МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем

и технологий

**Отчет**

**по лабораторной работе № 3**

по дисциплине: ”Системное программирование”

на тему: ***”*****Командный язык и скрипты Shell*”***

Выполнил**:** студент группы *10702121* Агеенко А. К.

Высоцкий М. Л.

Градусов А. Р.

Принял**:** пр. Давыденко Н. В.

Минск 2023

# Лабораторная работа №3.

**Цель работы:** Закрепить на практике основы работы со сценариями командной оболочки и перенаправление ввода/вывода.

## Задание 1

1. Написать скрипт, который записывает содержимое одного файла в другой и при этом оба файла существуют до операции. Дать два разных варианта выполнения.

2. Найти все файлы, начинающиеся на слово script и записать их имена в файл list.txt.

3. Написать скрипт, который проверяет, содержится ли скриптовый файл в директории, если да, то выполнить его.

4. Написать скрипт, подсчитывающий сумму от 1 до 10.

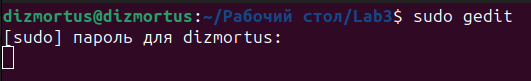
5. А какие командные интерпретаторы установлены в вашей системе?

Напишите скрипт поиска местонахождения командных интерпретаторов вашей системы. Также определите какой интерпретатор используется сейчас в вашей системе?

Решение

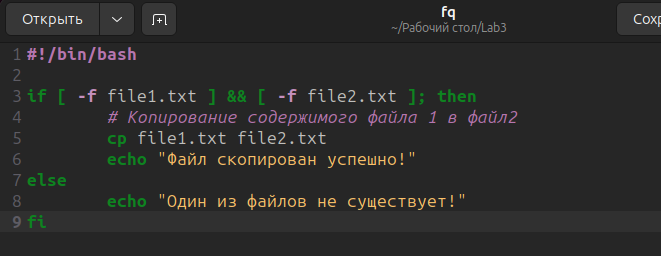
**1.** Написать скрипт, который записывает содержимое одного файла в другой и при этом оба файла существуют до операции. Дать два разных варианта выполнения.

1. Вызываем текстовый редактор

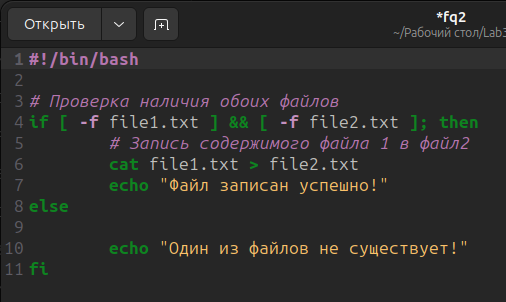


1. Пишем код сценария и сохраняем файл

1) Используя команду `cp` для копирования файла:



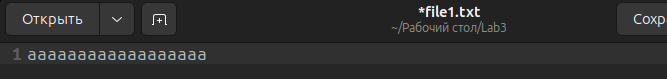
2) Используя оператор `cat` для чтения и записи файлов:



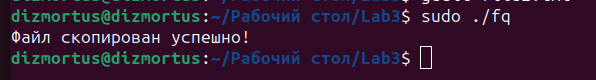
1. Делаем скрипт выполняемым файлом



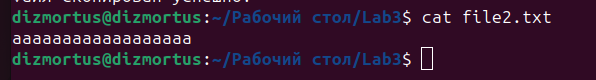
4. Создаем два текстовых файла и заносим текст в один из них.



