Отчёт по лабораторной работе №11

Операционные системы

Гусева Светлана Алексеевна

Содержание

# Цель работы

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

# Задание

1. Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.
3. Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.
4. Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки.

# Выполнение лабораторной работы

1. Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.

Создание папки backup (рис. [1](#fig:001)).

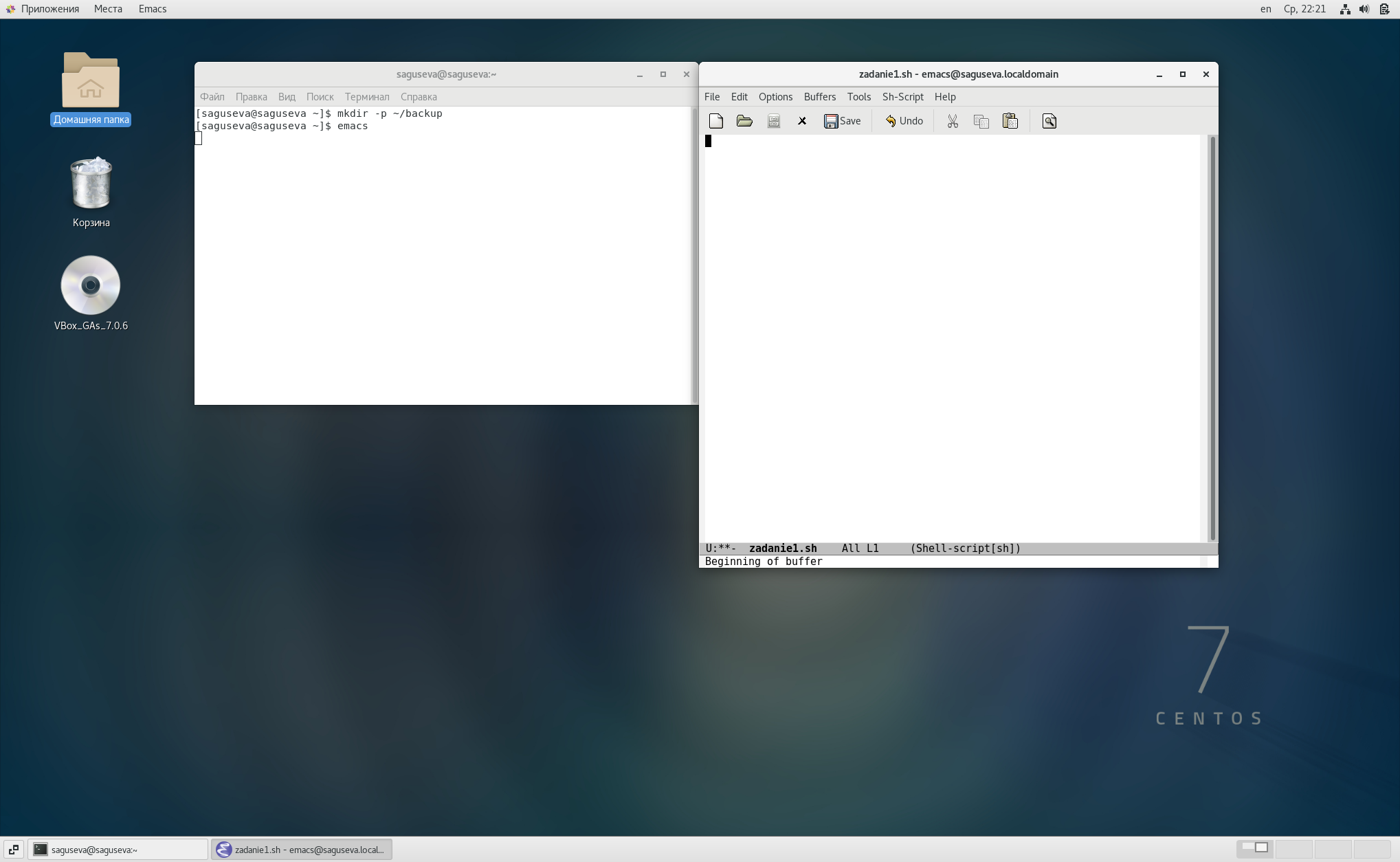


Figure 1: Создание папки backup.

Скрипт и результат выполнения задания 1 (рис. [2](#fig:002)).

