые строки
3. Написать аналог команды ls
4. Написать командный файл, вычисляющий количество файлов с указанным расширением

# 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.

# 4 Выполнение лабораторной работы

## 4.1 Написание скрипта, создающего резервную копию самого себя

Для начала создадим файл и дадим ему разрешение на исполнение (рис. 1). Откроем его и пропишем логику выполнения (рис. 2). В результате выполнения мы получим архив с копией нужного нам файла.

Создание первого файла

Рис. 1: Создание первого файла

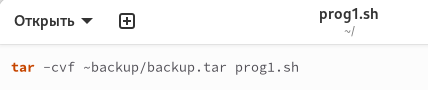


Рис. 2: Написание первого файла

## 4.2 Написание командного файла, выводящего в терминал введённые строки

Создадим новый файл (рис. 3) и напишем скрипт (рис. 4) - цикл, который проходит по каждому введённому значению и выводит его на экран (рис. 5).

Создание второго файал

Рис. 3: Создание второго файал



Рис. 4: Написание второго файла

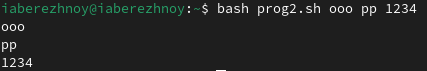


Рис. 5: Результат работы второго файла

## 4.3 Написание аналога команды ls

Снова создаём файл (рис. 6) и пишем в нём скрипт (рис. 7), который будет давать информацию о нужном каталоге и выводить информацию о возможностях доступа к его файлам (рис. 8).

Создание третьего файла

Рис. 6: Создание третьего файла

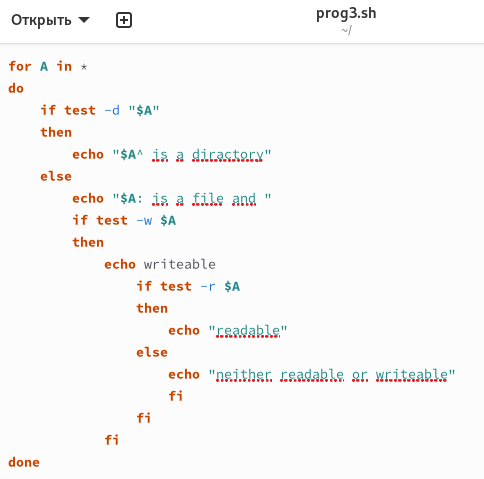


Рис. 7: Написание третьего файла

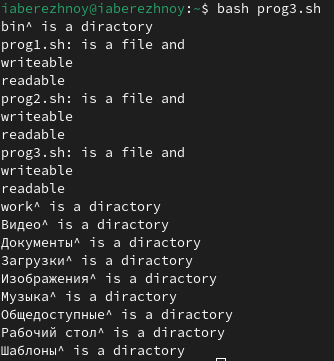


Рис. 8: Результат работы третьего файла

## 4.4 Написание командного файла, вычисляющего количество файлов с указанным расширением

Создадим последний на сегодня файл (рис. 9) и напишем логику (рис. 10). Он будет вычислять количество файлов с указанным расширением внутри указанного каталога (рис. 11).

Создание четвёртого файла

Рис. 9: Создание четвёртого файла



Рис. 10: Написание четвёртого файла

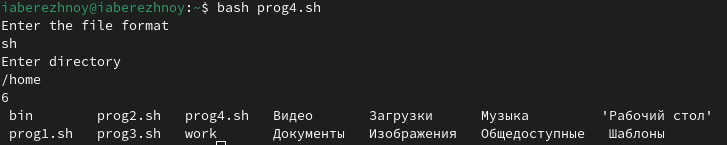


Рис. 11: Результат работы четвёртого файла

# 5 Выводы

В ходе выполнения лабораторной работы мы изучили основы программирования в оболочке ОС UNIX/Linux и научились писать небольшие командные файлы.

# Список литературы