5. Запускаем скрипт



6. Проверяем наш второй файл:

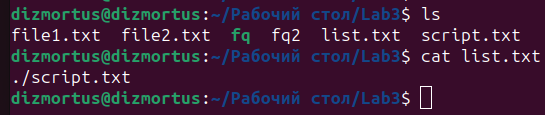


**2.** Найти все файлы, начинающиеся на слово script и записать их имена в файл list.txt.

1. Выполняем команду find

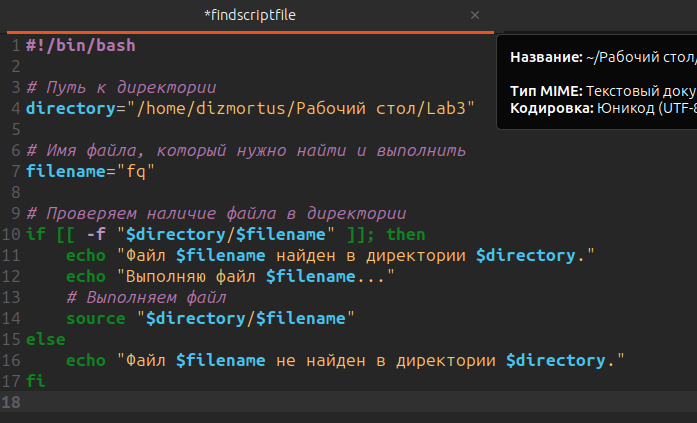


2. Проверяем результат

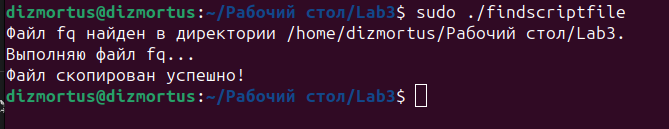


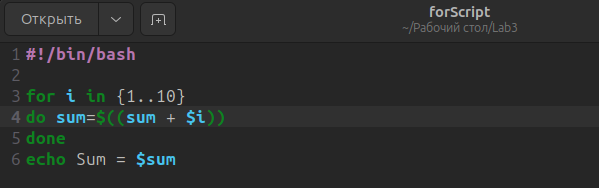
**3.** Написать скрипт, который проверяет, содержится ли скриптовый файл в директории, если да, то выполнить его.

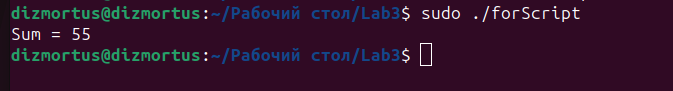
1. Пишем скрипт.



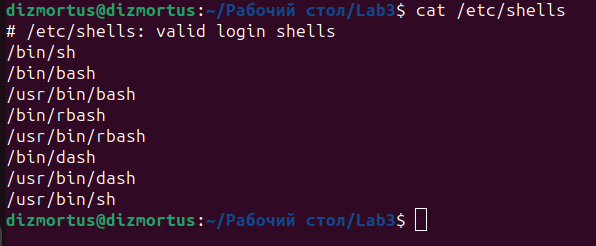
2. Запускаем и проверяем работу



**4.** Написать скрипт, подсчитывающий сумму от 1 до 10. 

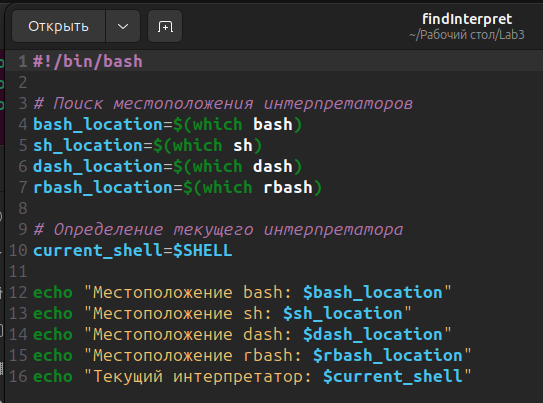


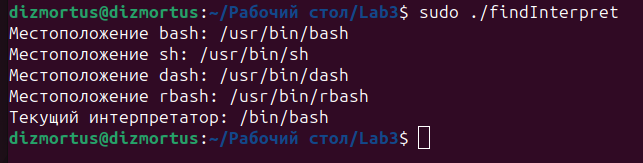
**5** А какие командные интерпретаторы установлены в вашей системе?



sh, bash, rbash, dash

Напишите скрипт поиска местонахождения командных интерпретаторов вашей системы. Также определите какой интерпретатор используется сейчас в вашей системе?