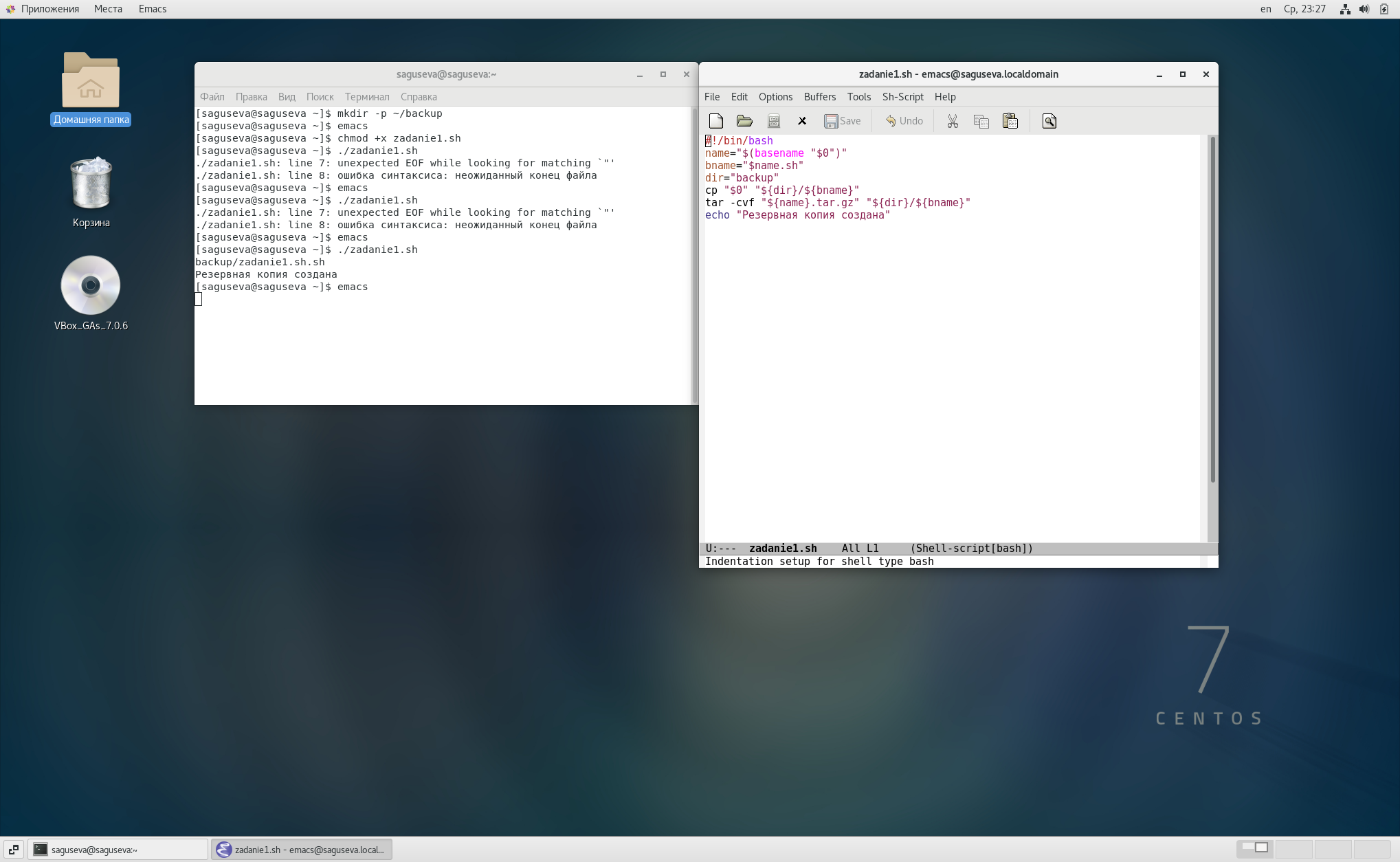


Figure 2: Скрипт и выполнение задания 1.

1. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов. Скрипт и результат выполнения задания 2 (рис. [3](#fig:003)).

