Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа №2

По дисциплине «Операционные системы»

На тему «Знакомство с командной оболочкой операционной системы»

Выполнила:

Студентка 3 курса 7 группы

Гибадулина Алина Фархадовна

Преподаватель: асс. Бернацкий П.В.

2025, Минск

# 1 Введение

**Цель работы:** Получение практических навыков в работе с командными оболочками такими как cmd, PowerShell, bash.

Для поиска необходимых команд и информации в целом можно использовать ресурс ss64.com

# 2 Основная часть

## 2.1. Выполнение заданий для Windows

Переходим в домашний каталог нашего пользователя (созданного в прошлой лабораторной работе) и создаем каталог OS2.

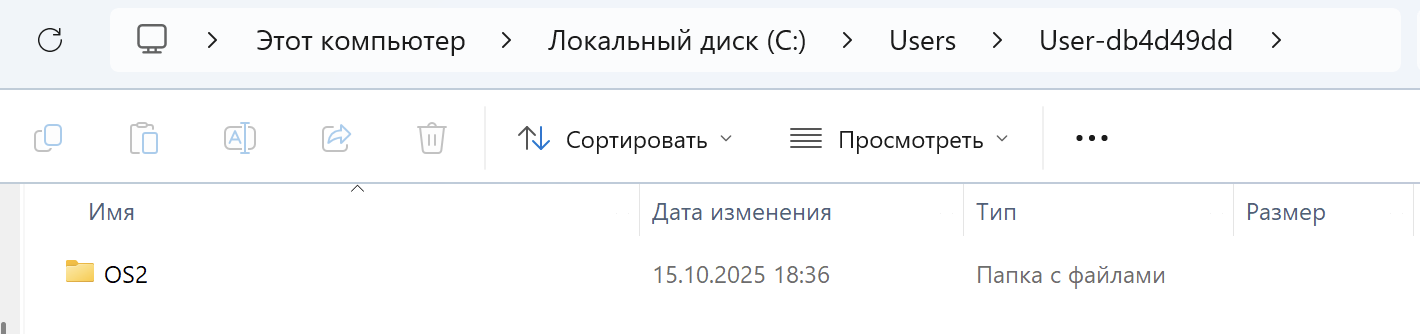


Рисунок 2.1.1 – Создание каталога OS2

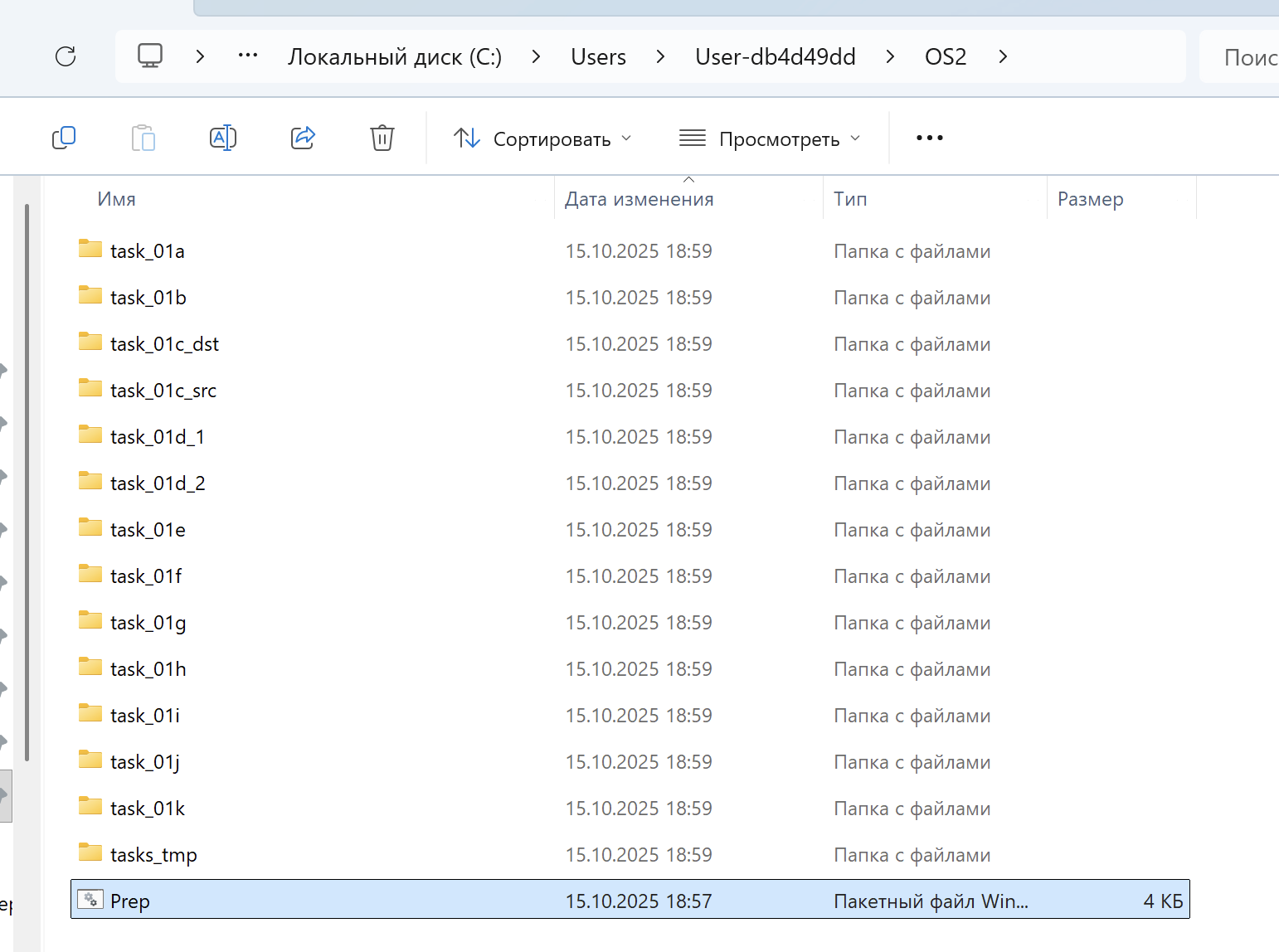
Выполнить в нём бат-файл prep.bat (лежит вместе с условием на diskstation).

Рисунок 2.1.2 – Результат запуска файла prep.bat

Открываем cmd в данном каталоге путём указания слова cmd в адресной строке проводника.

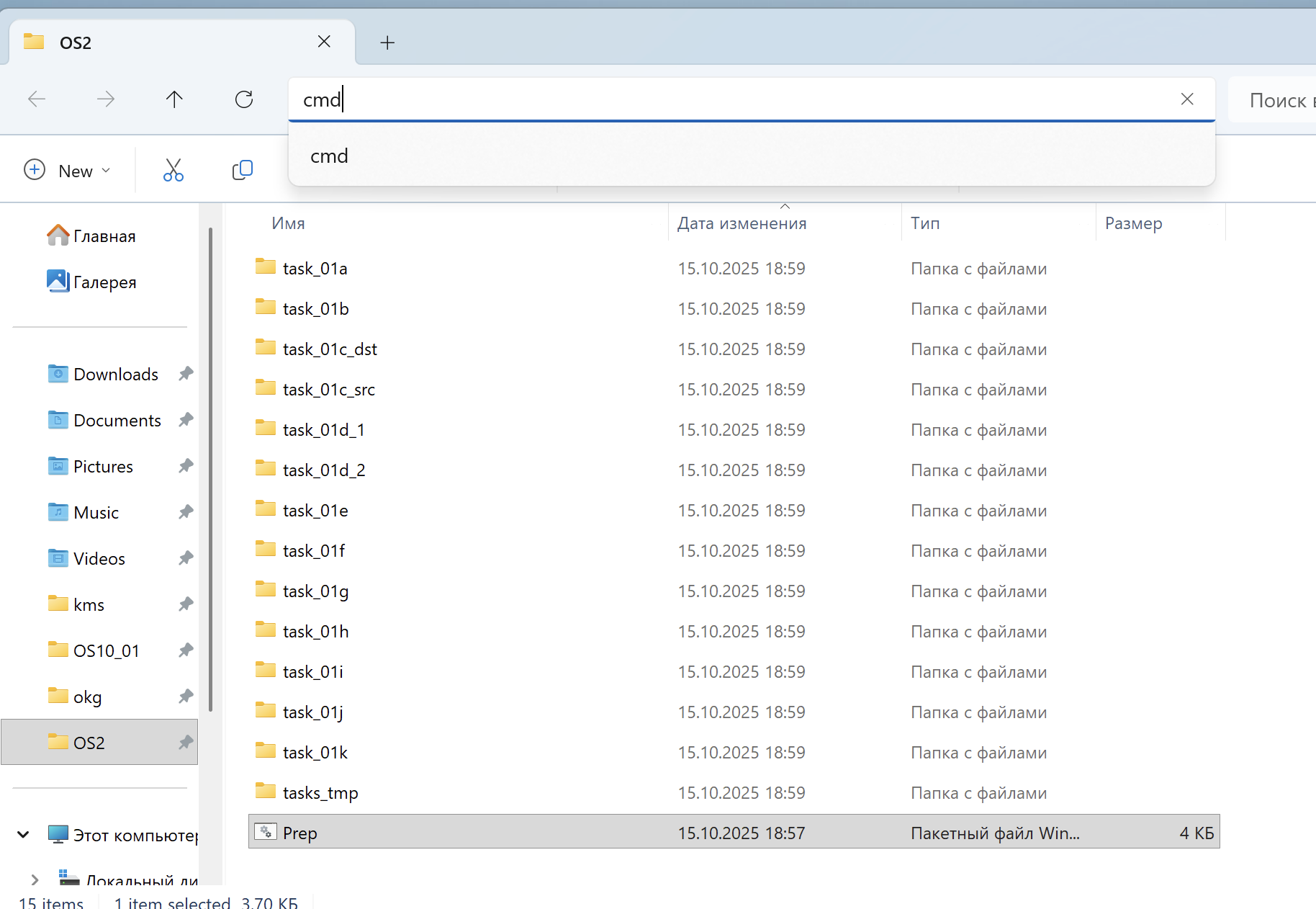


Рисунок 2.1.3 – Запуск cmd из проводника

# 2.1.1 Переходы по каталогам (cd)

На основе данных файлов и каталогов выполняем следующее (в скобках указана используемая в задании команда:

* (cd) перейти в каталог task\_01i/Русский с использованием абсолютного пути

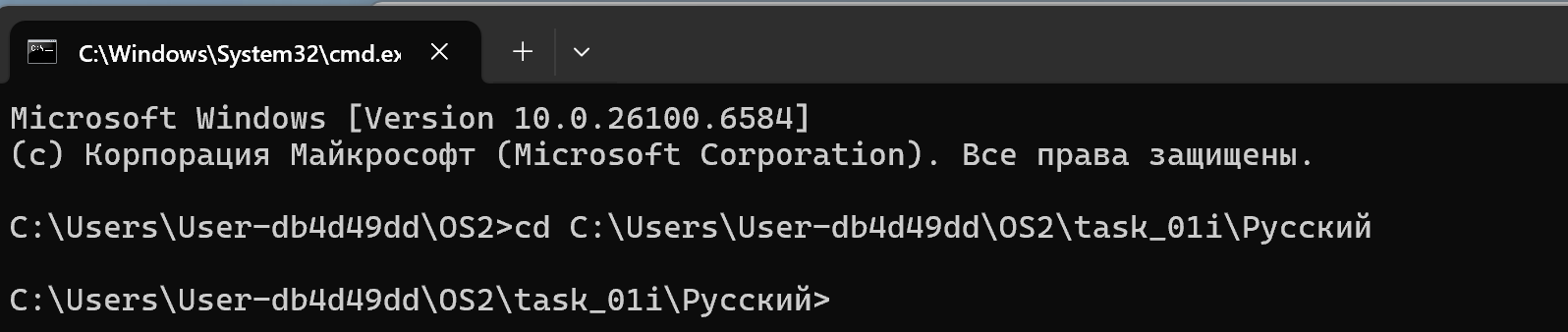


Рисунок 2.1.1.4 – Переход в каталог task\_01i/Русский с использованием абсолютного пути

