Отчет о выполнении лабораторной работы

Лабораторная работа №13

Филипьева Ксения Дмитриевна

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. Команд- ный 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 ко- мандный 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 Выполнение лабораторной работы

Создадим файл для первого задания (рис. 1).

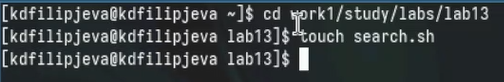


Рис. 1: создание файла

Введем в него код для первого задания (рис. 2).

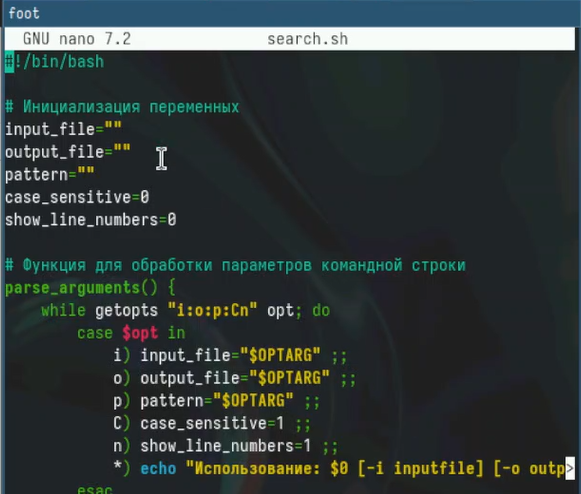


Рис. 2: ввод кода

Создадим файл в который будем вводить текст, с которым будет работать программа (рис. 3).

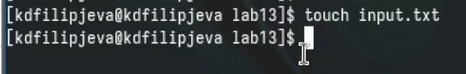


Рис. 3: создание файла

Текст для работы программы (рис. 4).

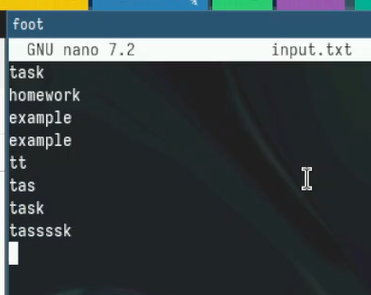


Рис. 4: текст в файле

Вывод найденного текста по установленному шаблону в командную строку (рис. 5).

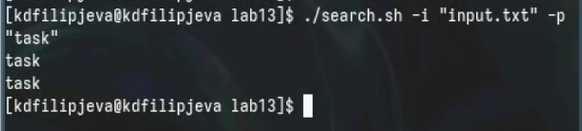


Рис. 5: вывод результата

Выведем найденный текст в отдельный файл (рис. 6).

вывод результата

Рис. 6: вывод результата

Выведенный текст в отдельном файле (рис. 7).

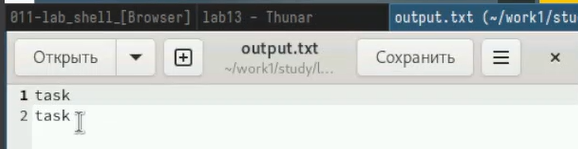


Рис. 7: вывод результата

Отредактируем текст для работы программы (рис. 8).

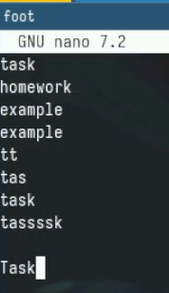


Рис. 8: исправленный текст

Выведем текст в файл с учетом регистра (рис. 9).

вывод результата

Рис. 9: вывод результата

Вывод текста с учетом регистра( видно, что вывело только с маленькой буквы, а заглавную не тронуло) (рис. 10).

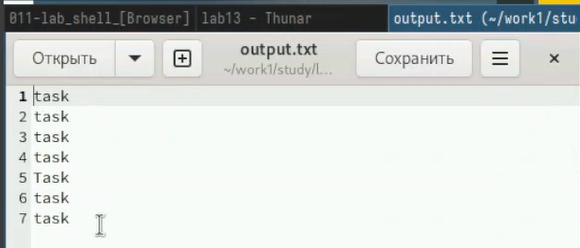


Рис. 10: вывод результата

Выведем текст с учетом регистра и нумерацией строк, из которых было взято слово (рис. **¿fig:11?**).