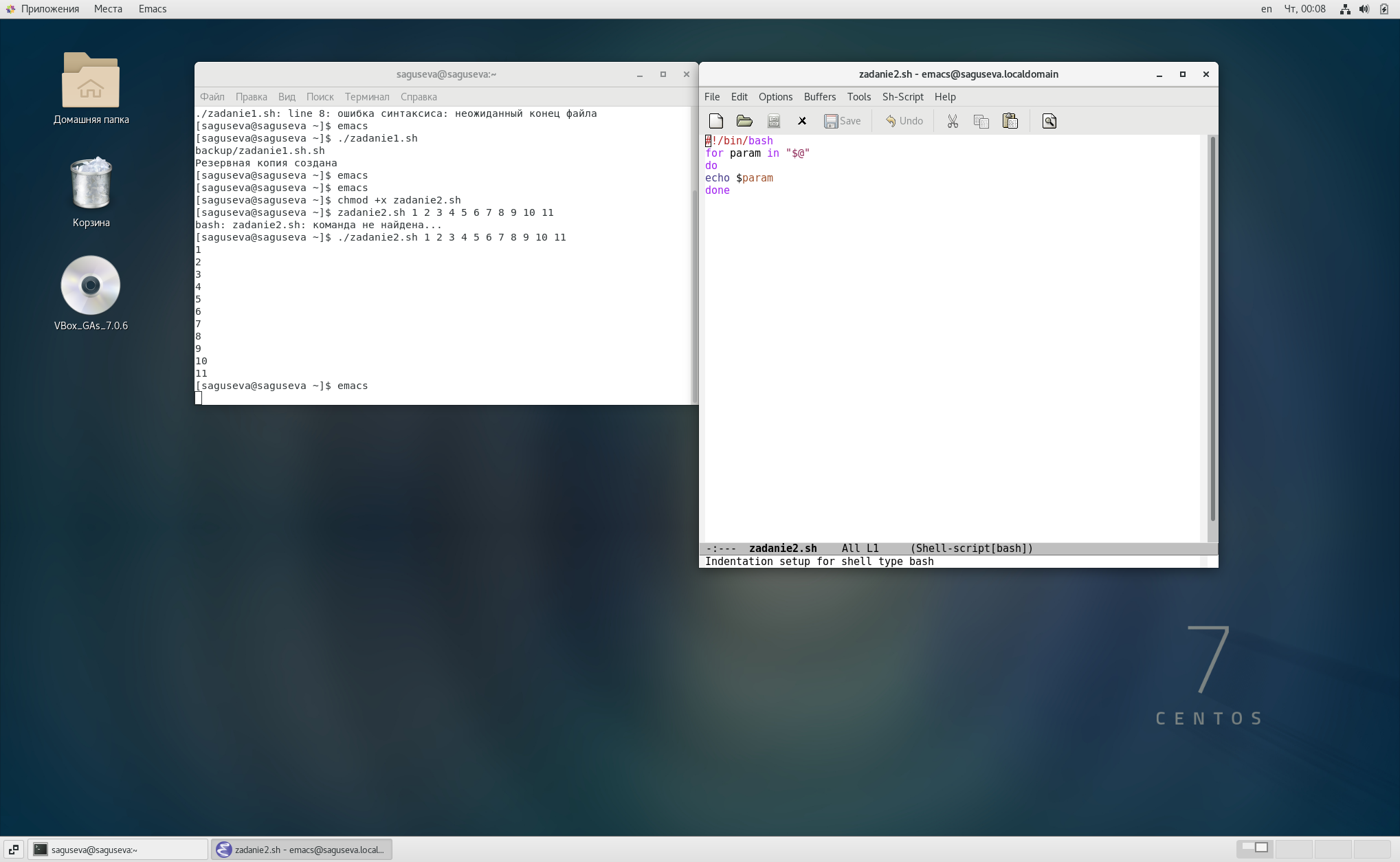


Figure 3: Скрипт и выполнение задания 2.

1. Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (рис. [4](#fig:004)). Скрипт выполнения задания 3 (рис. [4](#fig:004)), результат выполнения (рис. [5](#fig:005), [6](#fig:006), [7](#fig:007)).