* (cd) перейти в каталог taks\_01i/English с использованием относительных путей

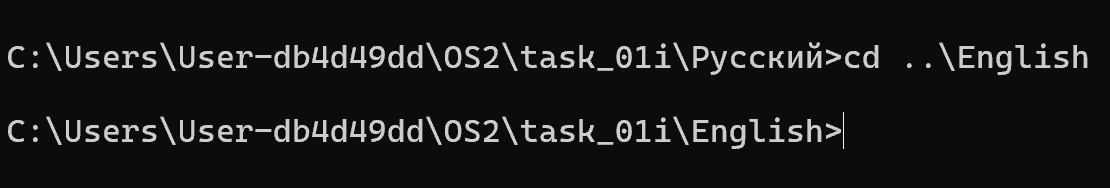


Рисунок 2.1.1.4 – Перехор в каталог task\_01i/English с использованием относительного пути

## 2.1.2 Копирование файлов (copy, xcopy)

* (copy) cкопируем все файлы, имеющие расширение txt из каталога tasks\_tmp в каталог task\_01a

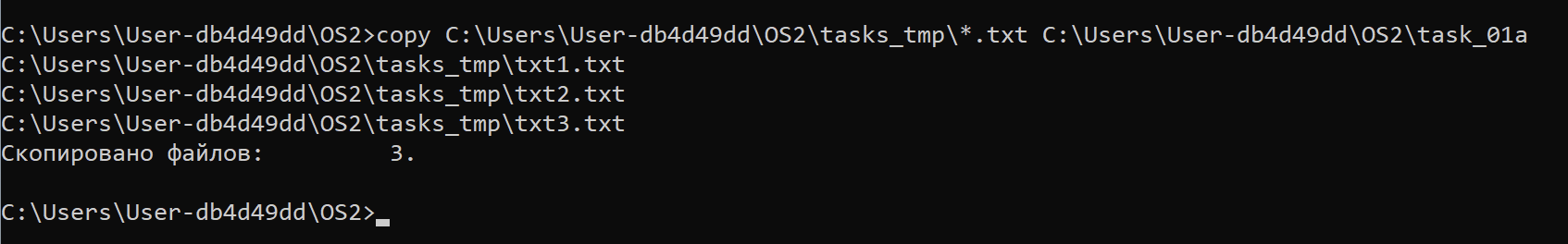


Рисунок 2.1.2.1 – Копирование текстовых файлов из временной директории

* (copy) повторяем туже операцию, но при этом команда должна запрашивать подтверждение на перезапись существующих файлов

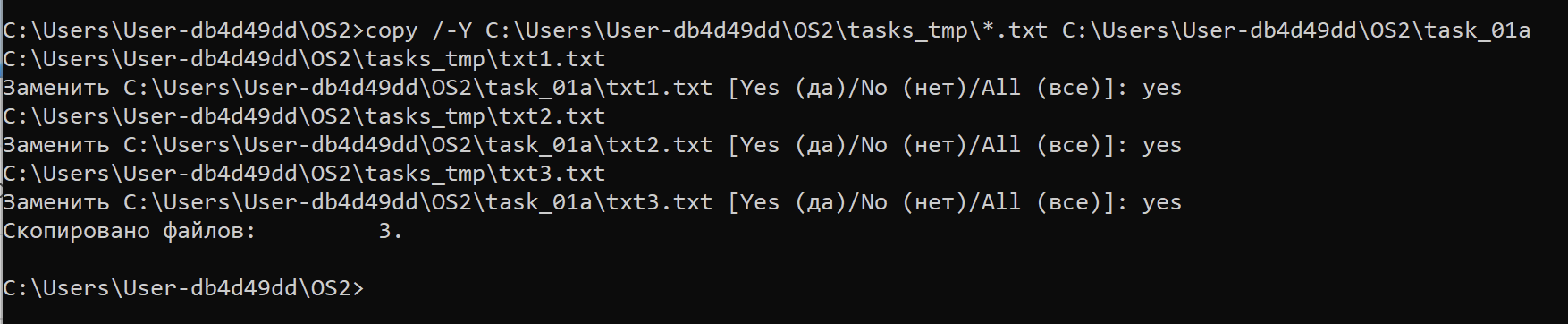


Рисунок 2.1.2.2 – Пример копирования с ручным подтверждением перезаписи

* (copy) скопировать файл из каталога task\_01a в каталог task\_01a, но с новым именем

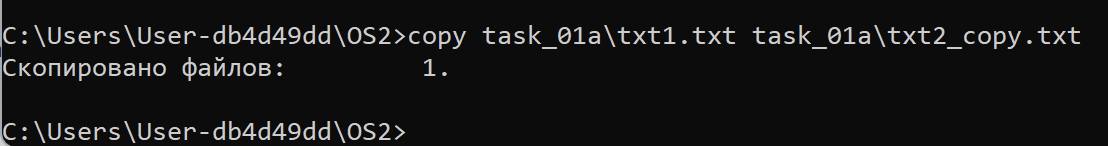


Рисунок 2.1.2.3 – Создание копии файла с другим названием

* (copy) объединить файлы txt1.txt и txt2.txt в файл merged.txt

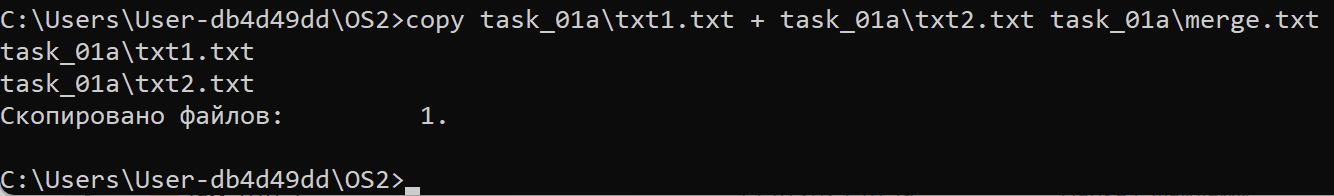


Рисунок 2.1.2.4 – Соединение содержимого двух файлов в один

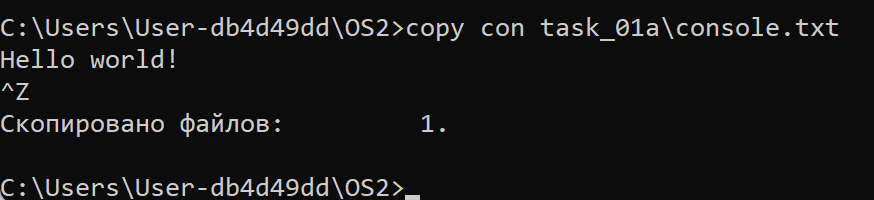
* (copy) скопировать текст, введенный в консоль в файл с именем console.txt 

Рисунок 2.1.2.5 – Запись текста напрямую из консоли в файл

* (copy) добавить в конец файла txt3 информацию, вводимую с консоли

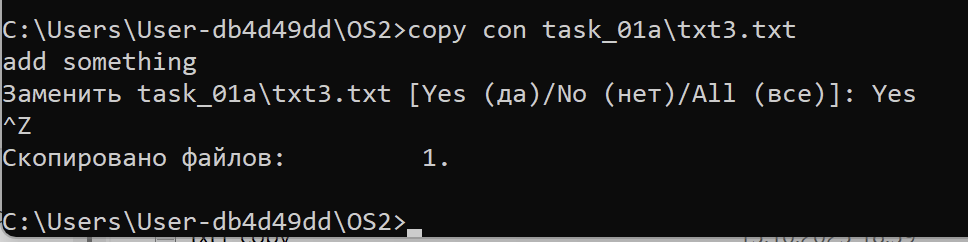


Рисунок 2.1.2.6 – Дополнение содержимого файла пользовательским вводом

* (xcopy) скопировать все файлы, каталоги и подкаталоги, включая пустые, а также скрытые и системные из task\_01b в task\_01b\_dst\_1

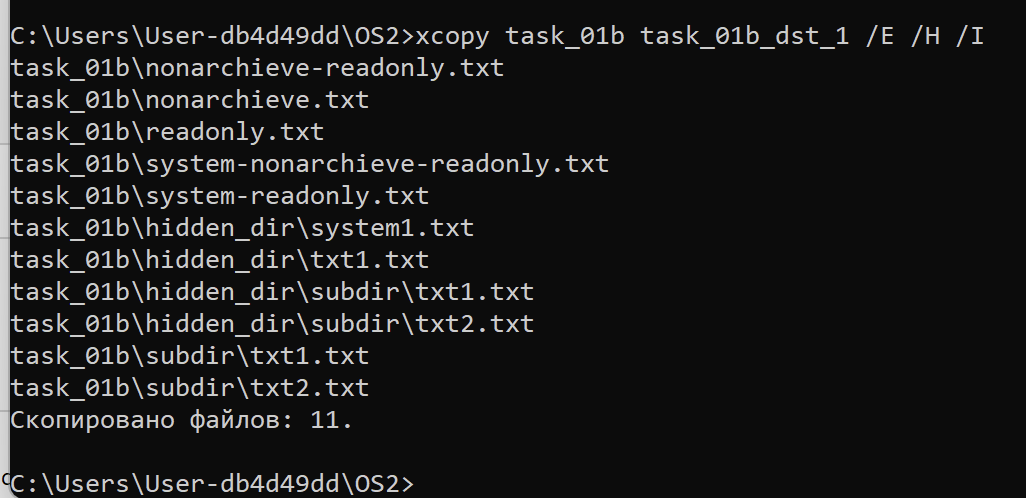


Рисунок 2.1.2.7 – Полное копирование структуры папки с сохранением скрытых и системных объектов

* (xcopy) скопировать все файлы с атрибутами «архивный» и «только для чтения» с сохранением этого атрибута для файлов-результатов из task\_01b в task\_01b\_dst\_2

