Лабораторная работа-11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Кузнецов Алексей НБИбд-02-21

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 Задания

1. Используя команды getopts grep,написать командный файл,который анализирует командную строку с ключами: – -iinputfile —прочитатьданные 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. Написатьна языке Си программу,которая вводитчисло и определяет,являетсяли 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 отом,какое число было введено.
3. Написать командный файл,создающий указанное число файлов,пронумерованных последовательноот1до𝑁(например1.tmp,2.tmp,3.tmp,4.tmp ит.д.).Числофайлов, 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 запаковываетв архив все файлы в указанной директории.Модифицировать еготак,чтобы запаковывались только те файлы,которые были изменены менее недели тому назад (использовать команду find).

# 3 Выполнение лабораторной работы

1. Используя команды getopts grep, написал 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.(рис. 1)(рис. 2)(рис. 3)

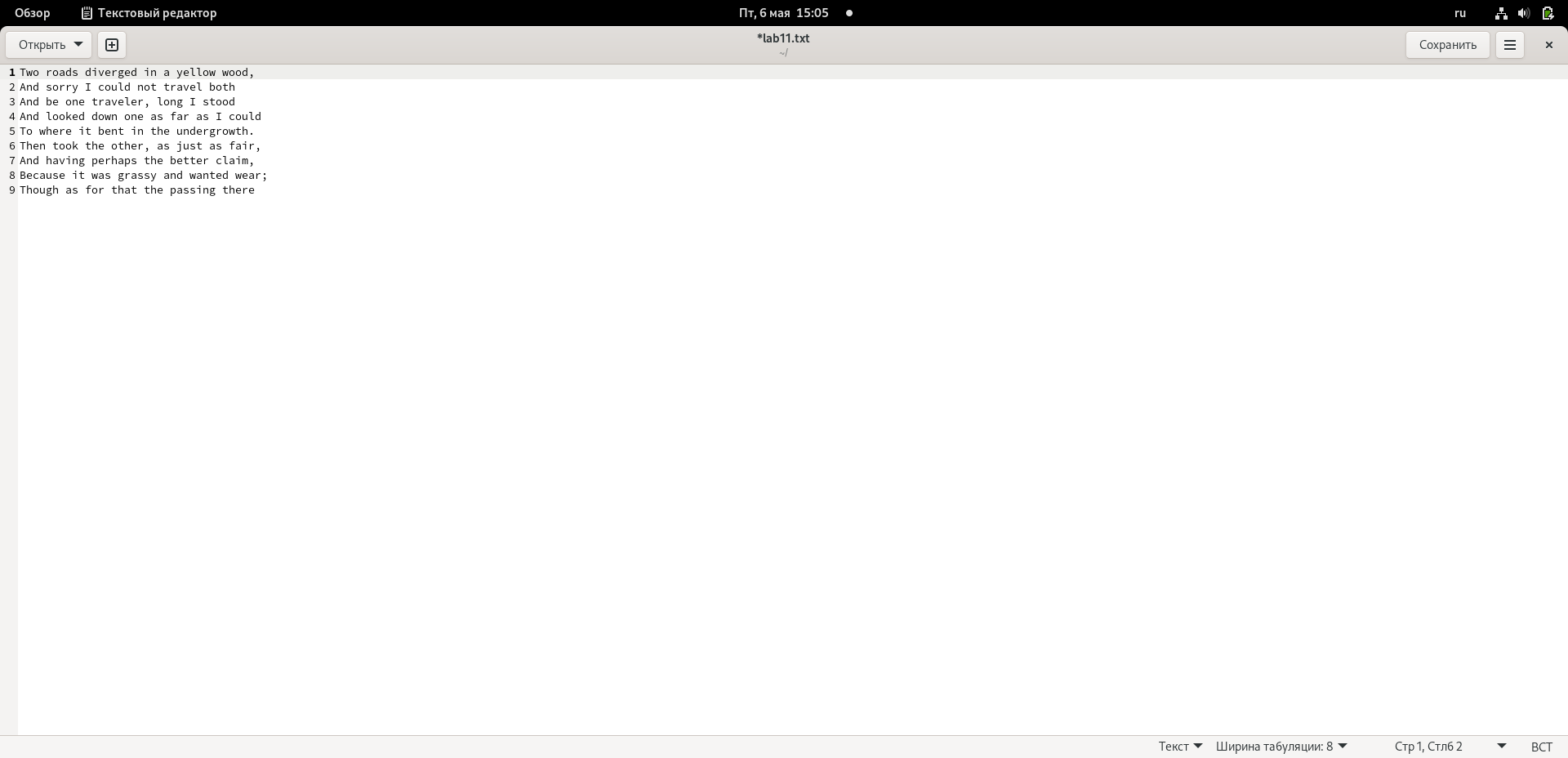


Рис. 1: Вставил в файл любой текст из интернета

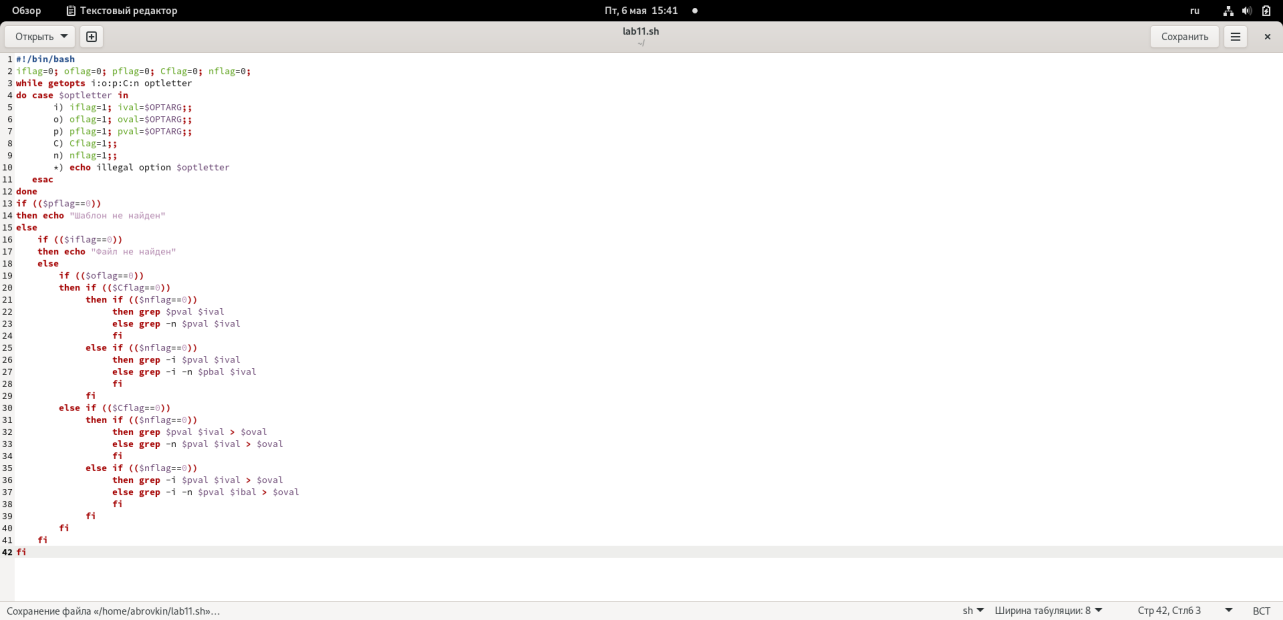


Рис. 2: Пишу первый скрипт

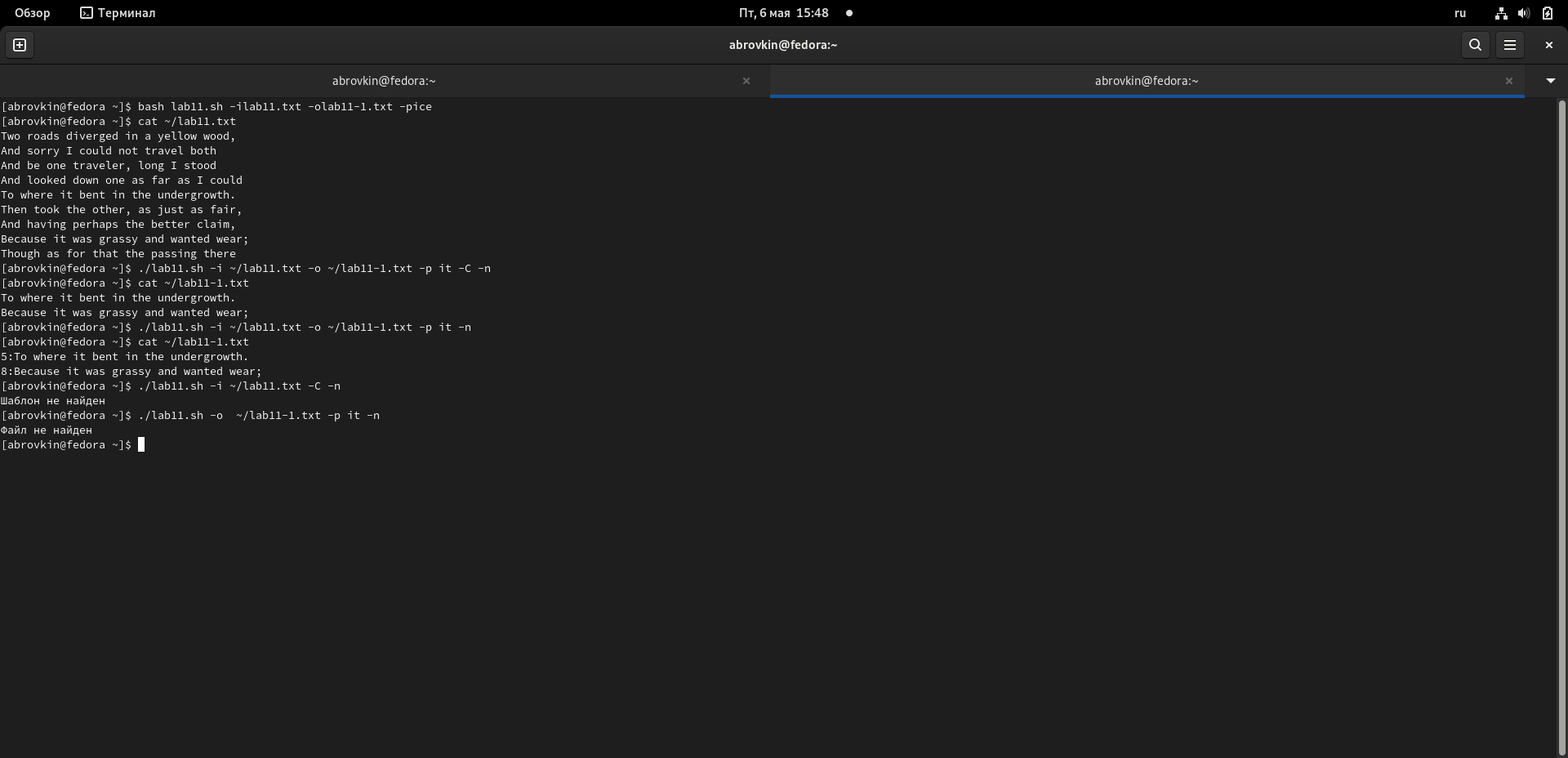


Рис. 3: Проверяю в терминале

1. Написал 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.(рис. 4)(рис. 5)(рис. 6)