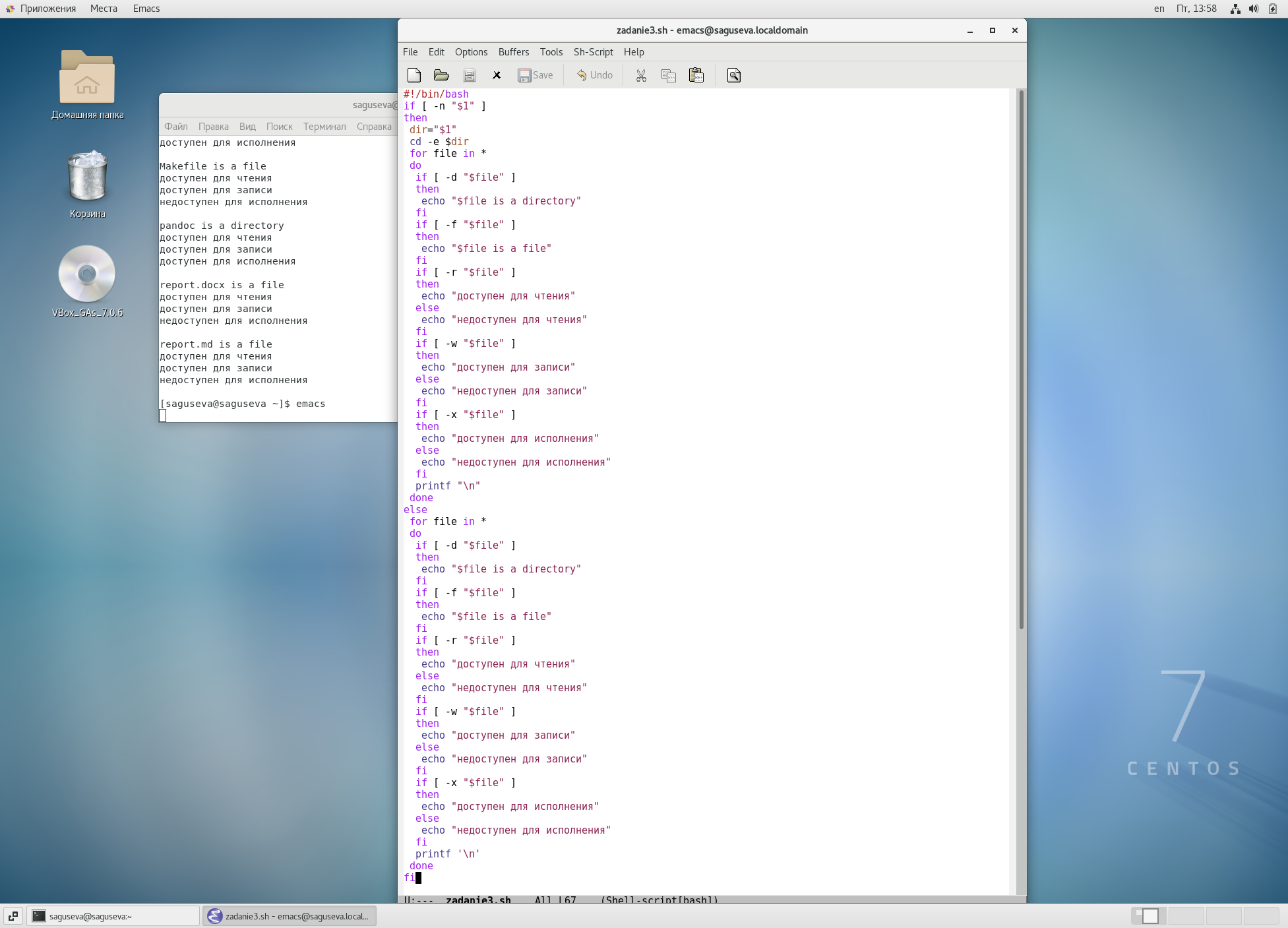


Figure 4: Скрипт задания 3.