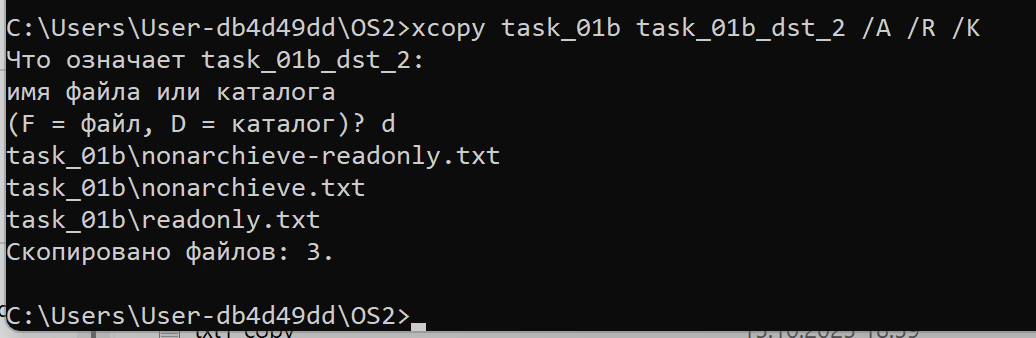


Рисунок 2.1.2.8 – Фильтрация и копирование файлов по заданным атрибутам

## 2.1.3 Перемещение и удаление (move, del, rd, replace)

* (move) переместить все текстовые файлы с расширением .txt из каталога task\_01c\_src в каталог task\_01c\_dst

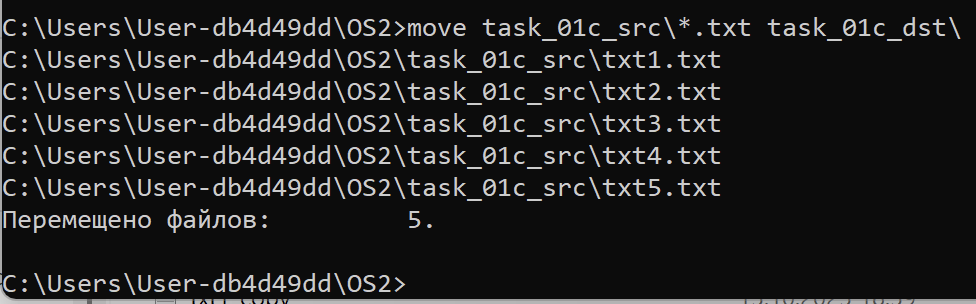


Рисунок 2.1.3.1 – Перенос текстовых файлов из одного каталога в другой

* зайти в каталог task\_01d\_2 и удалить текст в файлах

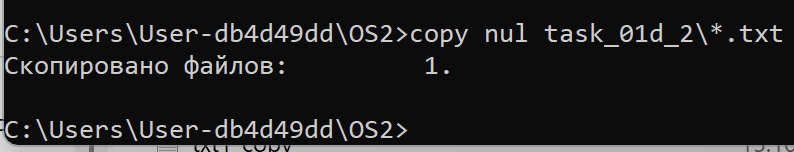


Рисунок 2.1.3.2 – Удаление содержимого файлов без их физического удаления

* (replace) заменить файлы в каталоге task\_01d\_2 файлами, расположенными в каталоге task\_01d\_1

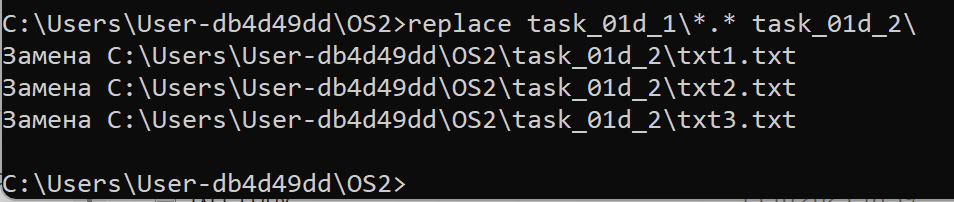


Рисунок 2.1.3.3 – Автоматическая замена содержимого папки актуальными файлами

* (ren) изменить расширение всех файлов в каталоге task\_01e на .bmp

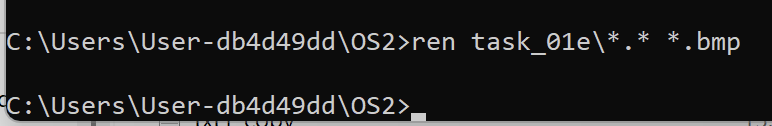


Рисунок 2.1.3.4 – Массовое изменение формата на BMP

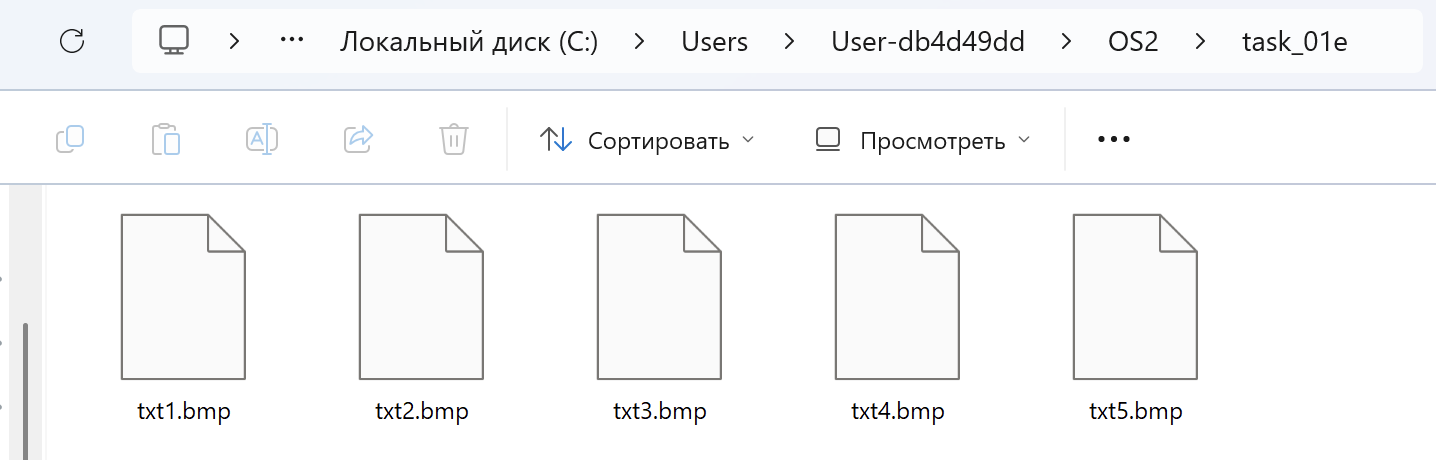


Рисунок 2.1.3.5 – Результат массового изменнеия формата на BMP

* (ren) изменить имя файла txt1.bmp на «имя вашего пользователя».Ren

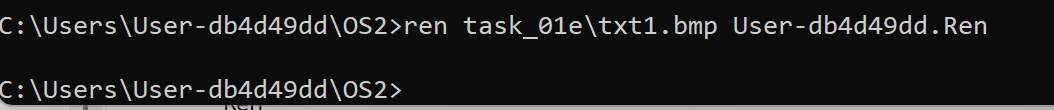


Рисунок 2.1.3.6 – Переименование файла в соответствии с учётной записью пользователя

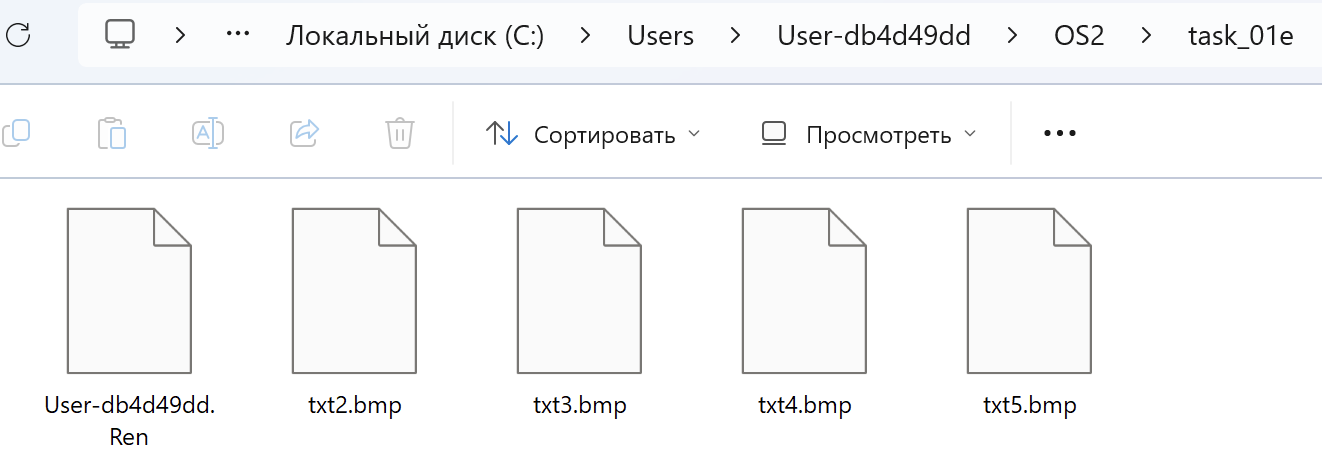


Рисунок 2.1.3.7 – Результат переименования файла в соответствии с учетной записью пользователя

* (fc) сравнить два файла в каталоге task\_01f и сохранить результат сравнения в result.txt



Рисунок 2.1.3.8 – Сравнение содержимого файлов с сохранением отчёта

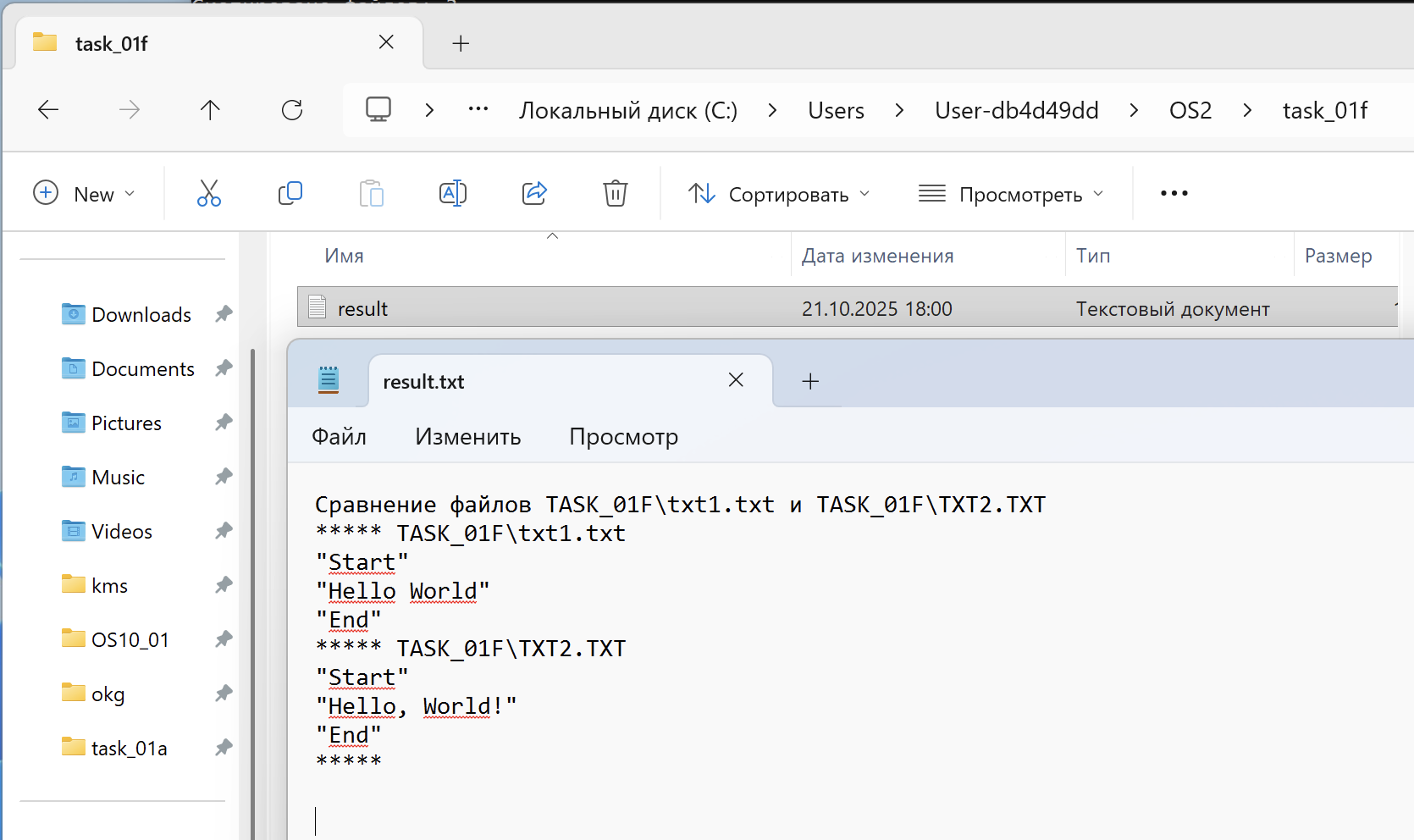


Рисунок 2.1.3.9 – Результат сравнения содержимого текстовых файлов и сам отчет

* (del) удалить файл txt1 из каталога task\_01g при этом запросив подтверждение на удаление