![вывод результата(image/1311.png){#fig:11 width=100%}

Вывод текста с нумерацией (рис. 11).

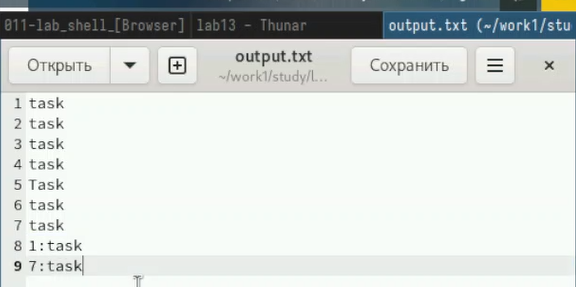


Рис. 11: вывод результата

Создадим файл для второго задания и выдадим права на выполнение (рис. 12).

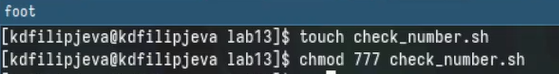


Рис. 12: создание файла

Вставим код программы, который ответственнен за определение числа (рис. 13).

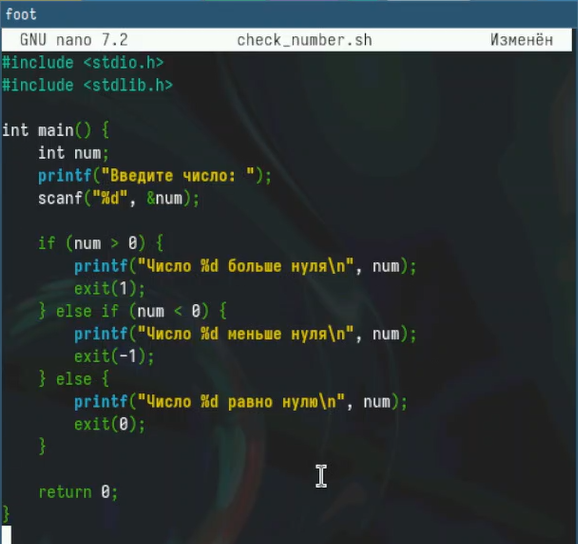


Рис. 13: код программы

Создадим файл для второго задания, который будет “общаться с пользователем” (рис. 14).

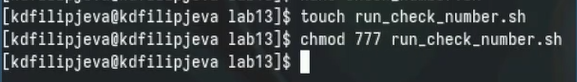


Рис. 14: создание файла

Вставим в него код программы (рис. 15).

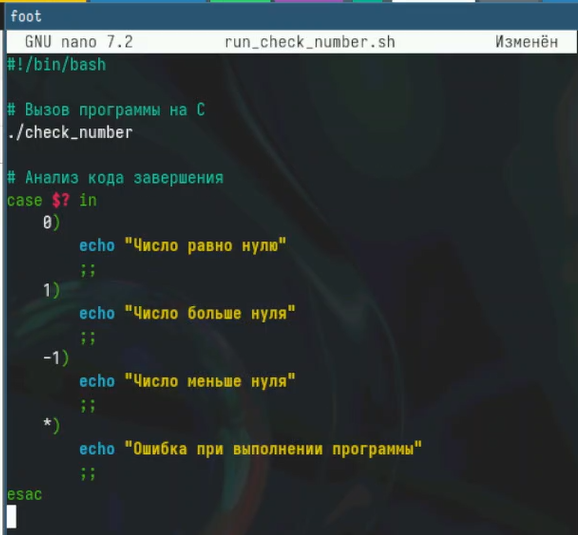


Рис. 15: код программы

Скомпилируем наш код на языке Си и проверим работоспособность (рис. 16).

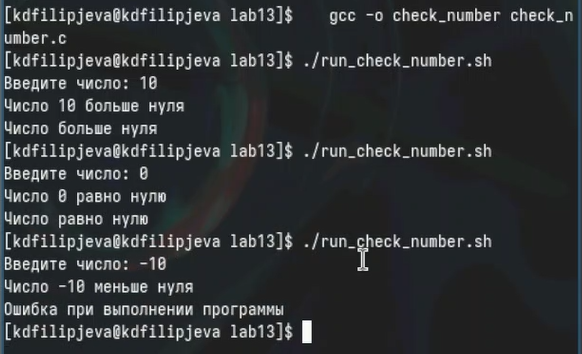


Рис. 16: вывод результата

Создадим файл для третьего задания и выдадим ему права на выполнение (рис. 17).