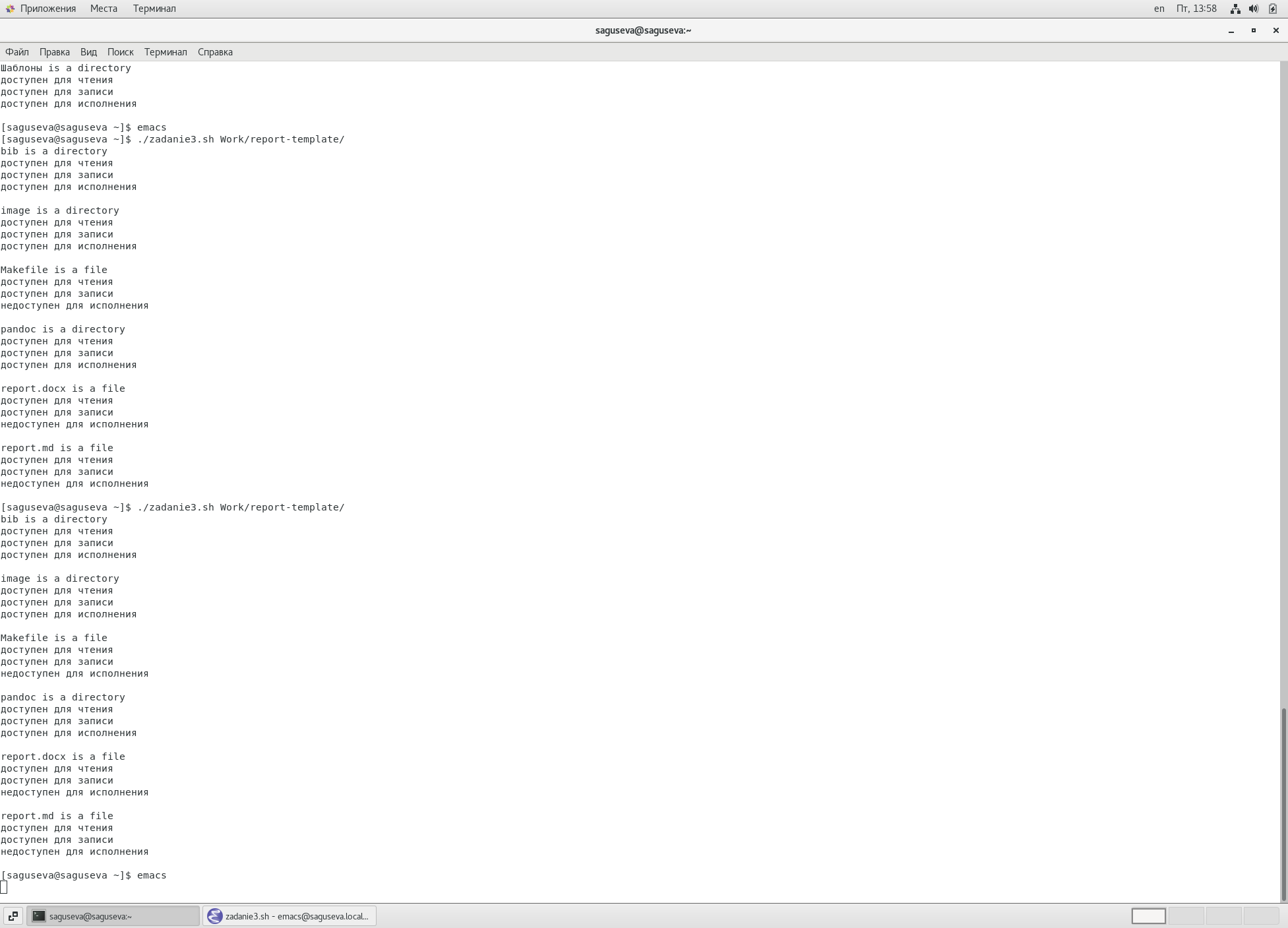


Figure 5: Результат выполнения задания 3.

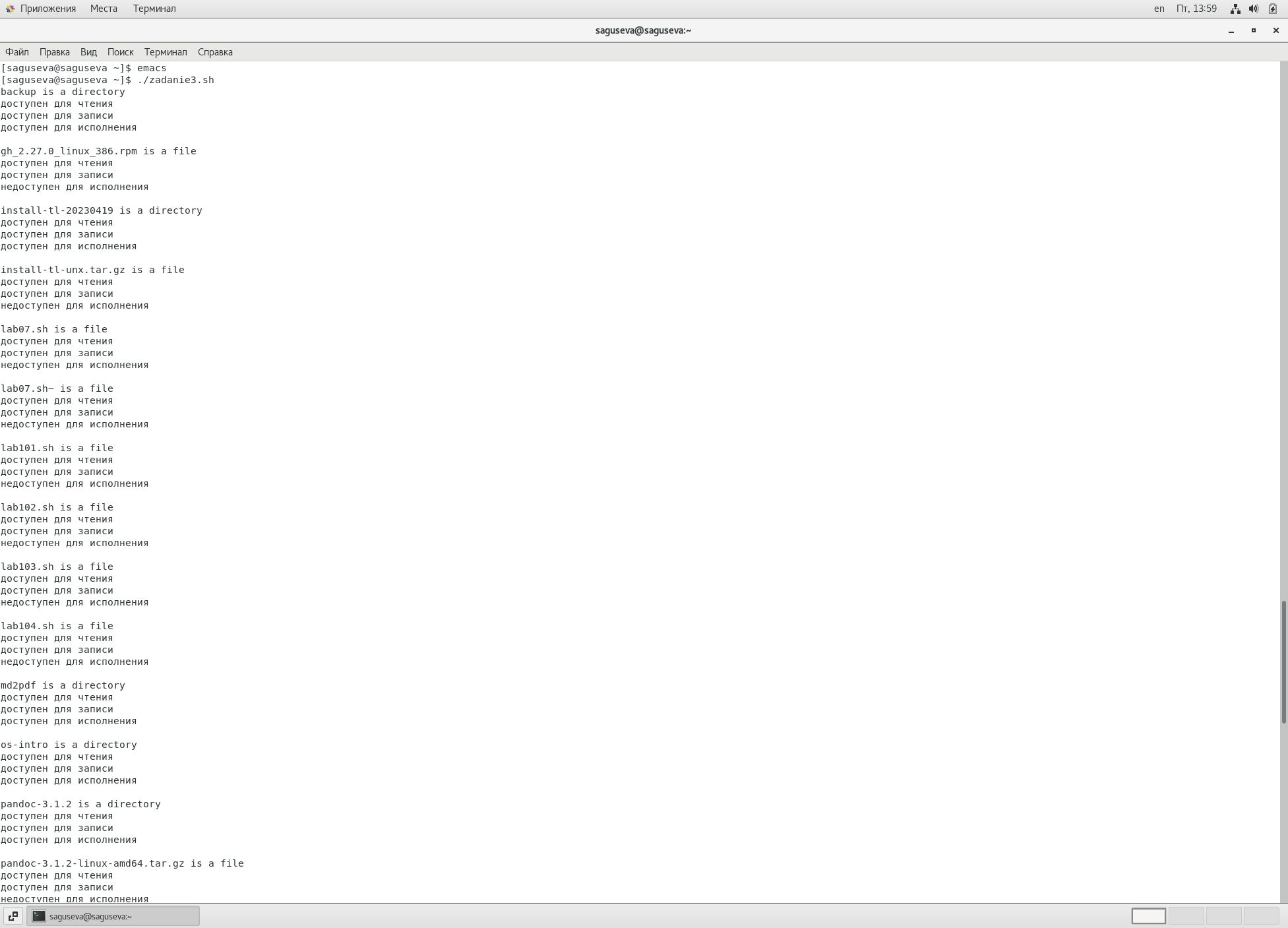


Figure 6: Результат выполнения задания 3.