Рисунок 2.1.3.9 – Удаление файла с запросом подтверждения

* (del) удалить все файлы с атрибутом «Системный» из каталога task\_01g

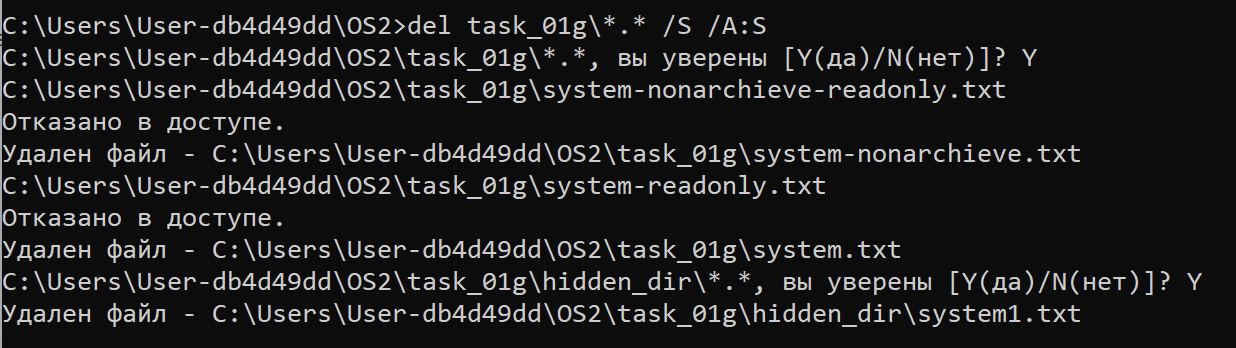


Рисунок 2.1.3.10 – Удаление объектов, имеющих системный атрибут

* (del) удалить все файлы из каталога task\_01g так, чтобы остались только файлы с атрибутом «Только для чтения»

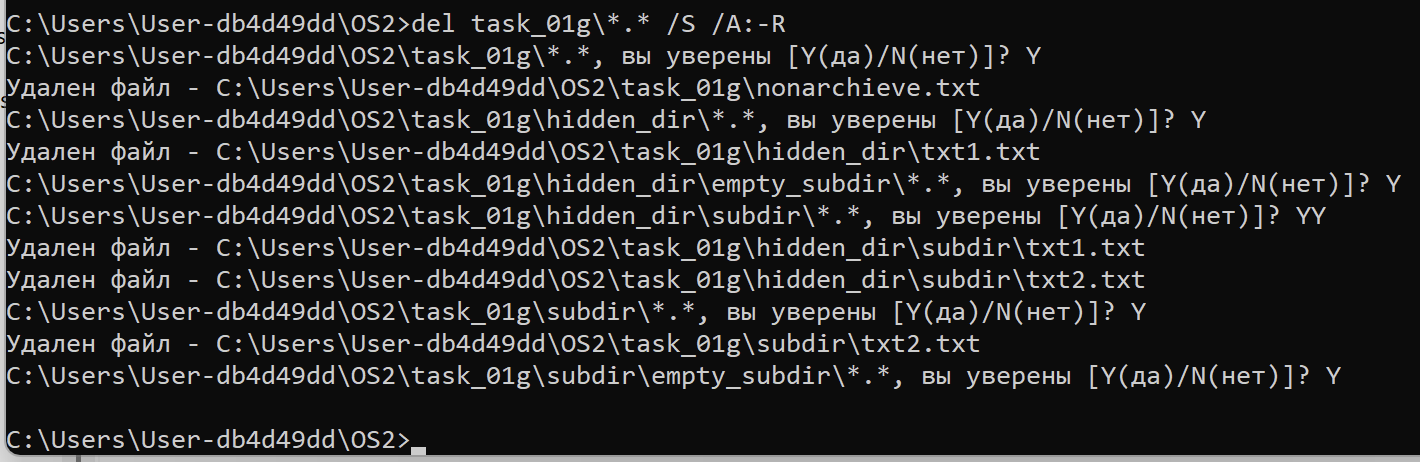


Рисунок 2.1.3.11 – Удаляем все файлы оставляем лишь файлы с правами «только чтение»

* (rd) удалить все оставшиеся файлы и дочерние каталоги из каталога task\_01g

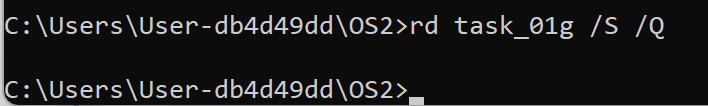


Рисунок 2.1.3.12 – Полная очистка папки вместе с вложенными каталогами

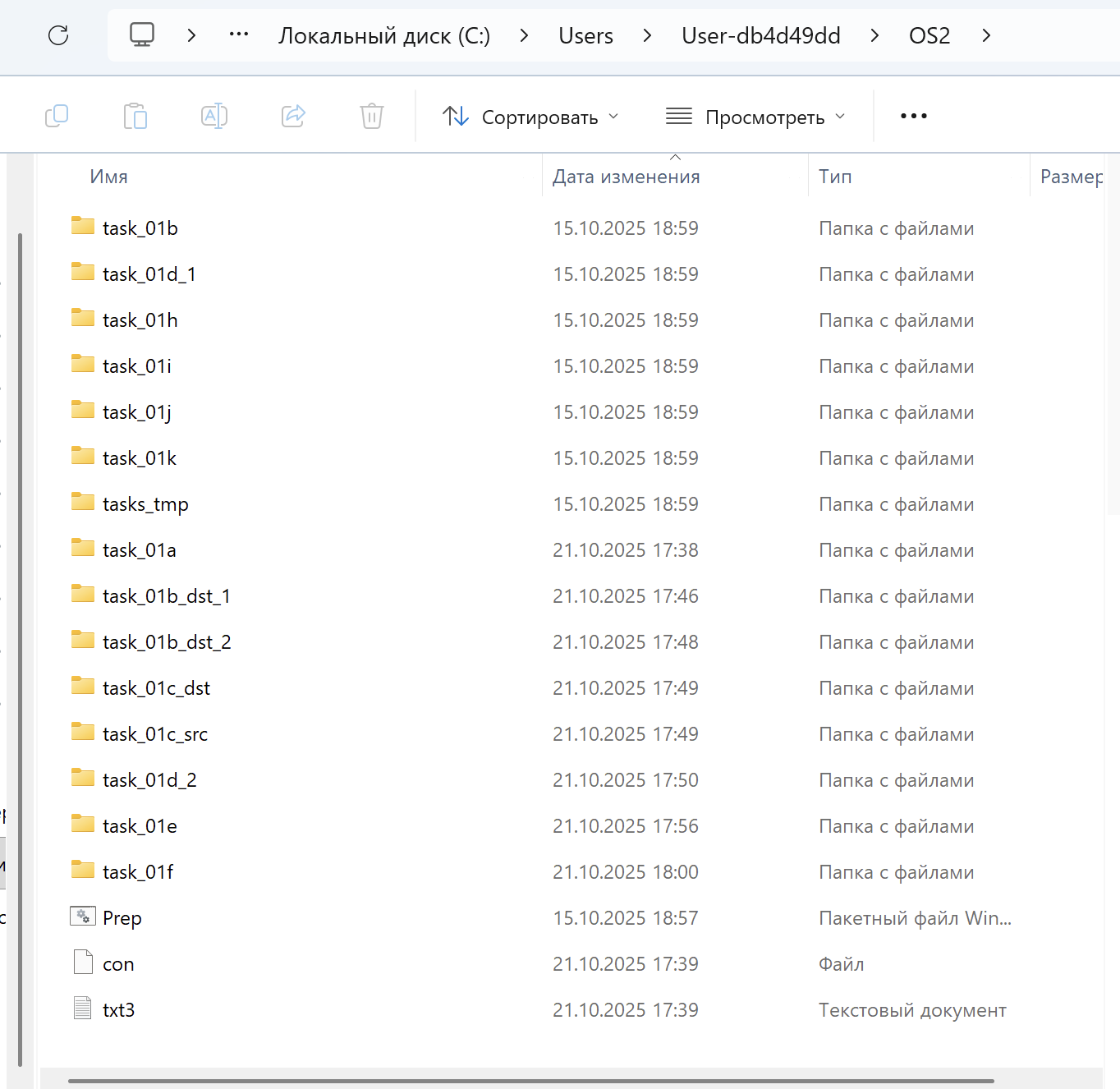


Рисунок 2.1.3.13 – Результат полной очистки папки

## 2.1.4 Просмотр содержимого (dir)

* (dir) постранично вывести содержимое каталога C:\Windows, включая вложенные каталоги и файлы (постранично – информация выводится частями с ожиданием нажатия клавиши для вывода следующей части информации)