**Задание 2.**

6. Используйте текст первого примера, заменить некоторые фамилии на фамилии членов вашей команды (например, Nutrina заменить на Stankewich).

7. Отсортируйте данные файла так, чтобы он не поменял свое собственное имя. (Решение. Нужно предварительно создать промежуточный отсортированный файл t11.txt. Затем выполнить нужные команды).

8. Подсчитайте число одинаковых слов в обоих файлах. Для подсчета числа слов в файле используйте команду: wc -w file1.

9. Написать скрипт, который создает отсортированный файл, содержащий слова из двух файлов, исключая их общую часть одинаковых слов.

10.Вывести фамилию самого молодого человека (в файле basa.txt – это второй столбец).

11.Вывести зарплату самого молодого человека (зарплата – третий столбец).

12.Вывести отсортированный список имен файлов, начальная часть имени есть script. 13.Написать скрипт, который в каждой строке файла оставляет только буквенные символы, а остальные символы выбрасывает.

Пример. Пусть содержимое файла есть:

Privet year 2022

Good bye year 2021.

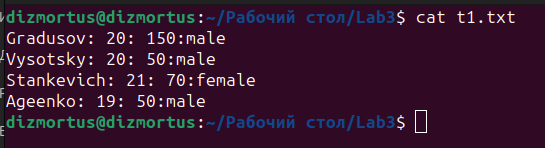
Должны получить такой файл:

Privet year

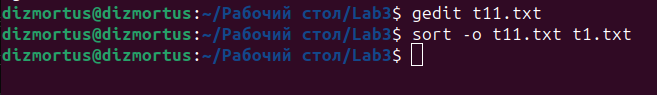
Good bye year

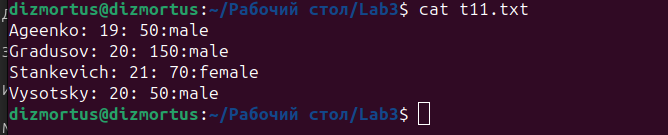
14.Вывести упорядоченный список имен файлов, в именах которых содержится символ подчеркивания, например, script\_sort.

**6.** Используйте текст первого примера, заменить некоторые фамилии на фамилии членов вашей команды (например, Nutrina заменить на Stankewich).

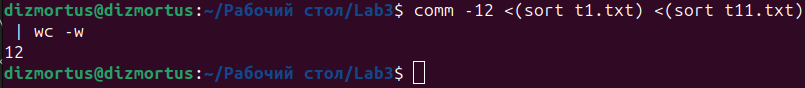


**7.** Отсортируйте данные файла так, чтобы он не поменял свое собственное имя. (Решение. Нужно предварительно создать промежуточный отсортированный файл t11.txt. Затем выполнить нужные команды).



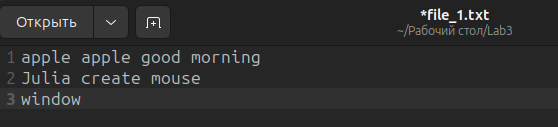


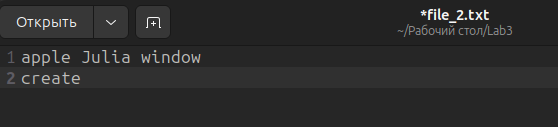
**8.** Подсчитайте число одинаковых слов в обоих файлах. Для подсчета числа слов в файле используйте команду: wc -w file1.



**9.** Написать скрипт, который создает отсортированный файл, содержащий слова из двух файлов, исключая их общую часть одинаковых слов.