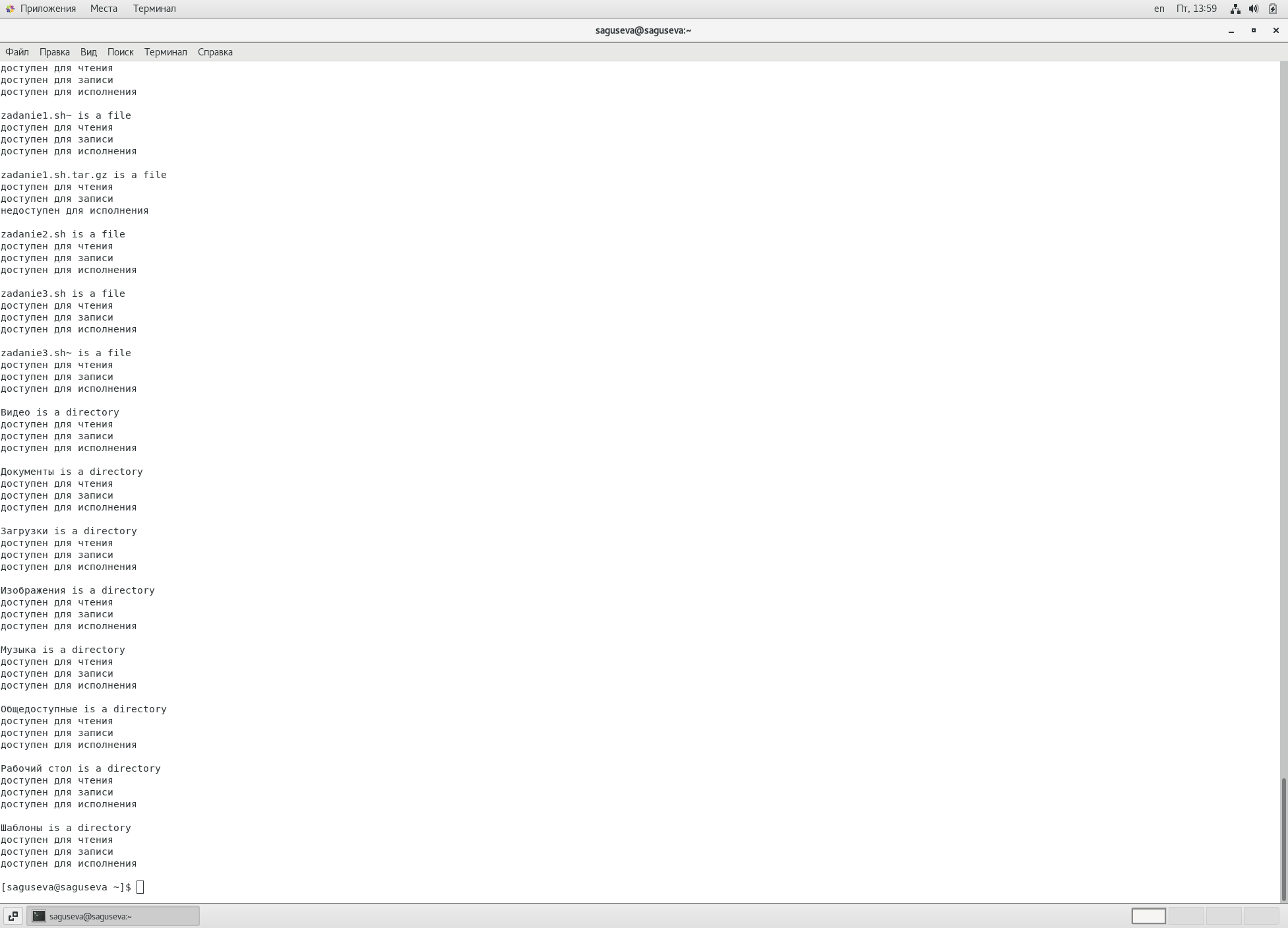


Figure 7: Результат выполнения задания 3.

1. Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки. Скрипт и результат выполнения задания 4 (рис. [8](#fig:008)).