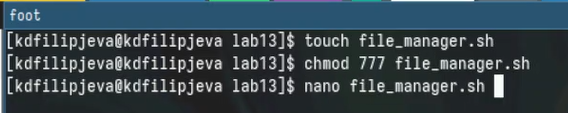


Рис. 17: создание файла

Вставим в него необходимый код для выполнения задания (рис. 18).

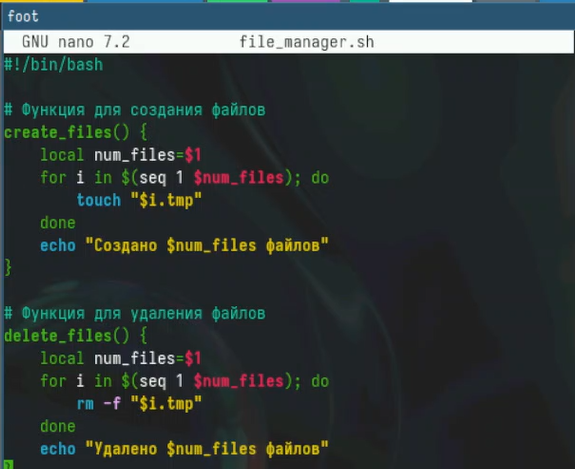


Рис. 18: вставка кода

Работоспособность кода (рис. 19).

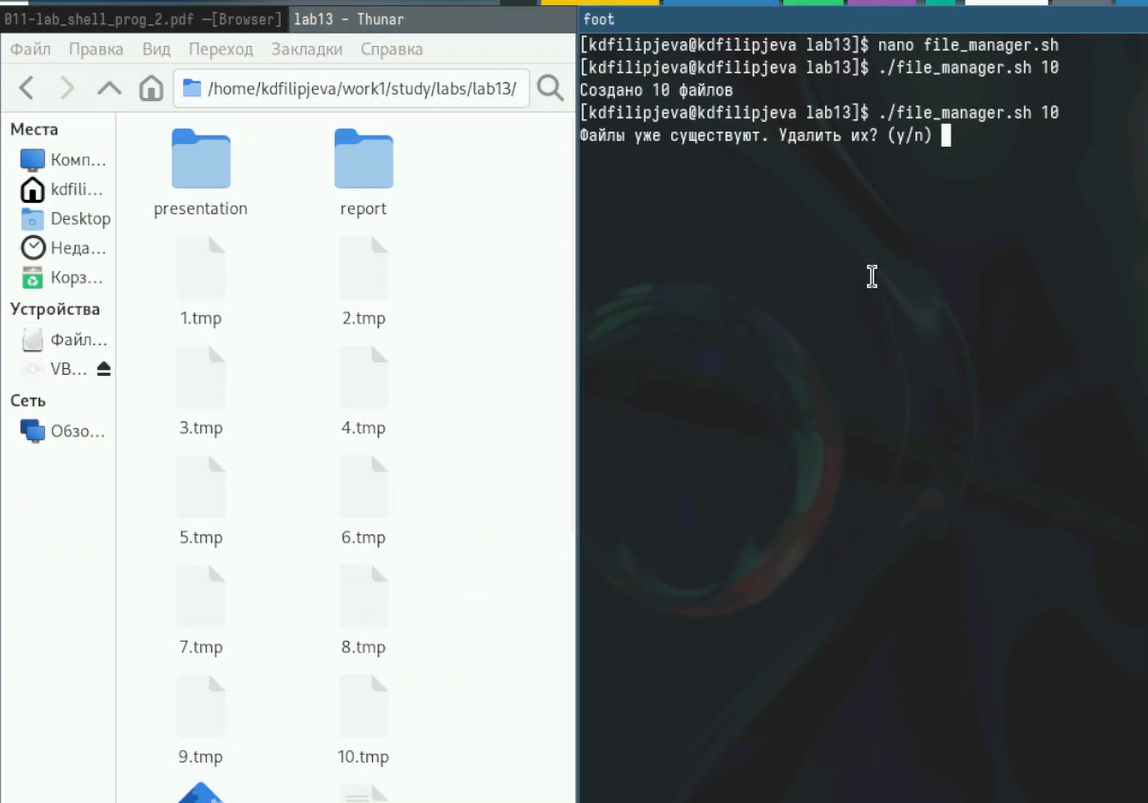


Рис. 19: вывод результата

Работоспособность кода в обратную сторону (рис. 20).