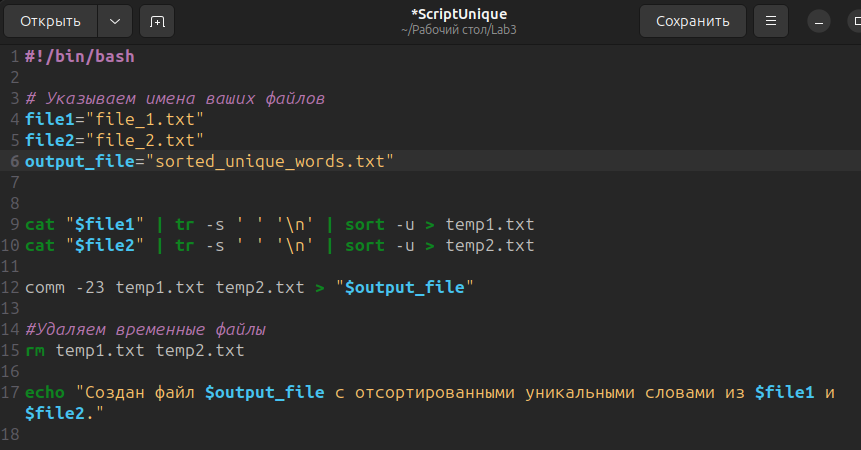
1. Создаем два файла и заполняем их.



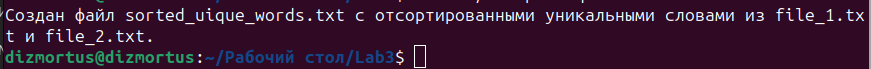


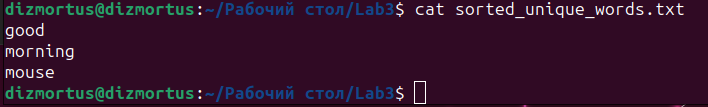
2. Пишем скрипт.

Здесь мы при помощи tr -s разделяем заменяем пробелы, но символ '\n', то есть переводим слова из одной строки на новые. Затем при помощи sort -u сортируем файл и удаляем дубликаты. И записываем результаты в два временных файла. Потом при помощи comm -23 сравниваем содержимое временных файлов и выводим только строки, которые есть только в первом файле (**-2**) и только во втором файле (**-3**).

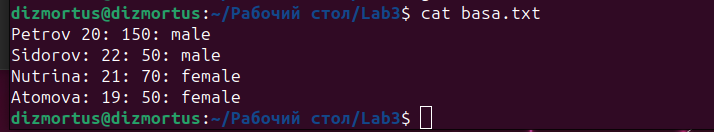


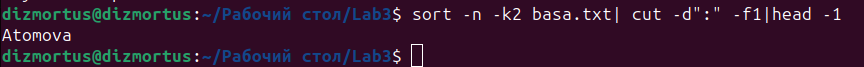
3. Проверяем результат.



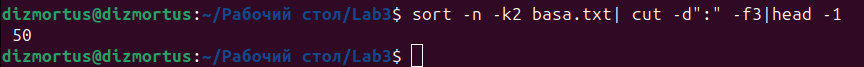


**10.** Вывести фамилию самого молодого человека (в файле basa.txt – это второй столбец).

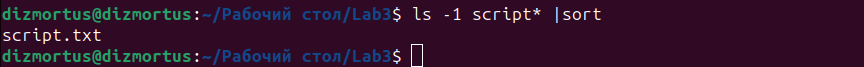




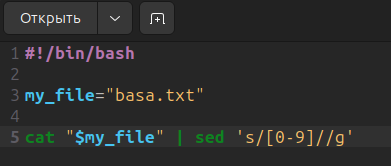
**11.** Вывести зарплату самого молодого человека (зарплата – третий столбец).