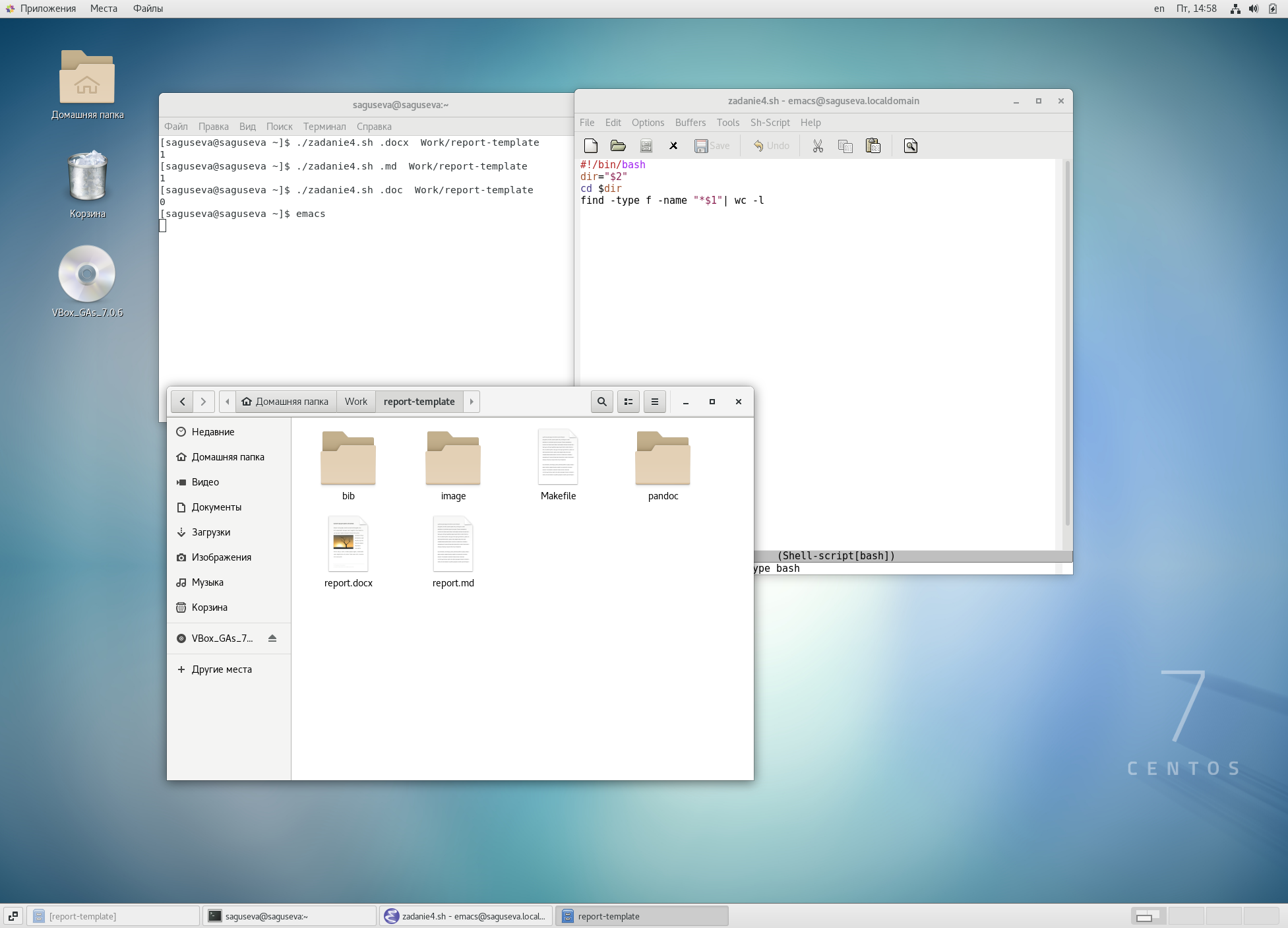


Figure 8: Скрипт и выполнение задания 4.