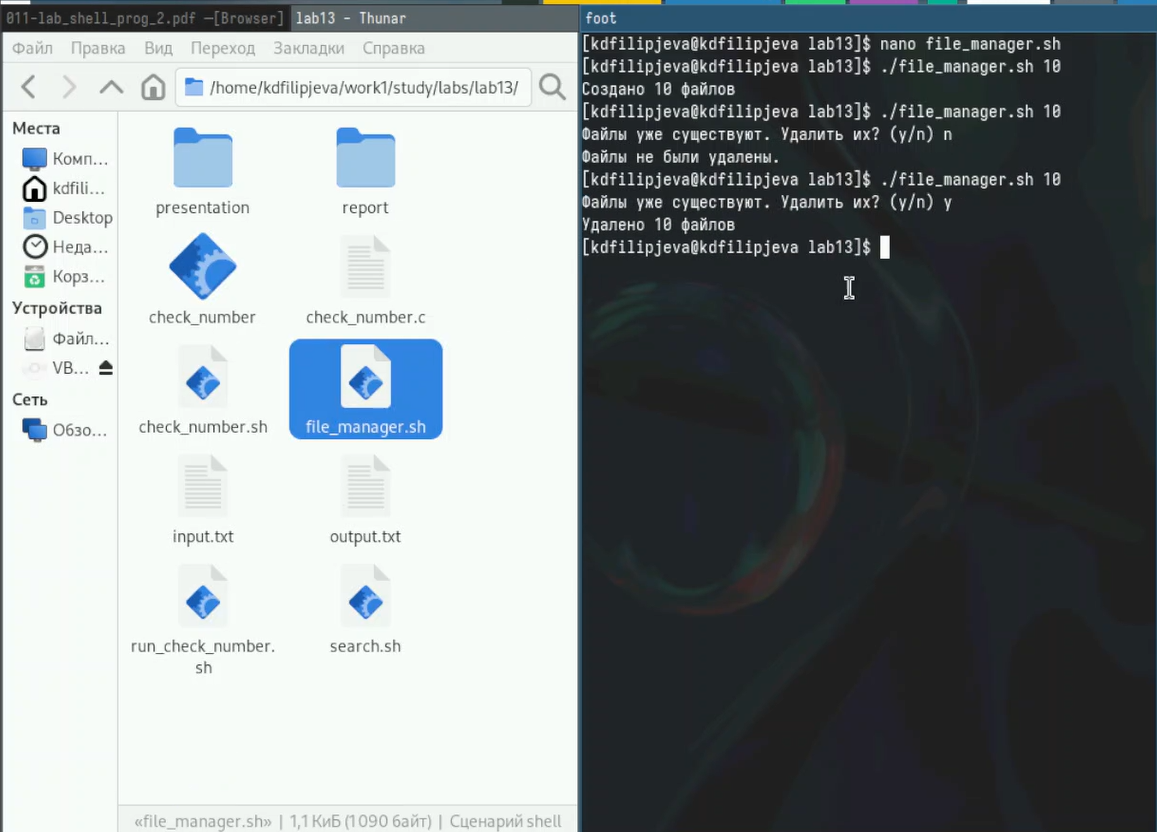


Рис. 20: вывод результата

Создадим файл для четвертого задания и выдадим ему права на выполнение (рис. 21).

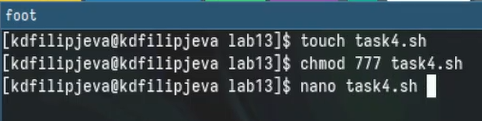


Рис. 21: создание файла

Вставим в него необходимый код программы (рис. 22).