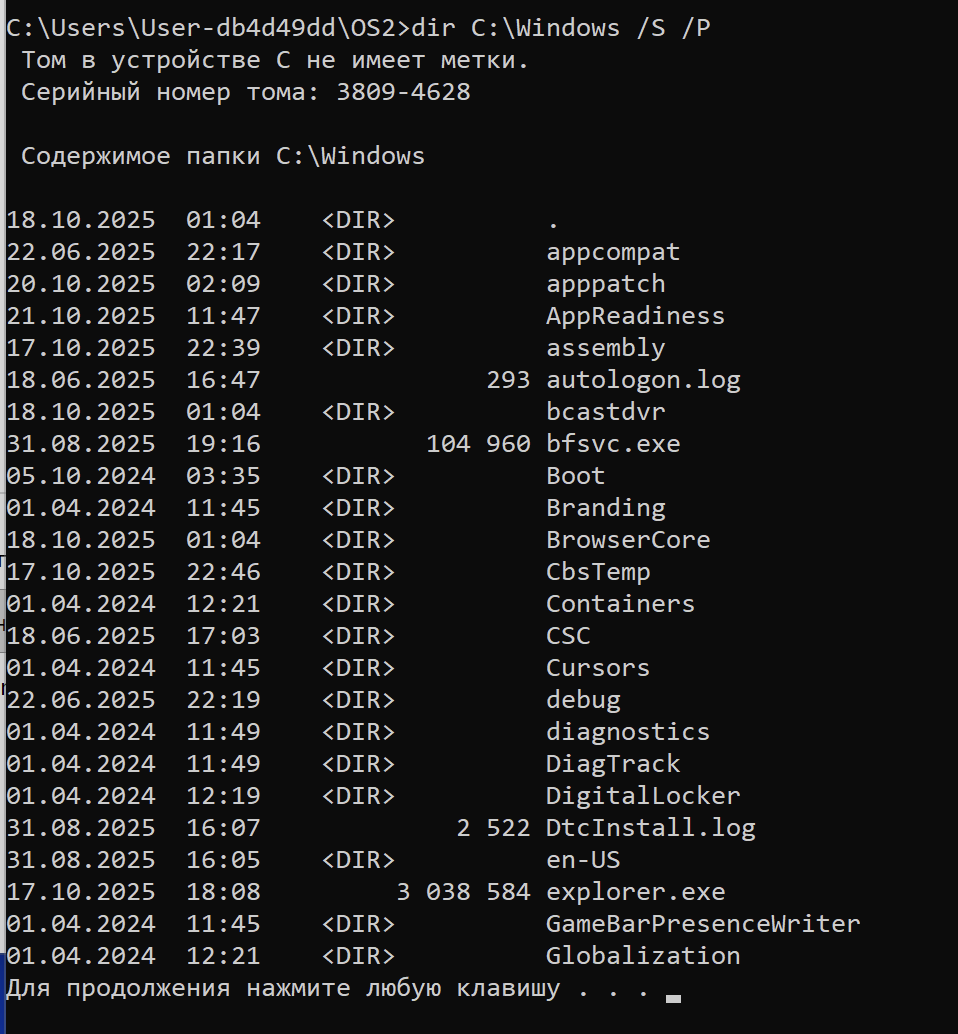


Рисунок 2.1.4.1 – Постраничный просмотр структуры Windows

* (dir) постранично вывести все каталоги и файлы, включая вложенные подкаталоги и файлы каталога C:\Users\«ваш пользователь» в отсортированном по алфавиту порядке с записью результата в файл user\_files.txt в каталоге task\_01h

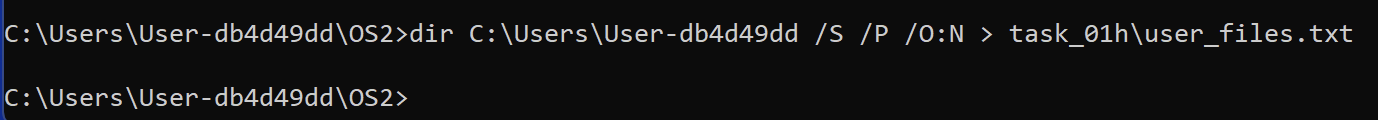


Рисунок 2.1.4.2 – Создание списка файлов пользователя с алфавитной сортировкой

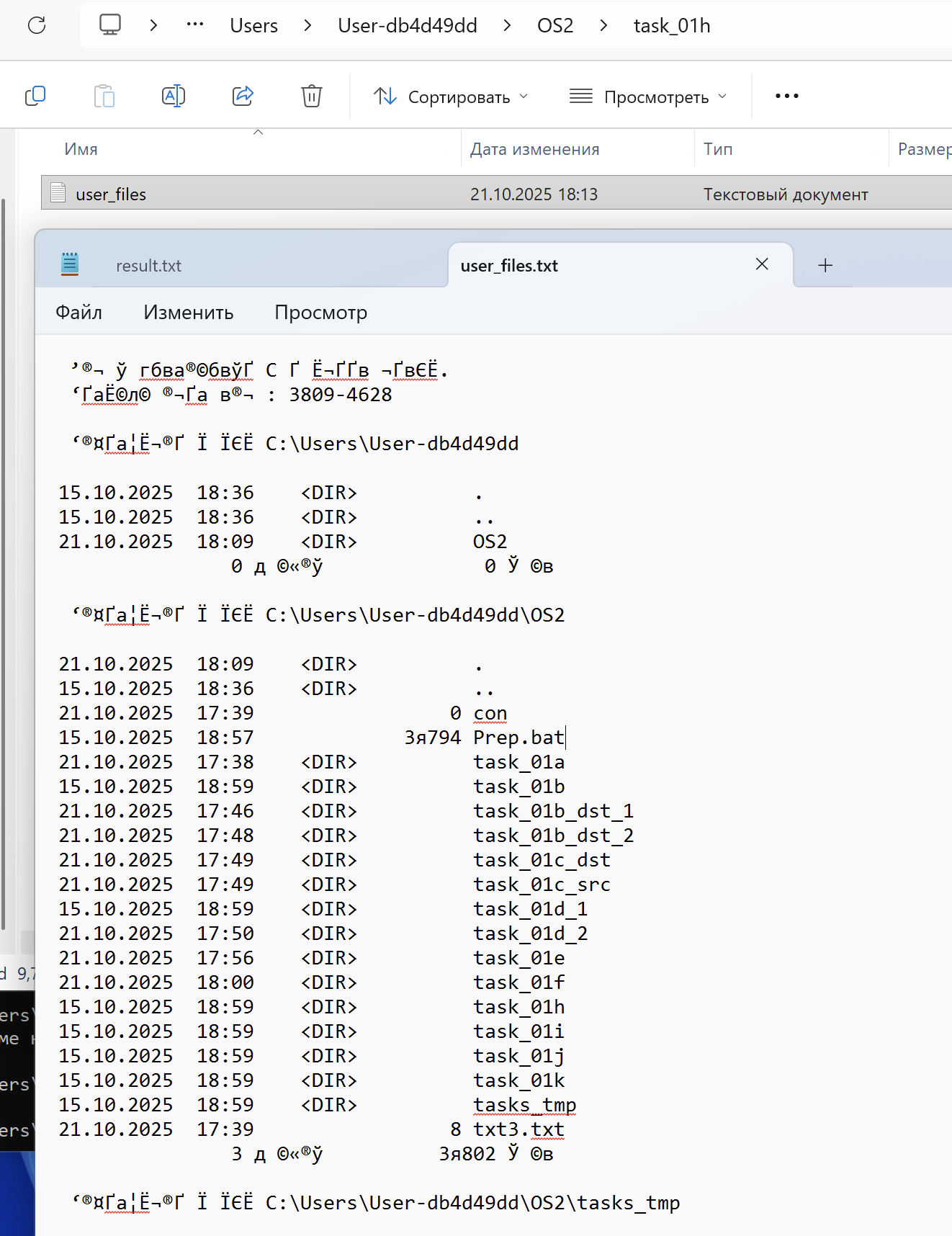


Рисунок 2.1.4.3 – Результат создания списка файлов пользователя

## 2.1.5 Создание и удаление каталогов (md, rd)

* (md) создать каталог task\_01j в вашем каталоге OS2

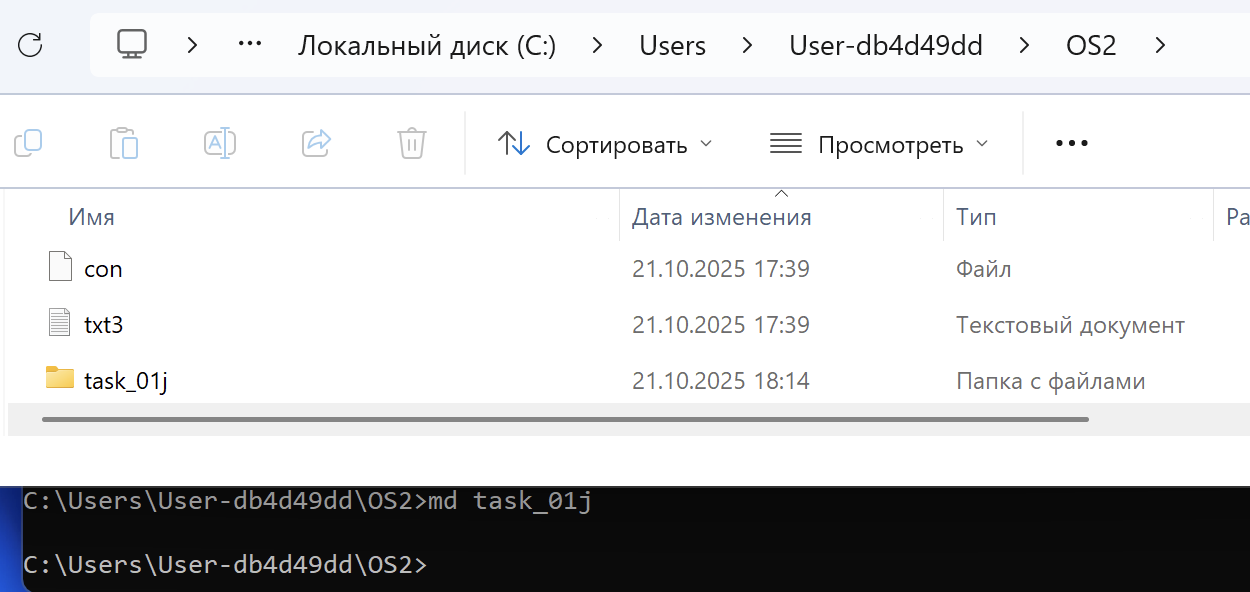


Рисунок 2.1.5.1 – Создание дополнительной директории внутри OS2

* (md) создать иерархию каталогов task\_01j\sub\dir\full\path

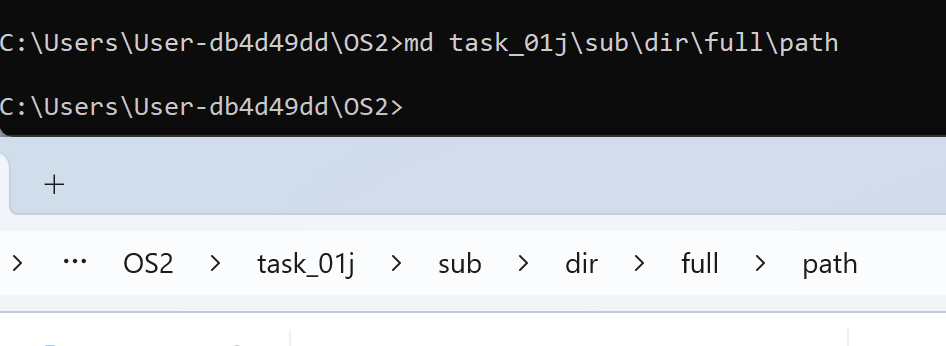


Рисунок 2.1.5.2 – Формирование вложенной структуры папок

* (rd) удалить пустой каталог \task\_01k\empty\_subdir

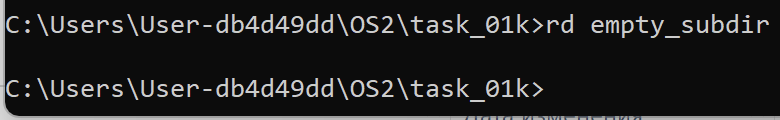


Рисунок 2.1.5.3 – Удаление заранее подготовленной пустой директории

* (rd) удалить не пустой каталог \task\_01k\subdir

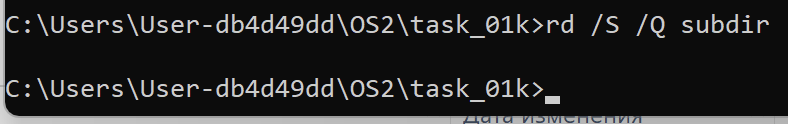


Рисунок 2.1.5.4 – Удаление каталога вместе с его содержимым

## 2.1.6 Использование операторов перенаправления вывода (>, >>)

Выполняем следующие задания с использованием различных команд, но с обязательным использованием таких операторов как перенаправление вывода (>, >>), конвейеризации (|) и соединения (&).