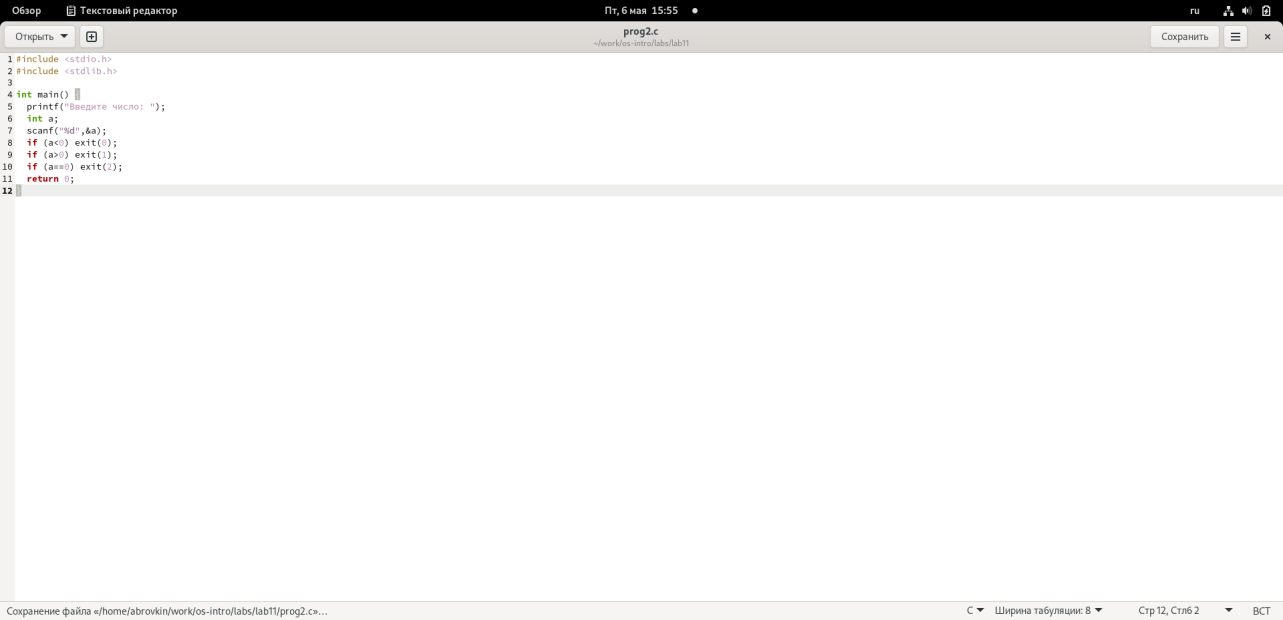


Рис. 4: Пишу новый скрипт-на языке Си

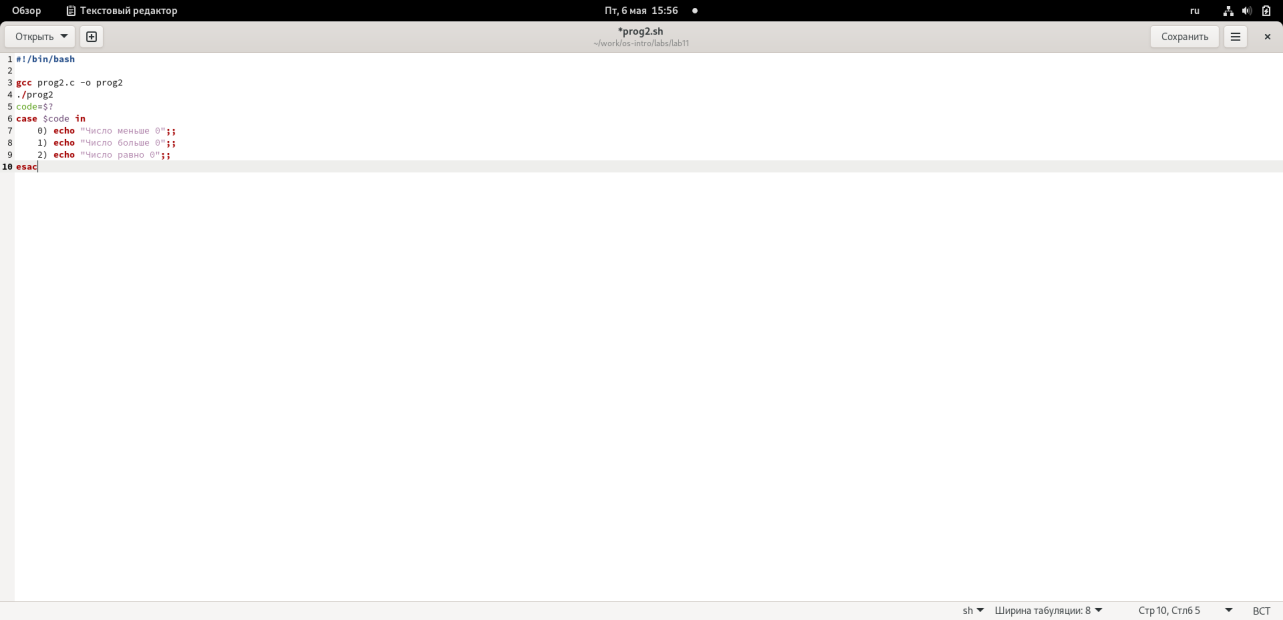


Рис. 5: Пишу еще один скрипт

