Отчёта по лабораторной работе №10:

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Кононов Алексей Сергеевич

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:

* -i - inputfile — прочитать данные из указанного файла;
* -o - outputfile — вывести данные в указанный файл;
* -p - шаблон — 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.

1. 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 о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды $?, выдать сообщение о том, какое число было введено.
2. 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).
3. 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

1. Напишем командный файл getopts grep, который анализирует командную строку с ключами (-i, -o, -p,-c, -n), а затем ищет в указанном файле нужные строки, определяемые ключом -p (рис. 1):

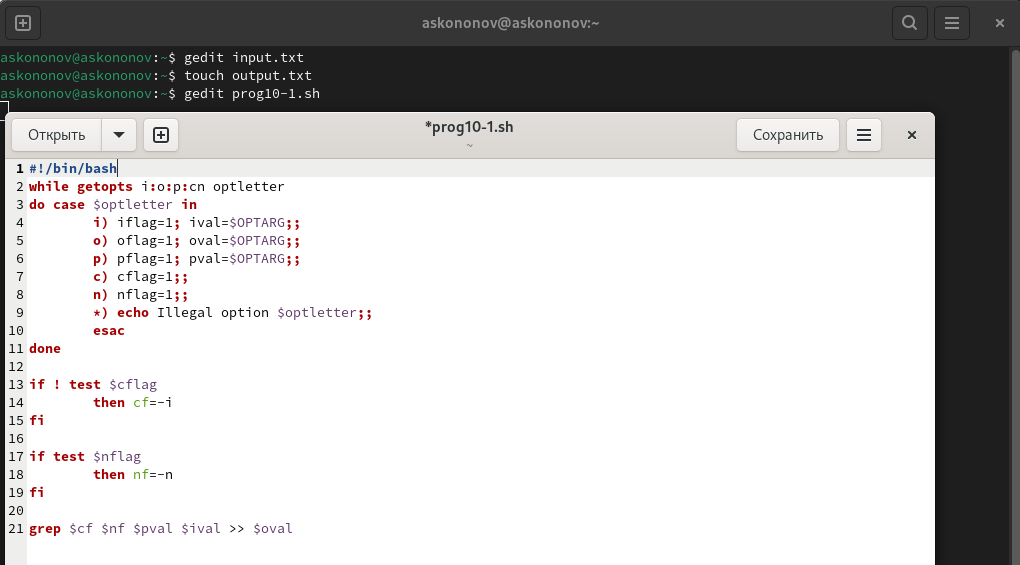


Рис. 1: Задание 1

Делаем текстоовый файл “input.txt” для ввода и “output.txt” для вывода. Делаем файл “prog10-1.sh” исполняемым и выводим результат (рис. 2).

