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

Черняк П.С.

Принял**:** Давыденко Н.В.

Минск 2023

# Лабораторная работа № 3. Командный язык и скрипты Shell

# 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?

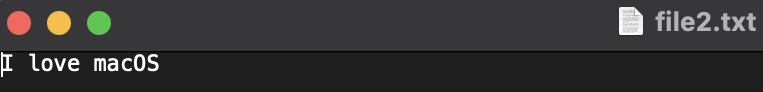
Напишите скрипт поиска местонахождения командных интерпретаторов вашей системы. Также определите какой интерпретатор используется сейчас в вашей системе?

**Решение**

**Задание 1.1**

Создаем два файла текстовых либо через терминал, либо вручную и потом можно использовать команду cp file1 file2 или cat file1> file2

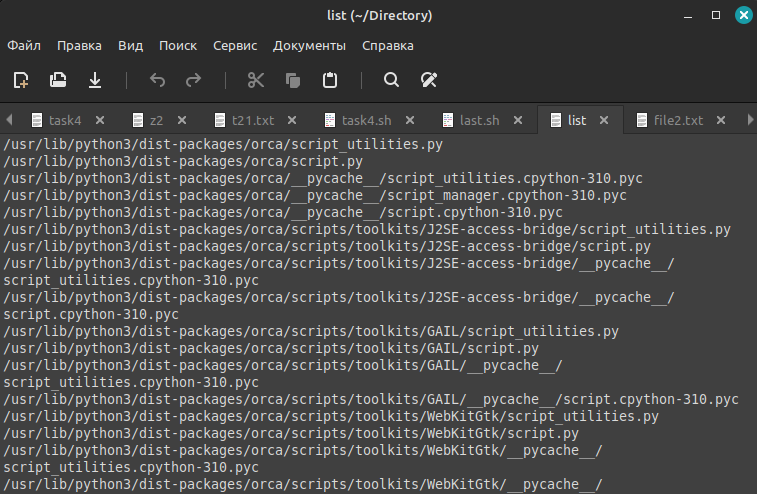


В результате получаем 

Второй вариант выполнения данного задания вместо cat в файле скрипта использовать cp, результат будет одинаковым

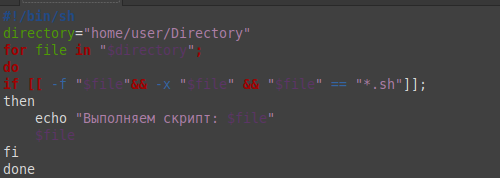
**Задание 1.2**

1. find / -type f -name "script\*" > list



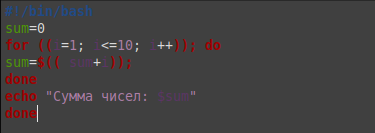
**Задание 1.3**

Создаем скрипт для нахождения и выполнения скрипта в директории



**Задание 1.4**

Создаем файл и пишем скрипт подсчитывающий сумму от 1 до 10

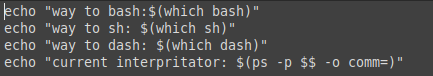


Делаем скрипт исполняемым и запускаем его

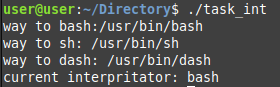


**Задание 1.5**

Создаем скрипт для поиска местонахождения командных интерпретаторов системы



Делаем скрипт исполняемым и запускаем его



**Задание 2**

1. Используйте текст первого примера, заменить некоторые фамилии на

фамилии членов вашей команды (например, Nutrina заменить на

Stankewich).

2. Отсортируйте данные файла так, чтобы он не поменял свое собствен-

ное имя. (Решение. Нужно предварительно создать промежуточный от-

сортированный файл t11.txt. Затем выполнить нужные команды).

3. Подсчитайте число одинаковых слов в обоих файлах. Для подсчета

числа слов в файле используйте команду: wc -w file1.

4. Написать скрипт, который создает отсортированный файл, содержащий

слова из двух файлов, исключая их общую часть одинаковых слов.

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

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

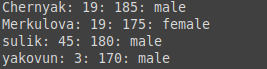
7. Вывести отсортированный список имен файлов, начальная часть имени есть script.