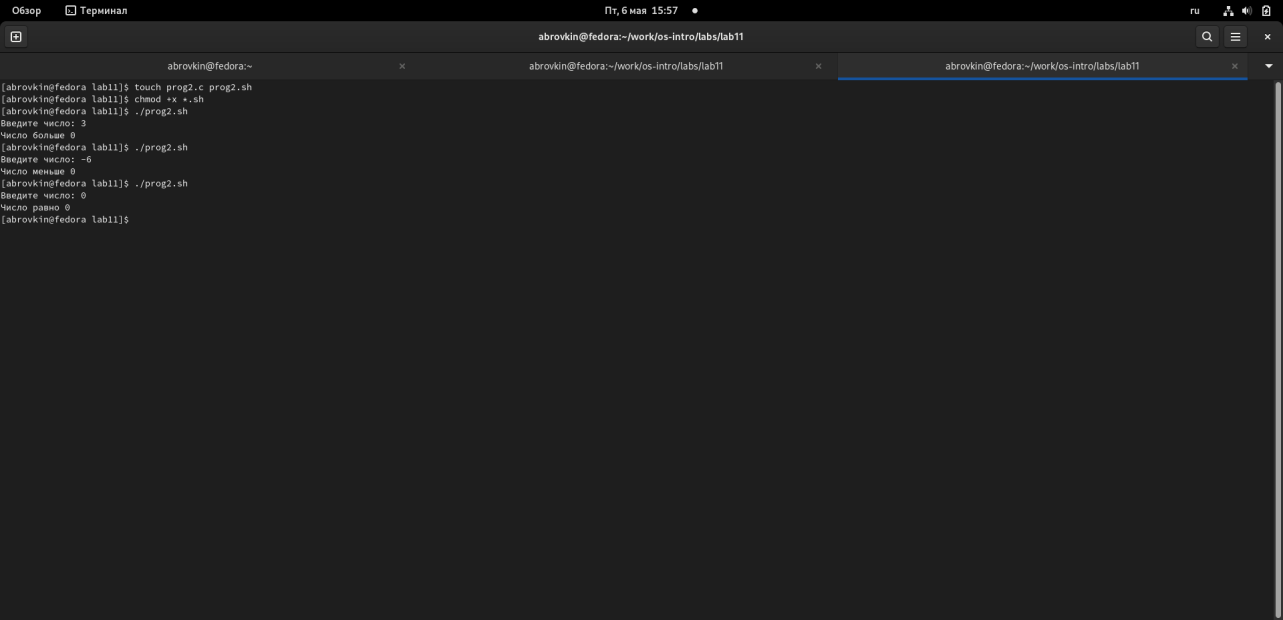


Рис. 6: Проверяю все в терминале

1. Написал 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).(рис. 7)(рис. 8)