# Контрольные вопросы

1. Объясните понятие командной оболочки. Приведите примеры командных оболочек. Чем они отличаются? Командная оболочка — программа, через которую пользователь или администратор управляет операционной системой и установленными программами, используя командную строку. Примеры и отличия 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 Сподобный 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).
2. Что такое POSIX? POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ.
3. Как определяются переменные и массивы в языке программирования bash? Значениями переменной могут быть число, строка или отдельные символы. Пример определения переменной: path="$HOME" Чтение переменной: "${path}" Пример определения массива: a=(1 2 3 4 5) В массивах можно хранить числа и строки одновременно, например: b=(1 2 "three" 4 "five") Обращение к элементу массива с номером i: ${a[$i]} Чтобы вывести все элементы массива, требуется использовать символ @: ${a[@]}
4. Каково назначение операторов let и read? Let — это встроенная функция bash, которая позволяет производить базовые арифметические операции. Read — это встроенная команда bash, которая считывает строку из стандартного ввода (или из файлового дескриптора) и разбивает строку на слова.
5. Какие арифметические операции можно применять в языке программирования bash? Сложение, вычитание, умножение, деление, возведение в степень, остаток от деления.
6. Что означает операция (( ))? Внутри двойных скобок записываются арифметические действия или условия.
7. Какие стандартные имена переменных Вам известны? HOME — имя домашнего каталога пользователя. Если команда cd вводится без аргументов, то происходит переход в каталог, указанный в этой переменной. IFS — последовательность символов, являющихся разделителями в командной строке, например, пробел, табуляция и перевод строки (new line). MAIL — командный процессор каждый раз перед выводом на экран промптера проверяет содержимое файла, имя которого указано в этой переменной, и если содержимое этого файла изменилось с момента последнего ввода из него, то перед тем как вывести на терминал промптер, командный процессор выводит на терминал сообщение You have mail (у Вас есть почта). TERM — тип используемого терминала. LOGNAME — содержит регистрационное имя пользователя, которое устанавливается автоматически при входе в систему.
8. Что такое метасимволы? Символы, которые имеют особое назначение. Например, \* соответствует произвольной, в том числе и пустой строке.
9. Как экранировать метасимволы? Экранирование может быть осуществлено с помощью предшествующего метасимволу символа , который, в свою очередь, является метасимволом.
10. Как создавать и запускать командные файлы? Последовательность команд может быть помещена в текстовый файл. Такой файл называется командным. Далее этот файл можно выполнить по команде: bash командный\_файл [аргументы] Чтобы не вводить каждый раз последовательности символов bash, необходимо изменить код защиты этого командного файла, обеспечив доступ к этому файлу по выполнению. Это может быть сделано с помощью команды chmod +x имя\_файла Теперь можно вызывать свой командный файл на выполнение, просто вводя его имя с терминала.
11. Как определяются функции в языке программирования bash? Функция определяется с помощью ключевого слова function, после которого следует имя функции и список команд, заключённых в фигурные скобки.
12. Каким образом можно выяснить, является файл каталогом или обычным файлом? Если выражение -d "$file" истино, то файл является каталогом. Если выражение -f "$file" истино, то файл является файлом.
13. Каково назначение команд set, typeset и unset? Set — это встроенная команда оболочки, которая позволяет отображать или устанавливать переменные оболочки и среды. Typeset имеет четыре опции для работы с функциями: – -f — перечисляет определённые на текущий момент функции; – -ft — при последующем вызове функции инициирует её трассировку; – -fx — экспортирует все перечисленные функции в любые дочерние программы оболочек; – -fu — обозначает указанные функции как автоматически загружаемые. Автоматически загружаемые функции хранятся в командных файлах, а при их вызове оболочка просматривает переменную FPATH, отыскивая файл с одноимёнными именами функций, загружает его и вызывает эти функции. Unset удаляет переменную по имени до конца текущей сессии. Удалить функцию можно с помощью команды unset c флагом -f.
14. Как передаются параметры в командные файлы? При вызове командного файла через пробел. В тексте командного файла первый из них мы обозначается записью $1, второй записью $2, третий записью $3 и т.д. до $9.
15. Назовите специальные переменные языка bash и их назначение. $\* — отображается вся командная строка или параметры оболочки; $? — код завершения последней выполненной команды; $$ — уникальный идентификатор процесса, в рамках которого выполняется командный процессор; $! — номер процесса, в рамках которого выполняется последняя вызванная на выполнение в командном режиме команда; $- — значение флагов командного процессора; ${#*} — возвращает целое число — количество слов, которые были результатом $*; ${#name} — возвращает целое значение длины строки в переменной name; ${name[n]} — обращение к n-му элементу массива; ${name[\*]} — перечисляет все элементы массива, разделённые пробелом; ${name[@]} — то же самое, но позволяет учитывать символы пробелы в самих переменных; ${name:-value} — если значение переменной name не определено, то оно будет заменено на указанное value; ${name:value} — проверяется факт существования переменной; ${name=value} — если name не определено, то ему присваивается значение value; ${name?value} — останавливает выполнение, если имя переменной не определено, и выводит value как сообщение об ошибке; ${name+value} — это выражение работает противоположно ${name-value}. Если переменная определена, то подставляется value; ${name#pattern} — представляет значение переменной name с удалённым самым коротким левым образцом (pattern); ${#name[\*]} и ${#name[@]} — эти выражения возвращают количество элементов в массиве name.

# Выводы

В процессе выполнения мной были изучены основы программирования в оболочке ОС UNIX/Linux и получены навыки написания небольших командных файлов.

# Список литературы

https://bestprogrammer.ru/izuchenie/komanda-set-linux?ysclid=lhklmm0xlv908627071 }