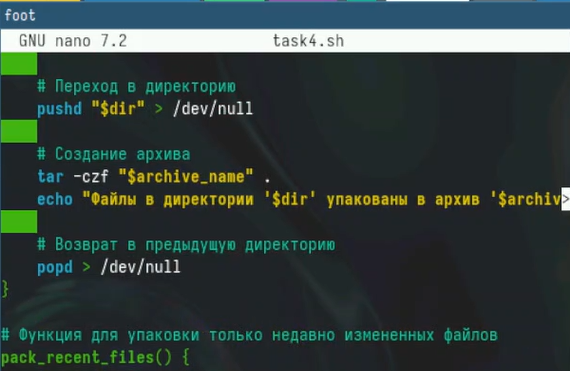


Рис. 22: код программы

Работоспособность кода (рис. 23).

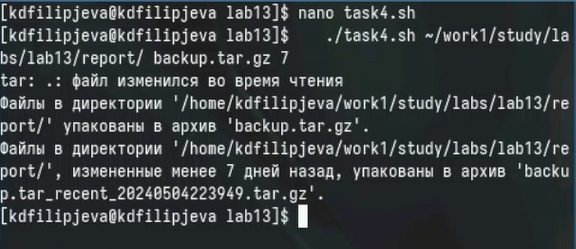


Рис. 23: вывод результата

Созданные 2 архива: всей папки и только файлов, которые были изменены менее чем неделю назад (рис. 24).

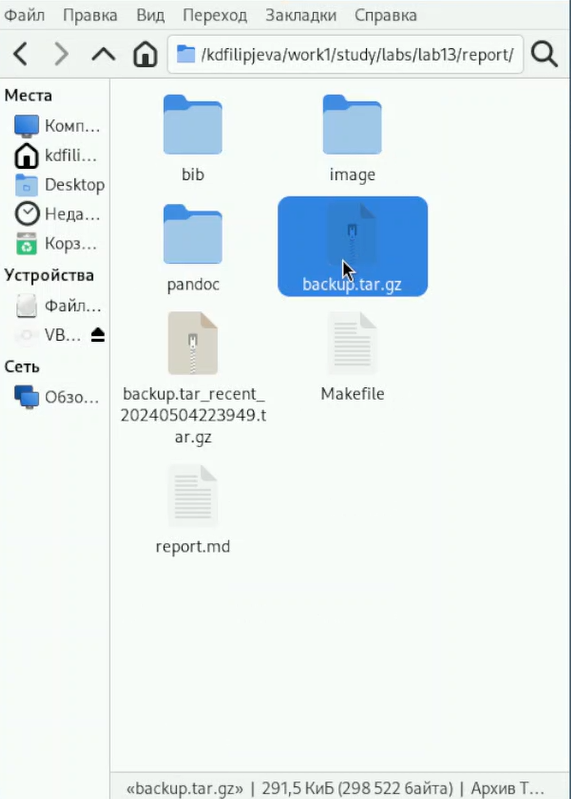


Рис. 24: вывод результата

# 4 Выводы

Мы получили новые и отработали уже имеющиеся навыки программирования в оболочке ОС Linux.

# 5 Ответы на вопросы

1. Команда getopts используется для разбора параметров командной строки в скриптах Bash. Она позволяет легко обрабатывать флаги и аргументы, переданные пользователем при запуске скрипта.
2. Метасимволы, такие как \*, ?, [], используются в командной строке для создания шаблонов имен файлов. Они позволяют быстро и гибко генерировать списки файлов, соответствующих определенным критериям.
3. Основные операторы управления действиями в Bash:
   * if-then-else-fi - для условного выполнения команд
   * case-esac - для многовариантного выбора
   * for-do-done - для итерации по списку значений
   * while-do-done - для выполнения команд, пока условие истинно
   * until-do-done - для выполнения команд, пока условие ложно
4. Для прерывания цикла в Bash используются:
   * break - для выхода из текущего цикла
   * continue - для перехода к следующей итерации цикла
5. Команды false и true возвращают соответственно ложное (1) и истинное (0) значение, которое можно использовать в управляющих конструкциях.
6. Строка if test -f mani.$s проверяет, существует ли файл с именем, сформированным из переменных $s и $i. Если файл существует (-f), то выполняются дальнейшие действия.
7. Конструкция while выполняет команды, пока условие истинно, а until выполняет команды, пока условие ложно. Таким образом, until можно рассматривать как инвертированный while.