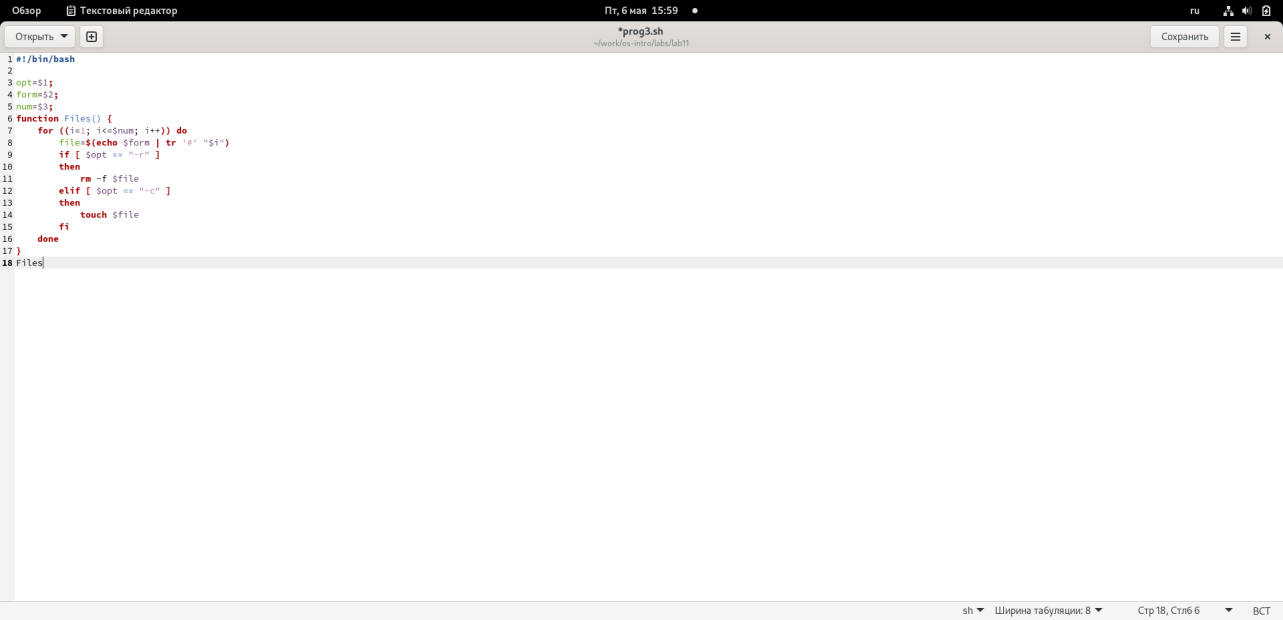


Рис. 7: Пишу новый скрипт

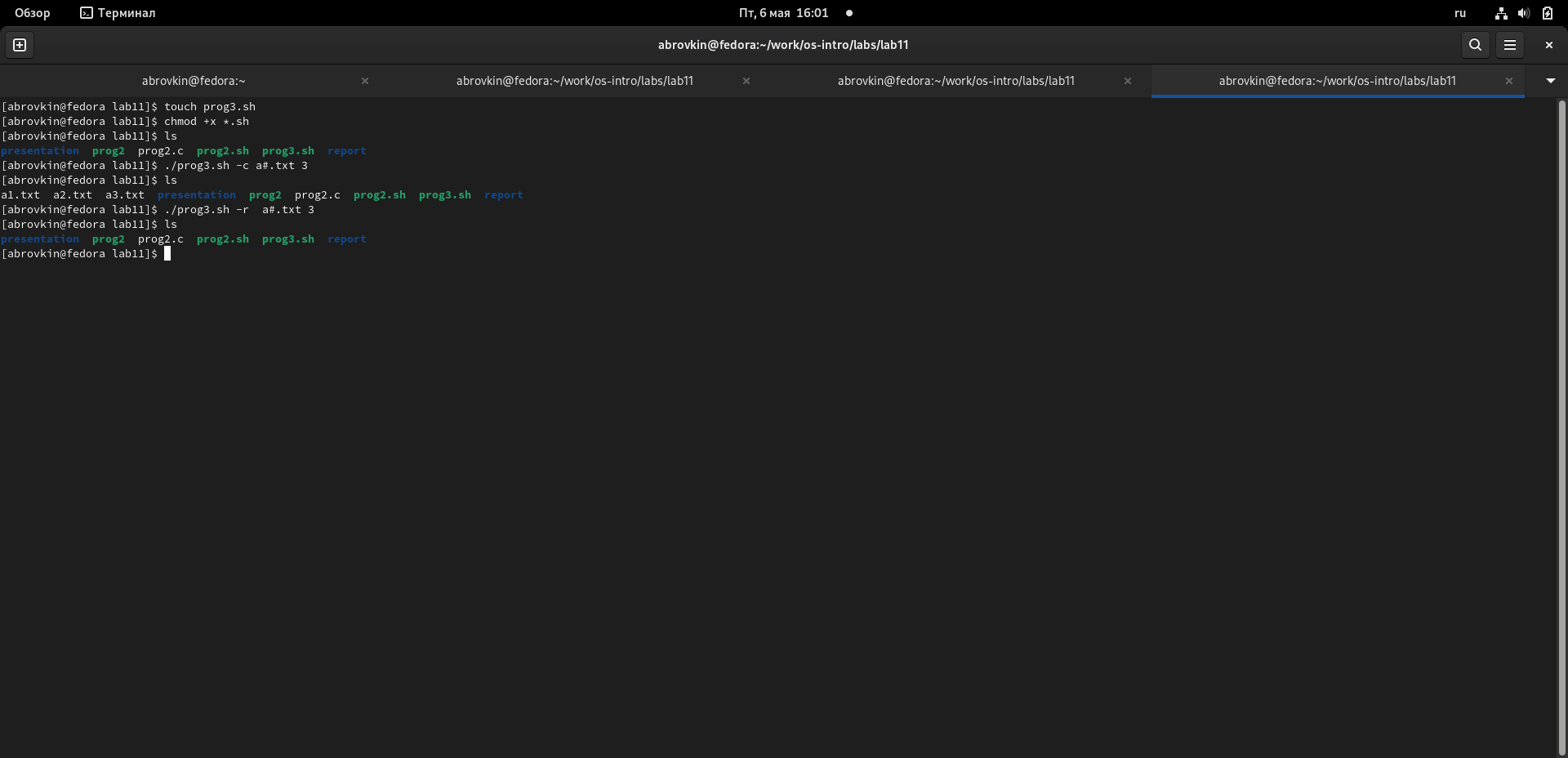


Рис. 8: Проверяю его в терминале

1. Написал командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировал его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовал команду find).(рис. 9)(рис. 10)