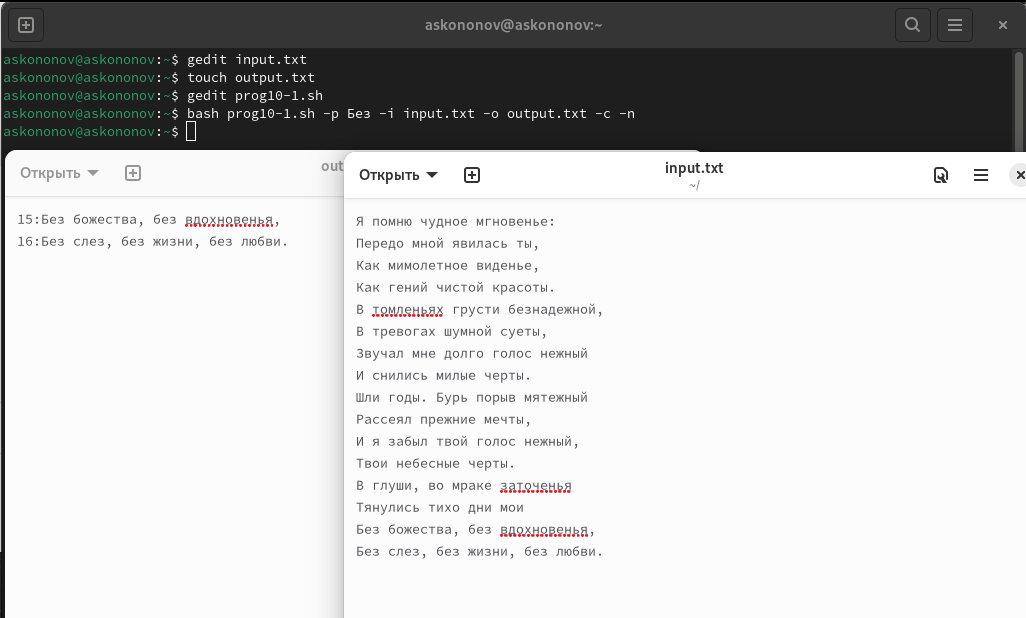


Рис. 2: Выполнение и результат 1

1. Напишем 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 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. 3), (рис. 4):

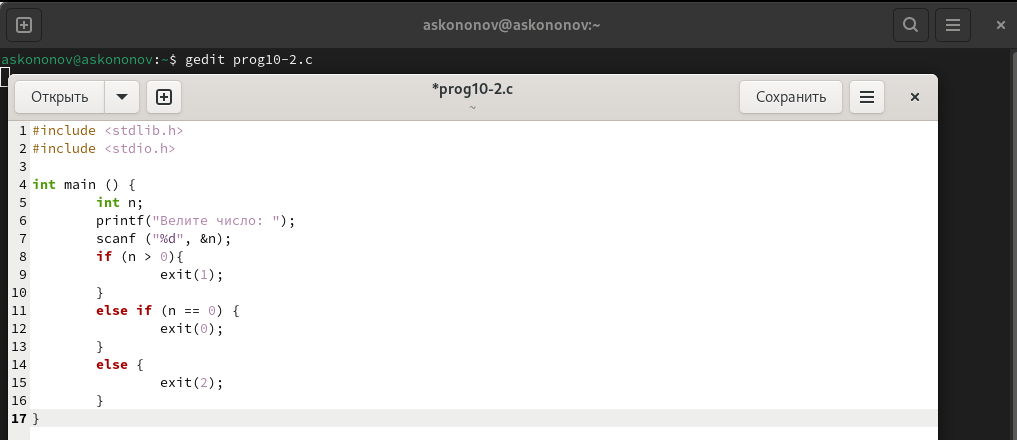


Рис. 3: Задание 2-1

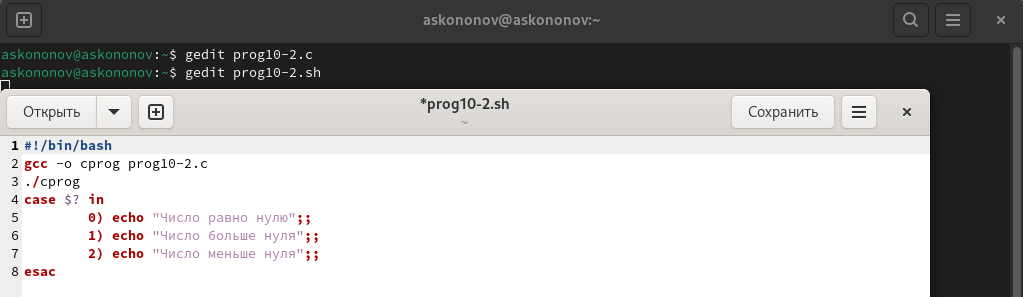


Рис. 4: Задание 2-2

Делаем файлы исполняемыми и выводим результат (рис. 5).