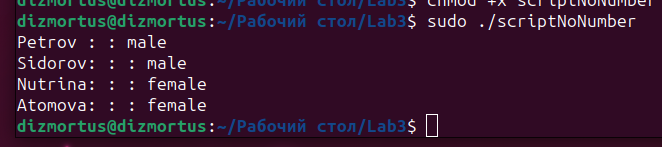


**12.** Вывести отсортированный список имен файлов, начальная часть имени есть script.

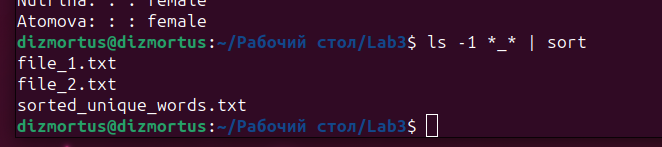


**13.** Написать скрипт, который в каждой строке файла оставляет только буквенные символы, а остальные символы выбрасывает.

****



**14.** Вывести упорядоченный список имен файлов, в именах которых содержится символ подчеркивания, например, script\_sort.

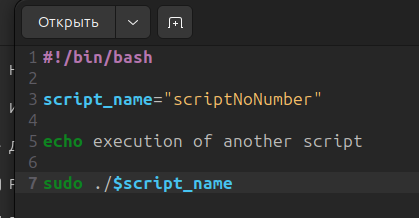


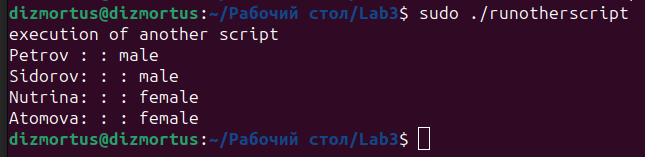
**Задание 3.**

15.Напишите скрипт запуска другого скрипта.

16.Расположите скрипт в одной из стантадртных директорий системы, и запустите его из домашней директории пользователя.

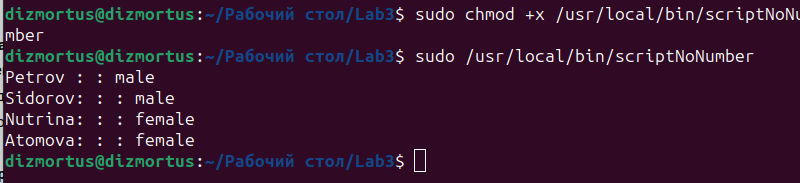
**15.** Напишите скрипт запуска другого скрипта.

****



**16.** Расположите скрипт в одной из стантартных директорий системы, и запустите его из домашней директории пользователя.





1) Что такое скрипт или сценарий командной оболочки, для чего они используются?

Скрипт или сценарий командной оболочки - это файл, содержащий последовательность команд, которые выполняются в командной оболочке. Они используются для автоматизации задач и выполнения серии команд в определенной последовательности. Скрипты облегчают работу с командной оболочкой, позволяя создавать и запускать сложные команды и комбинации команд.

2) Какими провами доступа должны бить наделены файлы скриптов?

Файлы скриптов должны иметь права доступа на выполнение (execute) для пользователя, который планирует выполнять скрипт. Обычно назначаются права 755, что означает, что владелец имеет права на чтение, запись и выполнение, а остальные пользователи могут только читать и выполнять скрипт.

3) Какие способы запуска выполнения скриптов вы знаете? Как запуститьскрипт на который отсутствуют права исполнения (запуска)?

Существует несколько способов запуска скриптов:

- Прямой вызов команды `./script.sh` - если файл скрипта имеет права исполнения.

- Запуск скрипта через интерпретатор командной оболочки: `bash script.sh` или `sh script.sh`.

- Из другого выполняемого скрипта с помощью команды `source` или `. script.sh`.

- Добавление скрипта в cron для планирования его выполнения в определенное время.

Если на скрипт отсутствуют права исполнения, его можно запустить, передавая его как аргумент интерпретатору оболочки: `bash script.sh`.

4) Какое значение имеет месторасположение скриптов в файловой системе для их запуска?

Месторасположение скриптов в файловой системе имеет значение для их запуска. Если скрипт находится в текущем рабочем каталоге, его можно запустить, указав полный путь до него (например, `/path/to/script.sh`) или использовав относительный путь (`./script.sh`, если он находится в текущем каталоге).