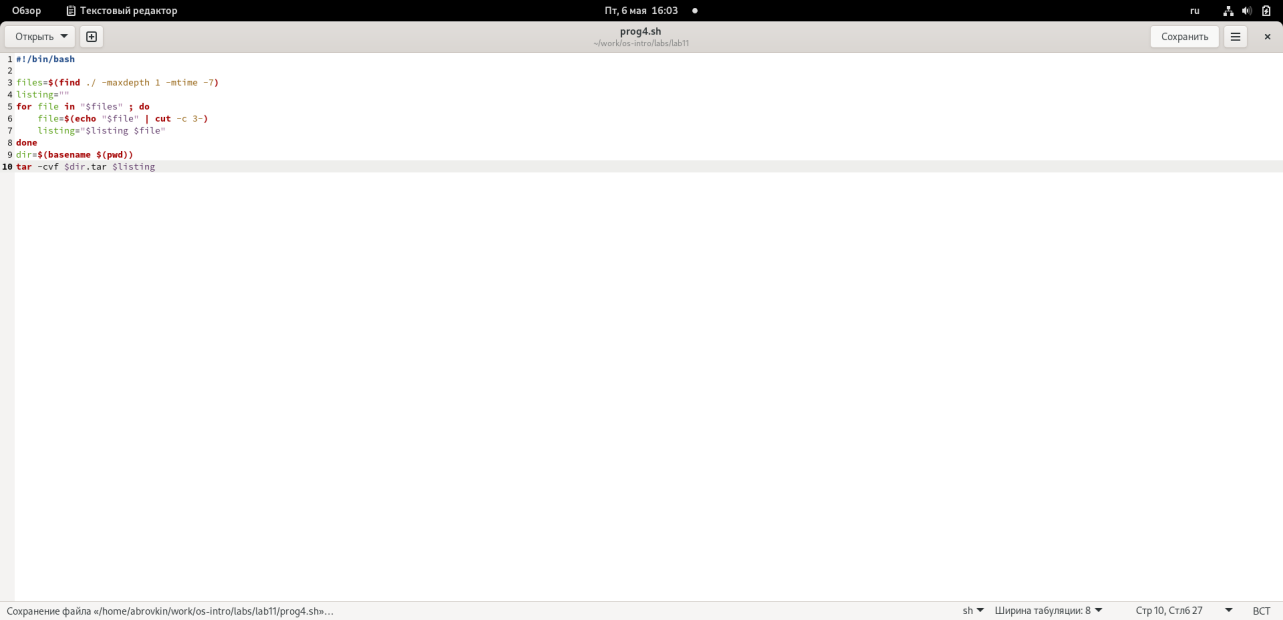


Рис. 9: Пишу новый скрипт

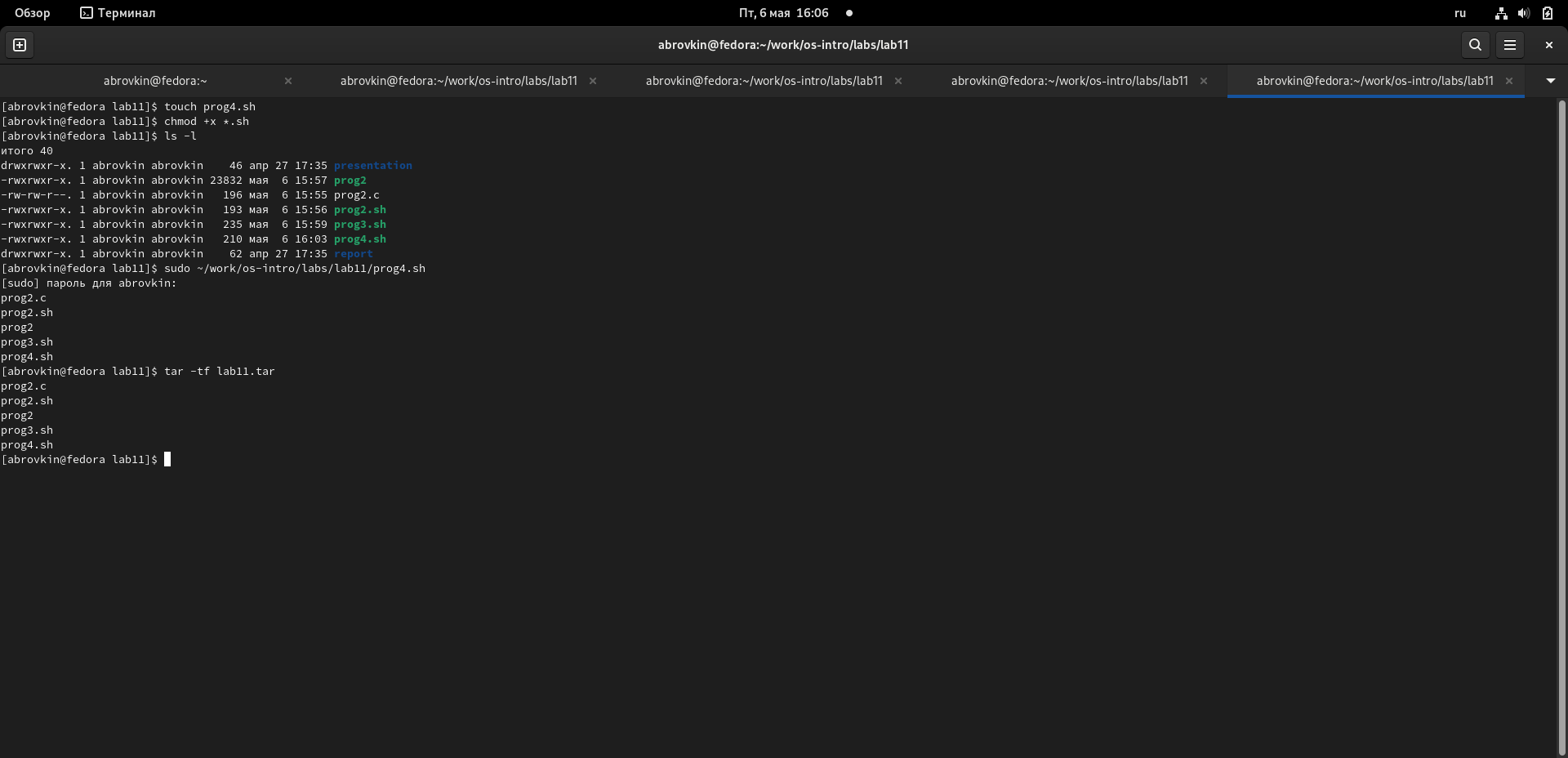


Рис. 10: Проверил его в терминале

# 4 Вывод:

Изучил основы программирования в оболочке ОС UNIX, научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 5 Ответы на контрольные вопросы:

1. Команда getopts является встроенной командой командной оболочки bash, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.
2. При генерации имен используют метасимволы:

* произвольная (возможно пустая) последовательность символов; ? один произвольный символ; […] любой из символов, указанных в скобках перечислением и/или с указанием диапазона; cat f\* выдаст все файлы каталога, начинающиеся с “f”; cat *f* выдаст все файлы, содержащие “f”; cat program.? выдаст файлы данного каталога с однобуквенными расширениями, скажем “program.c” и “program.o”, но не выдаст “program.com”; cat [a-d]\* выдаст файлы, которые начинаются с “a”, “b”, “c”, “d”. Аналогичный эффект дадут и команды “cat [abcd]*” и ”cat [bdac]*”.

1. Операторы && и || являются управляющими операторами. Если в командной строке стоит command1 && command2, то command2 выполняется в том, и только в том случае, если статус выхода из команды command1 равен нулю, что говорит об успешном ее завершении. Аналогично, если командная строка имеет вид command1 || command2, то команда command2 выполняется тогда, и только тогда, когда статус выхода из команды command1 отличен от нуля.
2. Оператор break завершает выполнение ближайшего включающего цикла или условного оператора, в котором он отображается.
3. Команда true всегда возвращает ноль в качестве выходного статуса для индикации успеха. Команда false всегда возвращает не-ноль в качестве выходного статуса для индикации неудачи. Во всех управляющих конструкциях в качестве логического значения используется код возврата из программы, указанной в качестве условия. Код возврата 0 – истина, любое другое значение – ложь. Программа true – всегда завершается с кодом 0, false – всегда завершается с кодом 1.
4. Введенная строка означает условие существования файла mani.$s
5. Цикл While выполняется до тех пор, пока указанное в нем условие истинно. Когда указанное условие становится ложным - цикл завершается. Цикл Until выполняется до тех пор, пока указанное в нем условие ложно.