Вернуться в папку OS2.

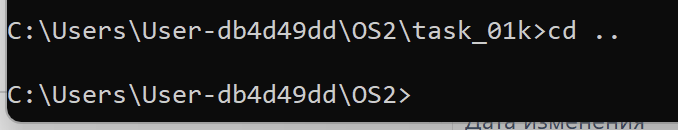


Рисунок 2.1.6.1 – Возврат в рабочий каталог OS2

Создать текстовый файл со списком студентов группы и их номерами книжек (например, Иванов 1) используя команду type и ввод с консоли.

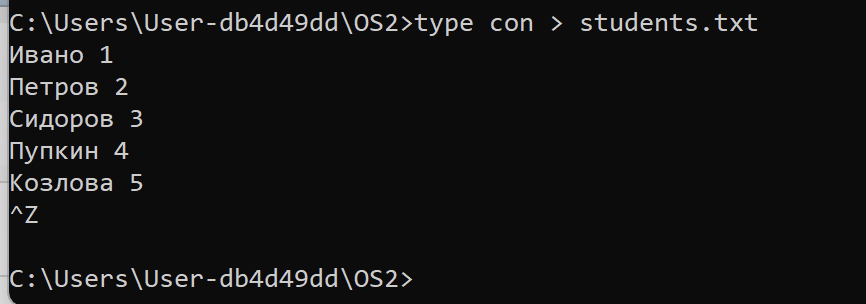


Рисунок 2.1.6.2 – Создание текстового файла со списком студентов через ввод с клавиатуры

Выполнить поиск студента в файле по его фамилии.

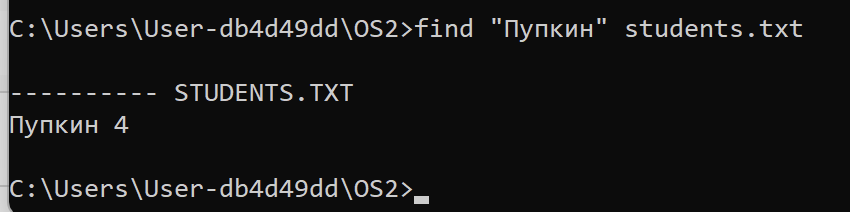


Рисунок 2.1.6.3 – Поиск фамилии студента в общем списке

Добавить ещё студентов в конец файла.

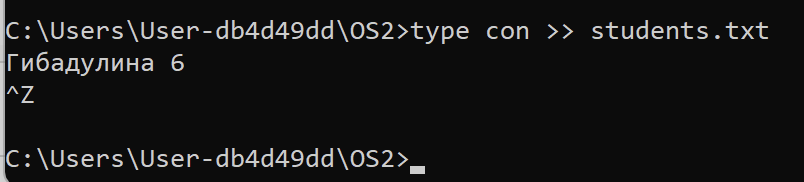


Рисунок 2.1.6.4 – Дополнение списка студентов новыми записями

Выведите список студентов в отсортированном порядке и после этого выйдите из папки OS2.

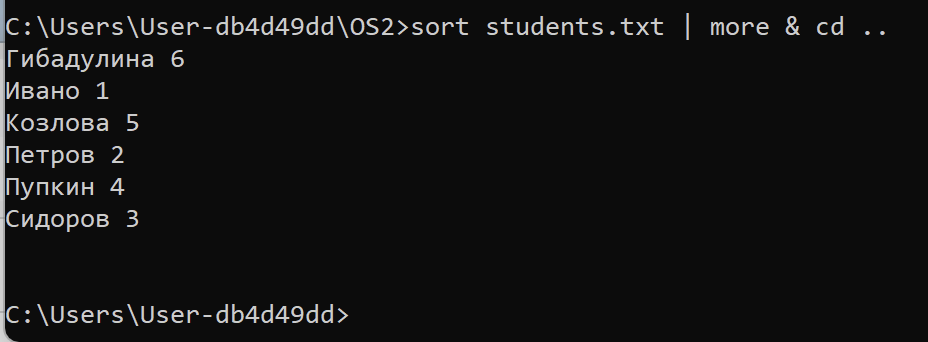


Рисунок 2.1.6.5 – Сортировка студентов по алфавиту с последующим выходом из папки

Выведите версию BIOS (systeminfo) через файл (результат должен выводиться на консоль, файл лишь промежуточный буфер).

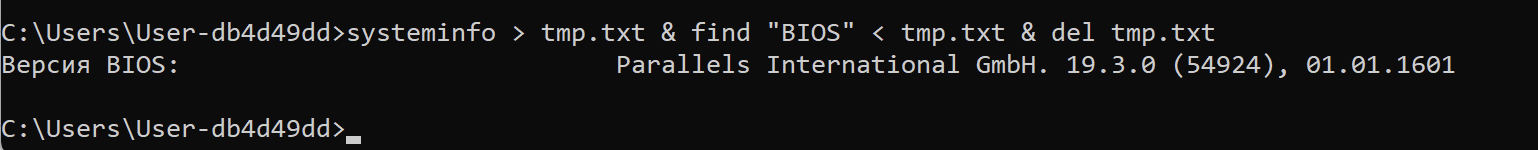


Рисунок 2.1.6.5 – Определение версии BIOS через временный файл и вывод результата на экран

## 2.1.7 Создание файлов .bat

Создать bat-файл bat\_01 который выводит следующую информацию, используя системные переменные (ss64): имя текущего пользователя, текущие дату и время, имя компьютера.

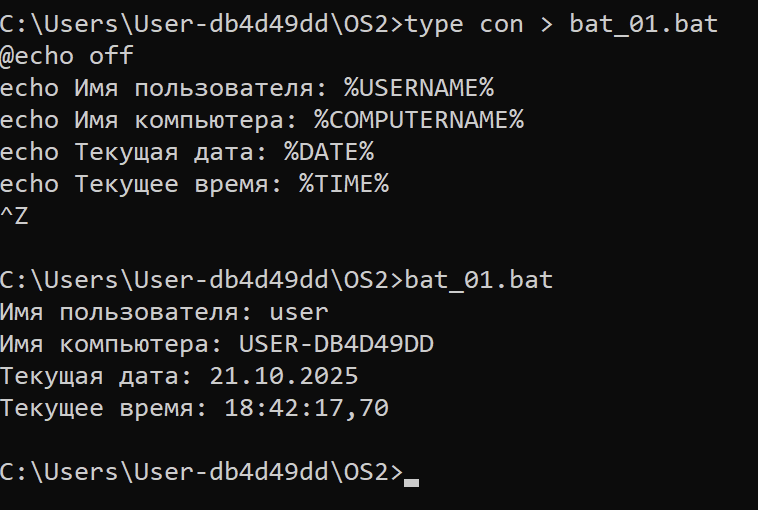


Рисунок 2.1.7.1 – Создание и запуск файла выводящего имя пользователя, дату, время и имя компьютера

Создать bat-файл bat\_02 который выводит следующую информацию, используя расширения параметров (ss64): имя текущего бат-файла, время последней модификации и полный путь к текущему бат-файлу.

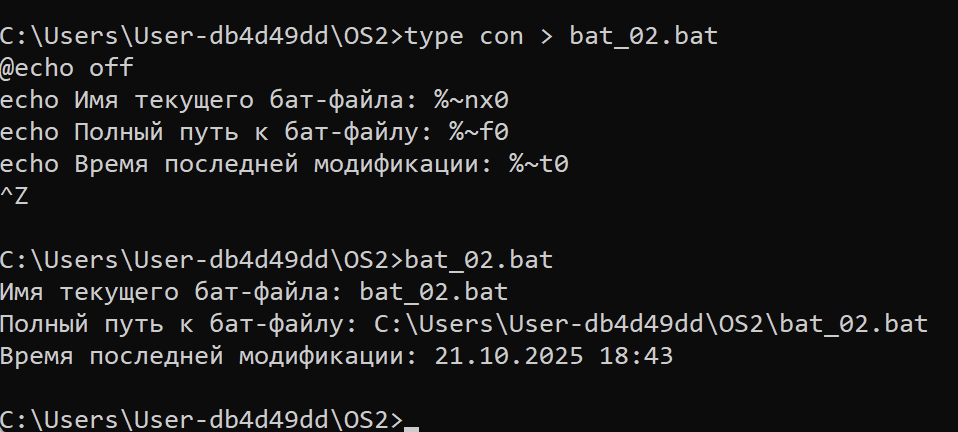


Рисунок 2.1.7.2 – Создание и запуск файла выводящего информацию о себе: имя, путь и время модификации

Создать bat-файл bat\_03 который выводит следующую информацию, используя параметры (ss64): строку со всеми переданными параметрами, а также первые 4 параметра по отдельности.

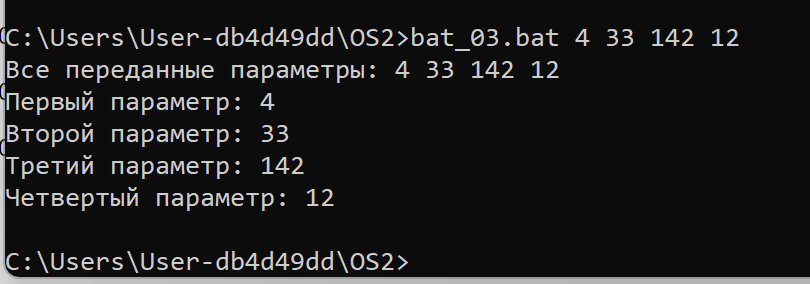
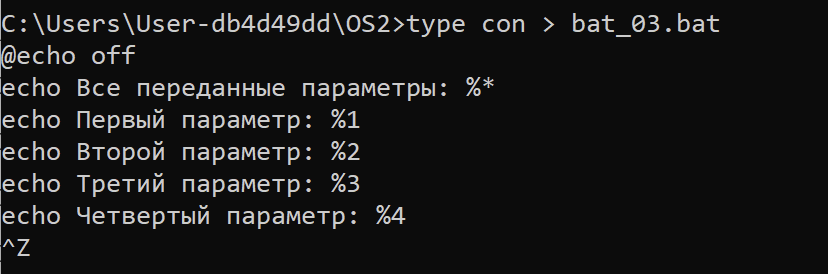


Рисунок 2.1.7.3 – Демонстрация передачи параметров в бат-файл: полный список и первые четыре отдельно

Создать bat-файл bat\_04 который выводит следующую информацию (ss64): результаты следующих математических выражений a-b, a+b, c/b, c\*b, (a-b)\*(b-a), где a,b,c – 1,2 и 3 параметры передаваемые бат-файлу.