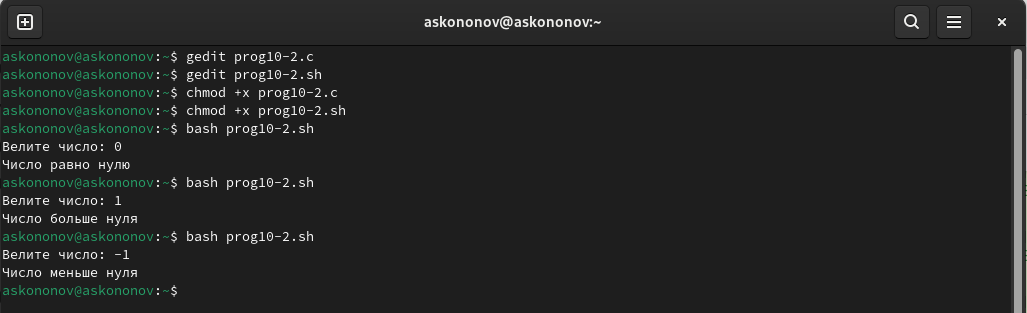


Рис. 5: Выполнение и результат 2

1. Напишем командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N. Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (рис. 6):

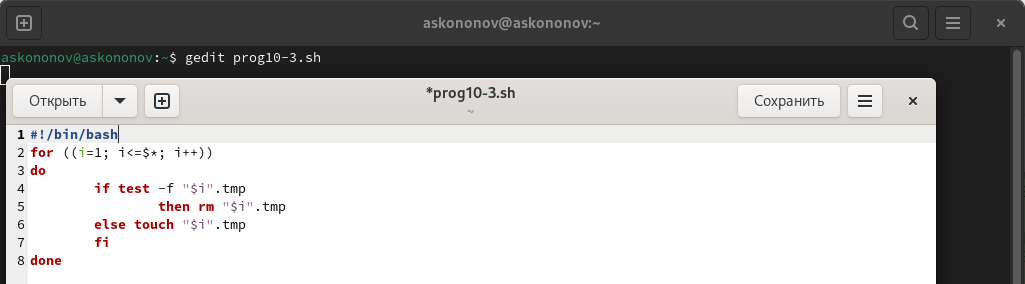


Рис. 6: Задание 3

Делаем файлы исполняемыми и выводим результат (рис. 7).

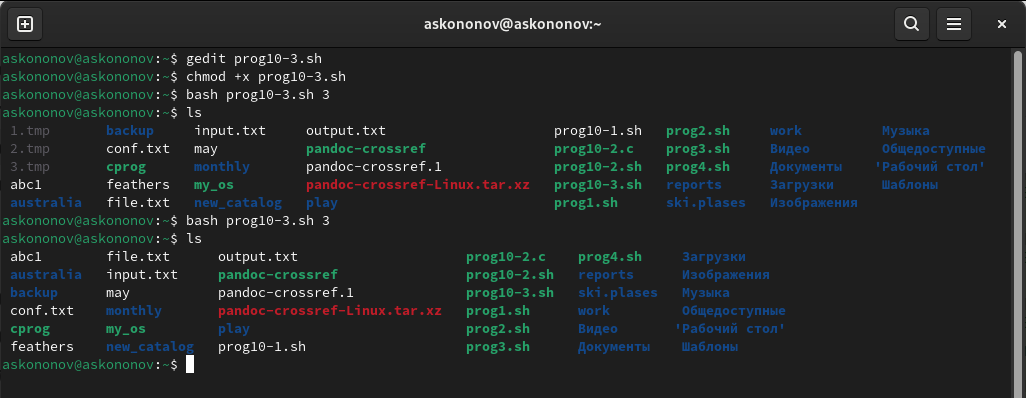


Рис. 7: Выполнение и результат 3

1. Напишем 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, использовав команду find (рис. 8):

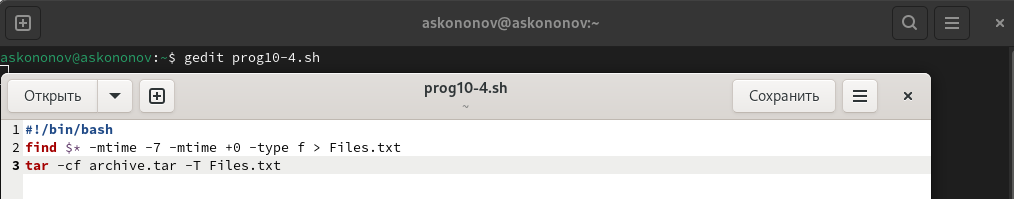


Рис. 8: Задание 4

Делаем файлы исполняемыми и выводим результат (рис. 9).