Кроме того, система может иметь определенные каталоги, указанные в переменной окружения PATH, в которых она ищет исполняемые файлы. Если скрипт находится в одном из этих каталогов (например, `/usr/local/bin`), его можно запустить, просто указав имя скрипта без указания пути.

5) Какие стандартные каталоги ситемы Linux вы знаете, и для скрипты и программ каких пользователей эти каталоги предназначены?

Некоторые стандартные каталоги системы Linux, используемые для размещения скриптов и программ:

- `/usr/bin` - каталог для скриптов и программ, предназначенных для всех пользователей системы.

- `/usr/local/bin` - каталог для локально установленных скриптов и программ, предназначенных для всех пользователей системы.

- `/bin` - каталог содержит основные исполняемые файлы, необходимые для базовой функциональности системы.

- `/sbin` - каталог содержит системные исполняемые файлы, предназначенные для администрирования системы.

- `/home/<username>/bin` - каталог для скриптов и программ, предназначенных для конкретного пользователя.

6) Что такое переменная окружения PATH, и какую роль она играет в запуске скриптов?

Переменная окружения PATH содержит список каталогов, в которых система ищет исполняемые файлы при выполнении команд в командной оболочке. Когда вы вводите команду в терминале, система ищет исполняемый файл с указанным именем в каждом каталоге, указанном в переменной PATH.

Роль переменной окружения PATH в запуске скриптов состоит в том, что если скрипт находится в одном из каталогов, перечисленных в переменной PATH, его можно запустить, просто вводя его имя в командной строке без указания полного пути. Таким образом, переменная PATH упрощает запуск скриптов, особенно если они расположены в стандартных каталогах системы.

Перенаправление ввода/вывода и потоки данных:

7) Что такое перенаправление ввода/вывода и потоки данных?

Перенаправление ввода/вывода - это процесс изменения источника или назначения данных, которые обрабатываются командами в командной оболочке. Потоки данных представляют собой последовательность байтов, которые передаются между процессами или командами.

8) Что такое потоки данных, и какие стандартные потоки данных существуют в системе? В какой системной директории располагаются их файлы?

Стандартные потоки данных в системе:

- Стандартный ввод (stdin) - представляет собой поток входных данных, который обычно считывается командой.

- Стандартный вывод (stdout) - представляет собой поток вывода данных, который обычно отображается на экране или перенаправляется в файл.

- Стандартный вывод ошибок (stderr) - представляет собой поток вывода сообщений об ошибках и предупреждений, который отображается на экране или перенаправляется в файл.

Файлы стандартного ввода, вывода и вывода ошибок обычно представлены в системной директории `/dev/`, их можно перенаправить или изменить с помощью операторов перенаправления.

9) Назовите три основновополагающие концепции операционных систем и системного программирования?

- Процессы - операционная система управляет выполнением процессов, которые представляют собой выполняемые программы в определенном состоянии.

- Потоки - в пределах процесса могут существовать несколько параллельных потоков выполнения, которые могут работать над различными задачами.

- Взаимодействие - операционные системы предоставляют механизмы для взаимодействия между процессами и потоками, такие как сигналы, разделяемая память, сокеты и т.д.

10) Для чего используются команды-фильтры? Назовите некоторые из них.

Команды-фильтры используются для обработки и модификации входных данных. Они принимают входные данные из стандартного ввода (stdin), выполняют некоторые операции и отправляют результат на стандартный вывод (stdout). Некоторые из команд-фильтров в Linux:

- `grep` - фильтр для поиска строк, соответствующих определенному шаблону.

- `sed` - команда для поиска и замены текста в потоке данных.

- `awk` - язык и утилита для обработки и анализа структурированного текста.

- `sort` - сортирует строки во входных данных.

- `cut` - выбирает определенные поля из входных данных.

- `uniq` - удаляет повторяющиеся строки из входных данных.

- `wc` - подсчитывает количество строк, слов и символов во входных данных.