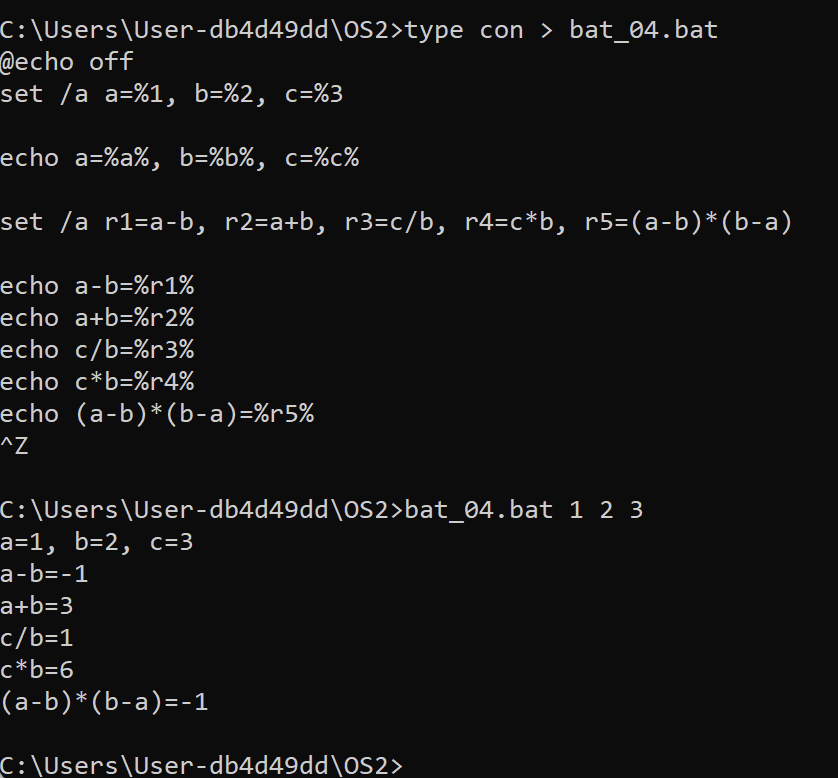


Рисунок 2.1.7.4 – Вычисления с параметрами

Создать bat-файл bat\_05 который работает следующим образом:

* вызывается в формате bat\_05 {режим работы} {имя файла}.
* поддерживает режимы работы – создать, удалить.
* имя файла – любое.
* если не задан ни режим, ни файл, то выводится подсказка по использованию файла.
* если не задано имя файла, то выводится соответствующее сообщение.
* режим «создать» – создаёт файл, указанный вторым параметром, но если файл уже существует, то выводится соответствующее сообщение.
* режим «удалить» – удаляет файл, указанный вторым параметром, но если файл отсутствует, то выводится соответствующее сообщение.

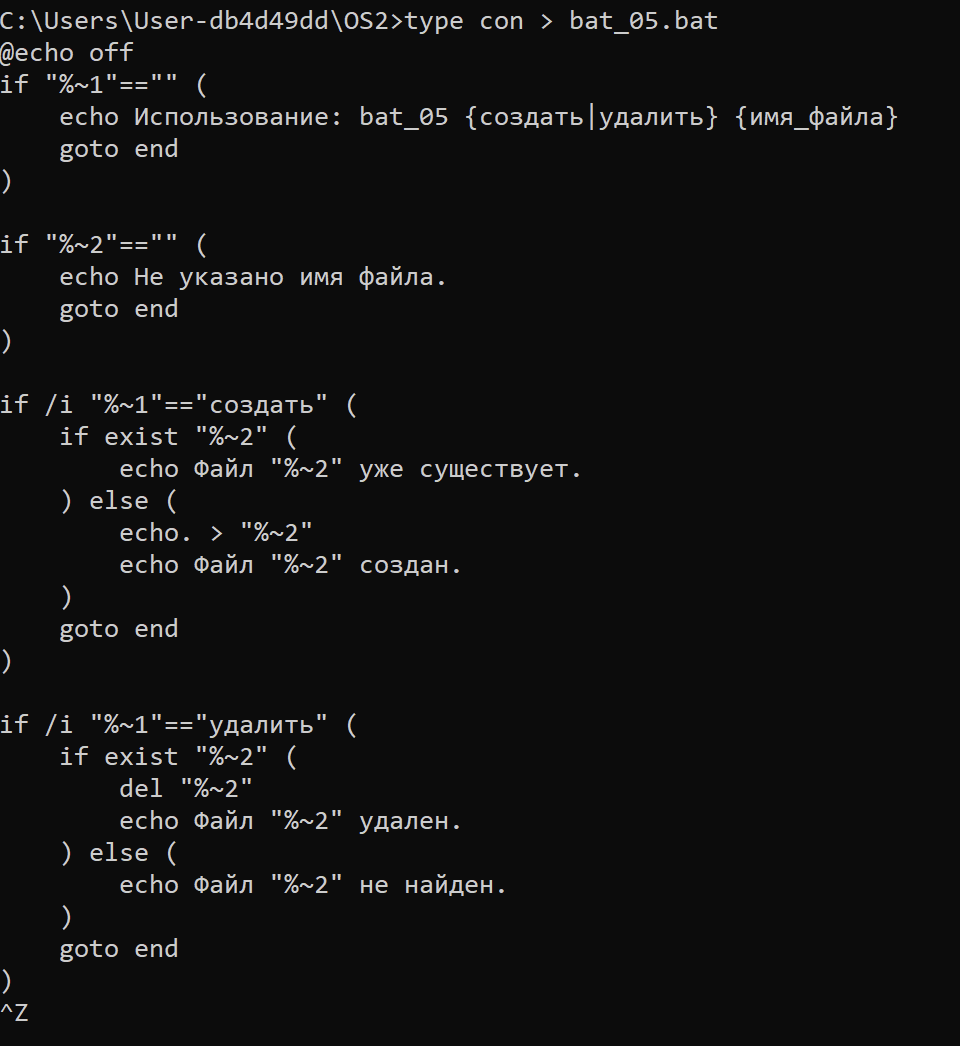


Рисунок 2.1.7.5 – Скрипт файла bat\_05

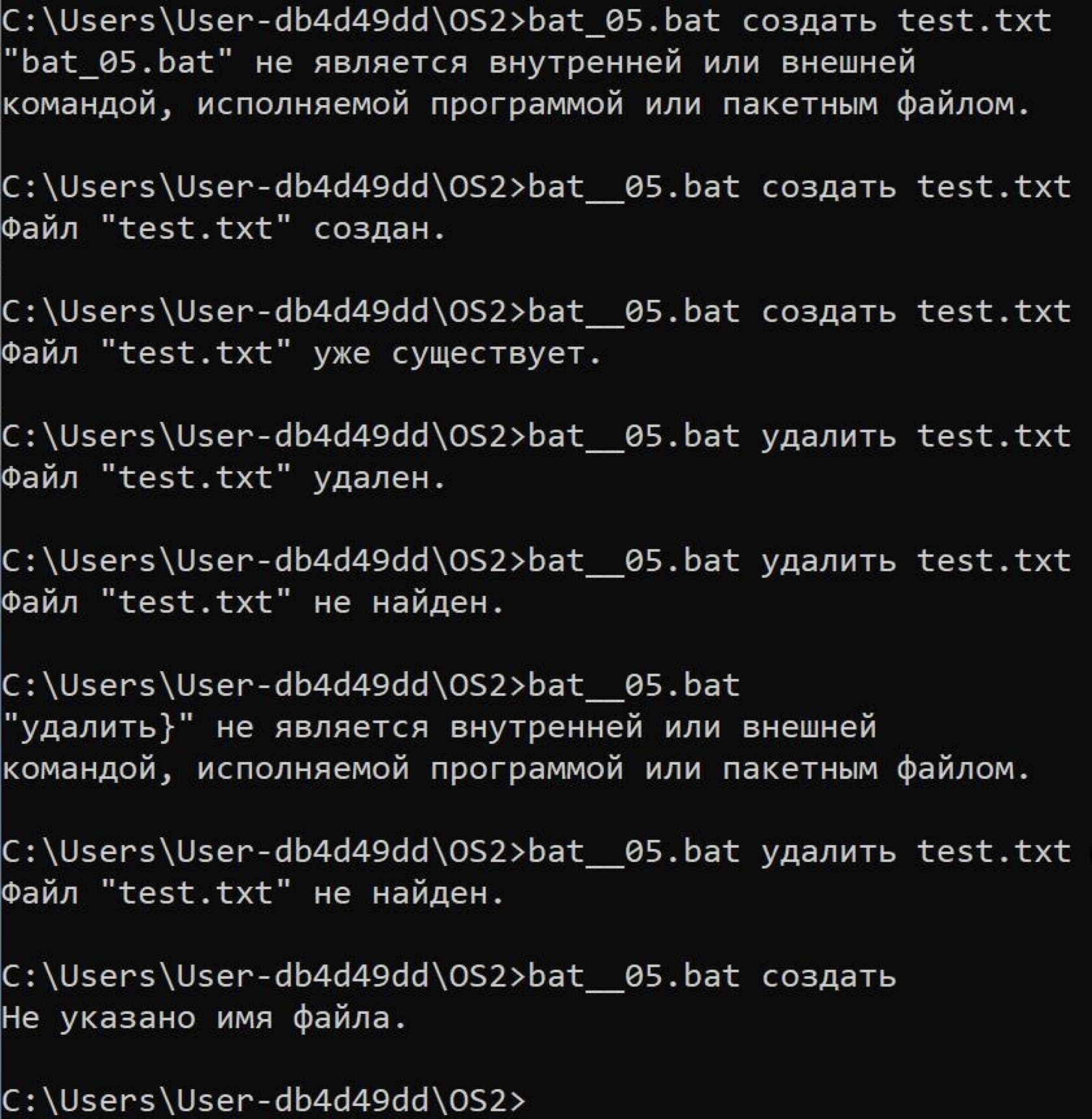


Рисунок 2.1.7.6 – Пример работы bat\_05 в разных режимах: создание и удаление файлов

## 2.1.8 Исследование правил наименования команд и параметров

На основе рассмотренных ранее консольных команд и с использованием документации по ним выделить правила их наименования, т.е. показать на примерах почему та или иная команда была так названа. И тоже самое провернуть относительно параметров данных команд.

Примеры:

* **copy** — от англ. copy, т.е. «копировать»;
* **del** — сокращение от delete, «удалить»;
* **dir** — сокращение от directory, «каталог»;
* **fc** — file compare, «сравнение файлов».

Параметры:

* **/P** — prompt, «запрос подтверждения»;
* **/S** — subdirectories, работа с подкаталогами;
* **/A** — attributes, фильтрация по атрибутам.

## 2.1.9 Переменные окружения: исследование

Переменные окружения — это глобальные значения, доступные всем процессам в системе.

Примеры важных переменных в Windows:

* %PATH% — список каталогов, в которых ищутся файлы.
* %USERNAME% — имя текущего пользователя.
* %COMPUTERNAME% — имя компьютера.
* %TEMP% — путь к временным файлам.

Самая важная переменная для работы в консоли: %PATH%, так как именно она определяет, какие команды можно вызвать напрямую.