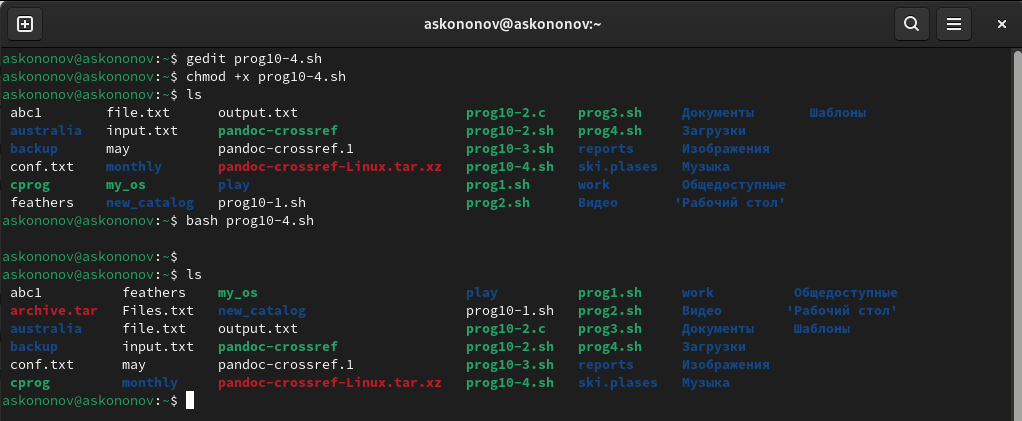


Рис. 9: Выполнение и результат 4

# 4 Контрольные вопросы

1. Каково предназначение команды getopts? С помощью getopts можно достаточно легко произвести разбор флагов переданных скрипту.
2. Какое отношение метасимволы имеют к генерации имён файлов? Метасимволы играют важную роль в генерации имен файлов, особенно в командных оболочках *Unix* и *Linux*. Эти символы представляют собой специальные символы, которые интерпретируются оболочкой для выполнения шаблонного поиска и подстановки. Например:
   * \* - соответствует любому количеству символов в имени файла.
   * ? - соответствует ровно одному символу.
   * [...] - соответствуют любому одному символу из перечисленных в скобках.
3. Какие операторы управления действиями вы знаете?
   * Условные операторы:
     + if, elif и else - позволяют выполнять команды, основываясь на условиях.
     + case - выбор действий в зависимости от значения переменной.
   * Циклы:
     + for - выполняет команды для списка значений.
     + while - цикл который выполняется пока условие истинно.
     + until - цикл который выполняется пока условие ложно.
4. Какие операторы используются для прерывания цикла? В Bash для прерывания циклов используются следующие операторы:
   * break - прерывает выполнение текущего цикла и передает управление на команду, следующую за циклом. Вы можете указать break n, где n - это количество уровней цикла, которые нужно прервать.
   * continue - пропускает оставшуюся часть тела текущего цикла и переходит к следующей итерации. Аналогично break, можно использовать continue n для пропуска итераций во вложенных циклах.
5. Для чего нужны команды false и true? Команды true и false в Bash являются простыми утилитами, которые возвращают статус выхода. Они используются в скриптах и условных операторах для управления логикой выполнения.
   * true всегда возвращает статус выхода 0, что означает успех. Эта команда может быть использована в местах, где требуется гарантированно успешный результат, например, в бесконечных циклах (while true; do ... done) или как заглушка для функции, которая еще не реализована.
   * false всегда возвращает статус выхода 1, что означает неудачу. Эта команда может быть использована для преднамеренного вызова ошибки или как условие, которое никогда не будет выполнено.
6. Что означает строка if test -f man$s/$i.$s, встреченная в командном файле? В этой строке проверяется существование файла man$s/$i.$s, где $s и $i переменные подставляющиеся в имя файла.
7. Объясните различия между конструкциями while и until.
   * while — цикл который выполняется пока условие истинно.
   * until — цикл который выполняется пока условие ложно.

# 5 Выводы

В этой работе мы поближе познакомились с циклами в языке bash. Научились писать более сложные командные файлы используя логические управляющие конструкции.

# Список литературы