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

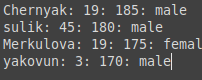
9. Вывести упорядоченный список имен файлов, в именах которых содержится символ подчеркивания, например, script\_sort

**Решение**

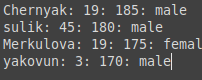
**Задание 2.1** Создадим файл peoples



**Задание 2.2** Создаем новый файл и сортируем данные в промежуточный файл

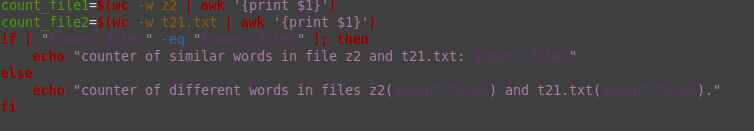


Полученный результат:



**Задание 2.3**

Создаем скрипт



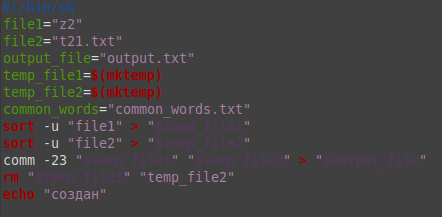
Делаем его исполняемым и запускаем

Полученный результат:

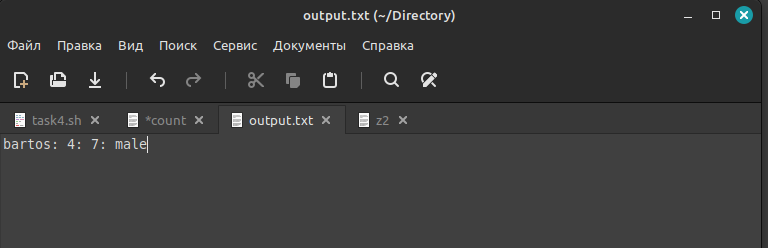
****

**Задание 2.4**

Создаем скрипт



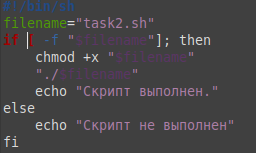
Результат выполнения



**Задание 3**

* + - 1. Напишите скрипт запуска другого скрипта.

Решение



Результат выполнения



1. Расположите скрипт в одной из стандартных директорий системы, и запустите его из домашней директории пользователя.



Вывод*:* В ходе выполнения лабораторной работы изучили основные команды, для написания скриптов.

Контрольные вопросы:

1. Какие способы запуска выполнения скрипта вы знаете? Как запустить скрипт, на который отсутствуют права исполнителя (запуска)

С помощью команды ./ или прямо через интерпретатор(bash, shell, dash). Если нет прав, нужно прописать команду chmod +x filename (изменения доступа) и после этого запустить файл, способами описанным выше.

1. Что такое перенаправление ввода/вывода и потоки данных?

Стандартный ввод Стандартный входной поток обычно переносит данные от пользователя к программе. Программы, которые предполагают стандартный ввод, обычно получают входные данные от устройства типа клавиатура. Стандартный ввод прекращается по достижении EOF (конец файла), который указывает на то, что данных для чтения больше нет. Реальное значение EOF является отрицательным числом, зависящим от системы (в основном −1), что гарантирует несовпадение с кодом символа. Чтобы прервать работу терминала мы можем нажать сочетания клавиш Ctrl+D. Это означает, что работа терминала достигла конца файла, EOF. А точнее мы сами ее туда направили. Стандартный вывод Стандартный вывод записывает данные, сгенерированные программой. Когда стандартный выходной поток не перенаправляется в какой-либо файл, он выводит текст на дисплей терминала. Для работы со стандартным выводом используются следующие команды (утилиты), так же представленные в виде файлов в директории /bin

Стандартный ввод при работе пользователя в терминале передается через клавиатуру. Стандартный вывод и стандартная ошибка отображаются на дисплее терминала пользователя в виде текста. Данные выводятся на экран и считываются с клавиатуры, так как стандартные потоки по умолчанию ассоциированы с терминалом пользователя. Ввод и вывод распределяется между тремя стандартными потоками данных: stdin, stdout, stderr. Потоки можно подключать к чему угодно: к текстовым файлам, программам и даже устройствам. В командном интерпретаторе такая операция называется перенаправлением. Команды с символами > или < означают перезапись существующего содержимого файла: > — стандартный вывод; < — стандартный ввод; 2> — стандартная ошибка. Команды с символами >> или << не перезаписывают существующее содержимое файла, а присоединяют данные к нему: >> — стандартный вывод; << — стандартный ввод; 2>> — стандартная ошибка. Примеры перенаправления с использованием файлов: • < file — использовать файл как источник данных для стандартного потока ввода. • > file — направить стандартный поток вывода в файл. Если файл не существует, он будет создан, иначе перезаписан сверху.