Лабораторная работа №11

Операционные системы”

Александрова Ульяна

Содержание

# 1 Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 2 Задание

1. Используя команды getopts grep, написать командный файл, который анализирует командную строку с ключами: – -iinputfile — прочитать данные из указанного файла; – -ooutputfile — вывести данные в указанный файл; – -pшаблон — указать шаблон для поиска; – -C — различать большие и малые буквы; – -n — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом -p.
2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции exit(n), передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды $?, выдать сообщение о том, какое число было введено.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до 𝑁 (например 1.tmp, 2.tmp, 3.tmp,4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
4. Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find).

# 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

Создала файл app1.sh для первой программы и дополнительные текстовые файлы (рис. fig. 1).

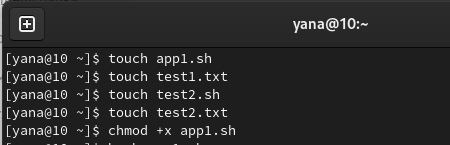


Рис. 1: Создание файла

Заполнила код (рис. fig. 2).

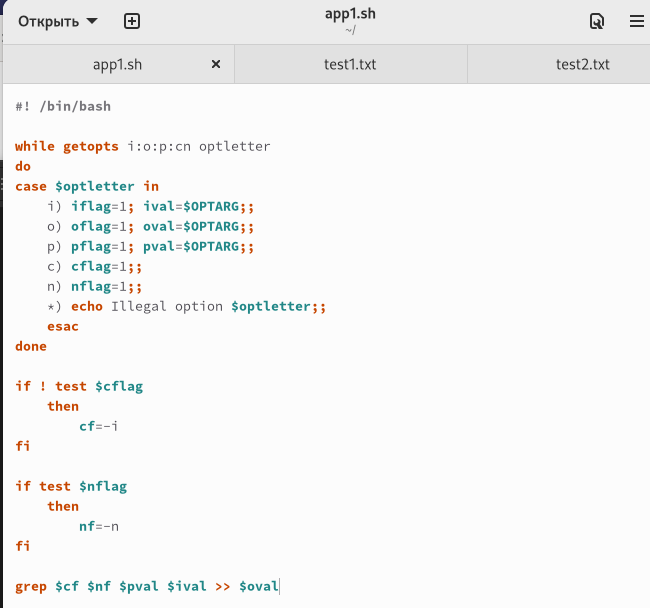


Рис. 2: Код 1ой программы

Запустила программу (рис. fig. 3). Она работает (рис. fig. 4), (рис. fig. 5).

Рис. 3: Запуск 1ой программы

Рис. 3: Запуск 1ой программы

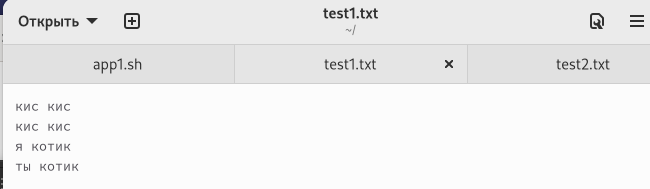


Рис. 4: Текстовый файл 1

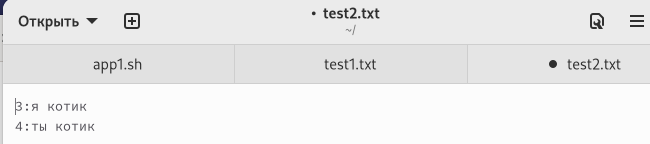


Рис. 5: Текстовый файл 2

Я создала второй командный файл, а также дополнительный файл (рис. fig. 6).



Рис. 6: Создание файла 2

Далее я написала тело обеих программ (рис. fig. 7), (рис. fig. 8).

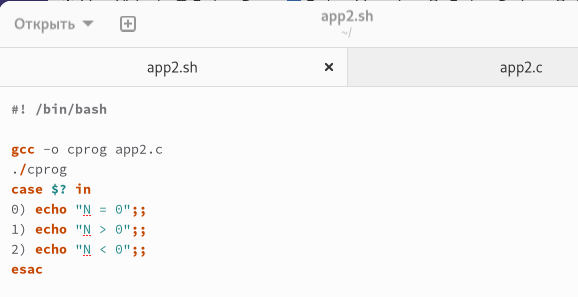


Рис. 7: Тело 1



Рис. 8: Тело 2

Проверила работу приложения (рис. fig. 9).



Рис. 9: Проверка работы 2

Создала третий файл app3.sh (рис. fig. 10).

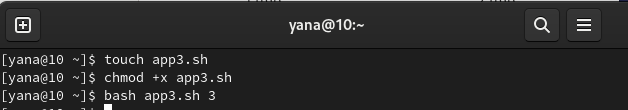


Рис. 10: Создание файла 3

Записала код программы (рис. fig. 11).

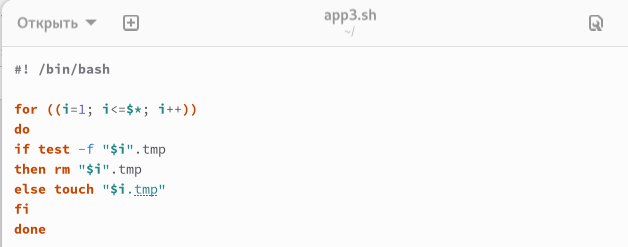


Рис. 11: Код программы

Проверила работу кода. Новые файлы создались (рис. fig. 12).

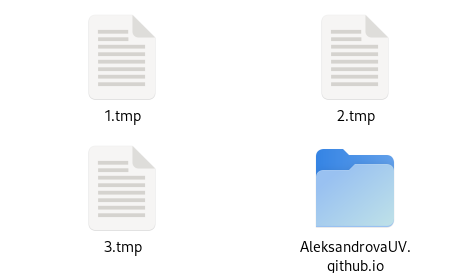


Рис. 12: Проверка программы

Я создала последний командный файл app4.sh и заполнила файл (рис. fig. 13).

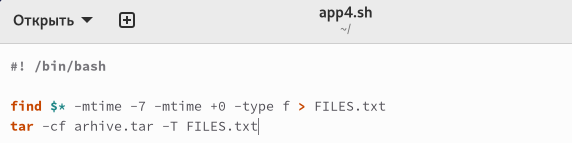


Рис. 13: Создание файла 4

Далее запустила приложение. Оно работает исправно (рис. fig. 14).

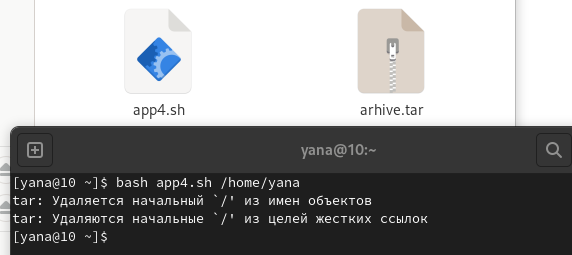


Рис. 14: Проверка программы

# 5 Выводы

Я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

:::