## 2.2 Выполнение заданий для Linux

## 2.2.1 Пять bash-скриптов

Написать пять bash-скриптов аналогичных пяти bat-файлам из заданий по Windows, но с использованием особенностей синтаксиса и возможностей bash. (ss64)

Создаём скрипт, который выводит основную информацию о пользователе и системе: имя пользователя, текущую дату, время и имя компьютера. Скрипт создаётся прямо из консоли при помощи команды cat.

|  |
| --- |
| cat > bash\_01.sh << 'EOF'  #!/bin/bash  echo "User: $USER"  echo "Date: $(date '+%Y-%m-%d')"  echo "Time: $(date '+%H:%M:%S')"  echo "Host: $(hostname)"  EOF  chmod +x bash\_01.sh |

Листинг 2.2.1.1 – Создание скрипта bash\_01.sh для вывода информации о пользователе

Теперь выполним его в консоли, сделаем исполняемым и запустим для проверки корректной работы.

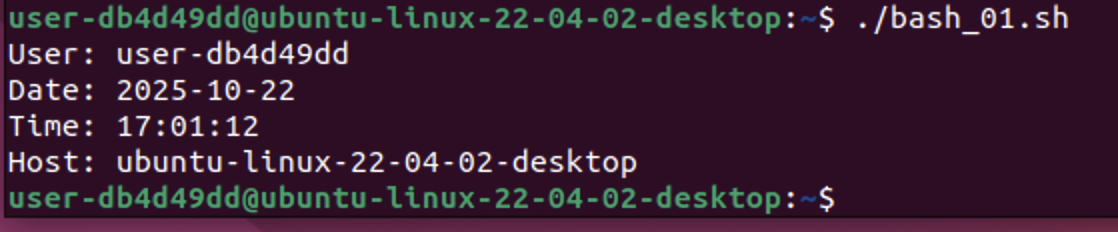


Рисунок 2.2.1.1 – Создание и запуск скрипта bash\_01.sh в терминале

Во втором скрипте выводятся сведения о самом файле: его имя, дата последней модификации и полный путь. Используются команды basename, stat и readlink.

|  |
| --- |
| cat > bash\_02.sh << 'EOF'  #!/bin/bash  echo "Script name: $(basename "$0")"  echo "Modified: $(stat -c %y "$0")"  echo "Full path: $(readlink -f "$0")"  EOF  chmod +x bash\_01.sh |

Листинг 2.2.1.2 – Создание скрипта bash\_02.sh для отображения сведений о файле

Теперь выполним его в консоли и убедимся, что выводятся корректные данные о самом скрипте.

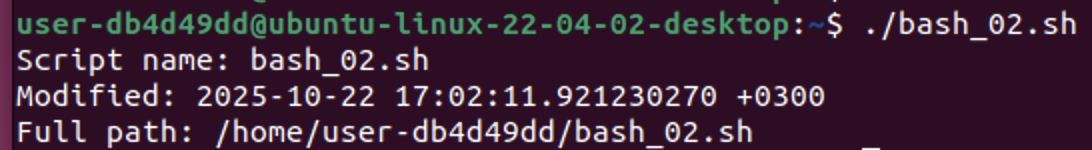


Рисунок 2.2.1.2 – Запуск скрипта bash\_02.sh и вывод информации о самом файле

Третий скрипт предназначен для демонстрации работы с параметрами командной строки. Он отображает все переданные аргументы и отдельно выводит первые четыре.

|  |
| --- |
| cat > bash\_03.sh << 'EOF'  #!/bin/bash  echo "All params: $@"  for i in 1 2 3 4; do  eval p=\$$i  echo "Param$i: ${p:-<empty>}"  done  EOF  chmod +x bash\_03.sh |

Листинг 2.2.1.3 – Создание скрипта bash\_03.sh для работы с параметрами

Теперь выполним его в консоли с набором тестовых аргументов и проверим, как они обрабатываются.

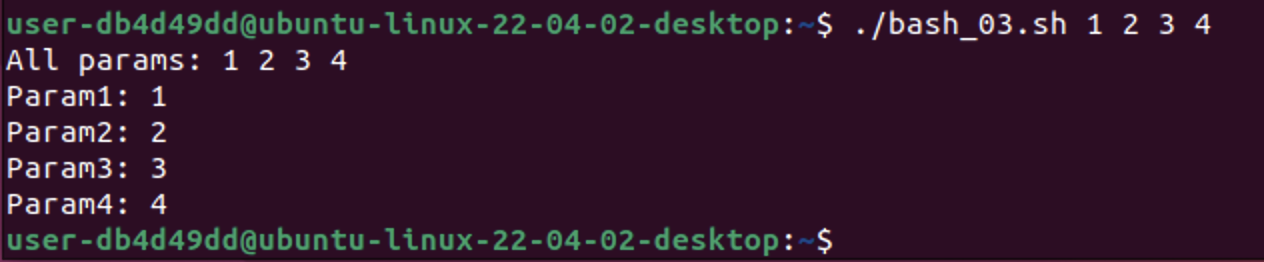


Рисунок 2.2.1.3 – Запуск bash\_03.sh с передачей параметров и результатом выполнения

Четвёртый скрипт выполняет арифметические операции над тремя параметрами: a, b и c. Используются встроенные возможности bash и утилита bc для точных дробных вычислений.

|  |
| --- |
| cat > bash\_04.sh << 'EOF'  #!/bin/bash  if [ $# -lt 3 ]; then  echo "Usage: $0 a b c"  exit 1  fi  a=$1; b=$2; c=$3  echo "a-b = $((a - b))"  echo "a+b = $((a + b))"  echo "c/b = $(echo "scale=6; $c/$b" | bc -l)"  echo "c\*b = $((c \* b))"  echo "(a-b)\*(b-a) = $(( (a - b) \* (b - a) ))"  EOF  chmod +x bash\_04.sh |

Листинг 2.2.1.2 – Создание скрипта bash\_04.sh для выполнения арифметических операций

Теперь выполним скрипт в консоли, передав три числовых параметра, и посмотрим результаты вычислений.

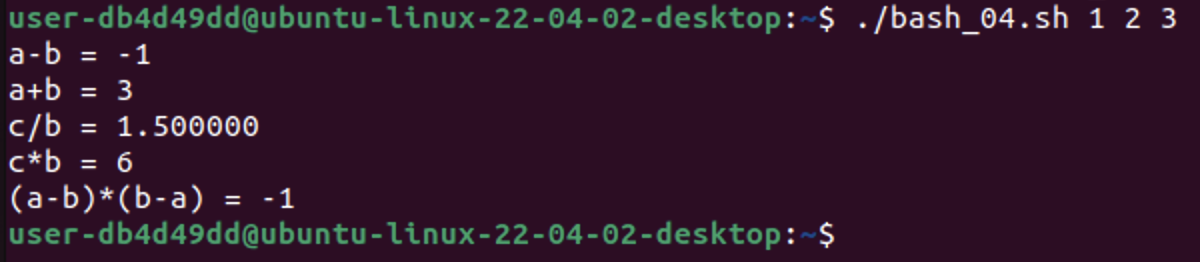


Рисунок 2.2.1.4 – Результаты выполнения арифметических операций в bash\_04.sh

Пятый скрипт реализует простейший файловый менеджер, поддерживающий два режима — «создать» и «удалить». В зависимости от введённых параметров создаётся или удаляется указанный файл.

|  |
| --- |
| cat > bash\_05.sh << 'EOF'  #!/bin/bash  mode=$1  file=$2  if [ -z "$mode" ]; then  echo "Usage: $0 {создать|удалить} {имя\_файла}"  exit 0  fi  if [ -z "$file" ]; then  echo "Не указано имя файла!"  exit 0  fi  case "$mode" in  создать)  if [ -e "$file" ]; then  echo "Файл $file уже существует!"  else > "$file"  echo "Файл $file создан."  fi;;  удалить)  if [ -e "$file" ]; then  rm "$file"  echo "Файл $file удалён."  else  echo "Файл $file отсутствует!"  fi;;  \*)  echo "Неверный режим: используйте 'создать' или 'удалить'";;  esac  EOF  chmod +x bash\_05.sh  ./bash\_05.sh создать test.txt  ./bash\_05.sh удалить test.txt |

Листинг 2.2.1.2 – Создание скрипта bash\_05.sh для управления файлами

Теперь выполним его в консоли в обоих режимах: сначала создадим новый файл, затем удалим его.

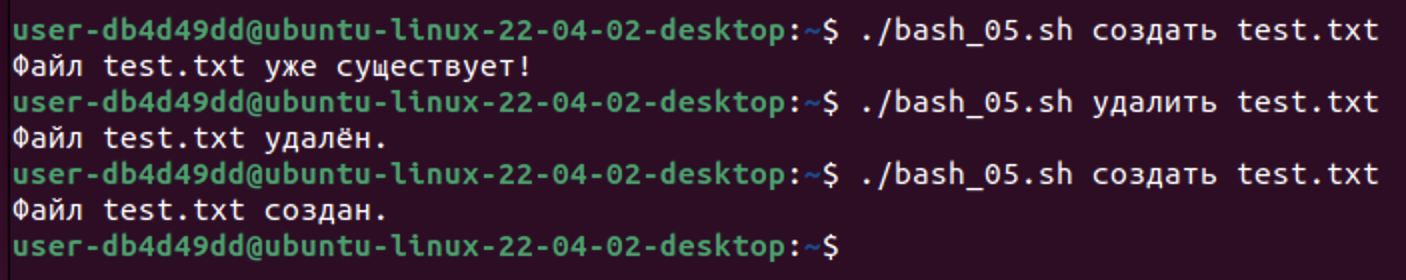


Рисунок 2.2.1.5 – Интерактивное управление файлами: создание и удаление по режиму

## 2.2.2 Консольные команды в Linux

Рассмотрим отличия в подходах к наименованию команд и параметров.

Таблица 2.2.2 – Сравнение основных комманд на Linux и Windows

|  |  |  |
| --- | --- | --- |
| Linux | Windows | Отличия |
| cd | cd | Аналогичны |
| copy source dest | cp source dest | В Linux часто применяют ключи -i (спросить при перезаписи), -r (рекурсивно), -p (сохранить атрибуты). В Windows всё через отдельные ключи /Y, /-Y. |
| xcopy source dest /E /H /K | cp -a source dest или rsync -a source dest | В Windows xcopy имеет много ключей (/E, /H, /K), а в Linux всё это объединено в -a (archive mode). |
| move source dest | mv source dest | Отличий почти нет; в Linux ключи -i (подтверждение перезаписи), -n (не перезаписывать). |
| del filename | rm filename | В Linux удаление безвозвратное (нет корзины), добавляются ключи -i (подтверждение), -r (рекурсивно). |
| rd dirname | rmdir dirname или rm -r dirname | В Windows rd удаляет только пустые каталоги без /S, в Linux rmdir тоже; для непустых каталогов нужен rm -r |
| ren old new | mv old new | В Linux mv объединяет функции перемещения и переименования; отдельной команды ren нет. |
| fc file1 file2 | diff file1 file2 | В Linux вывод построчный, ключи для форматов (-u, -c); в Windows fc даёт побайтовое или построчное сравнение. |
| dir | ls | В Linux для подробного вывода используют ls -l, рекурсивно — ls -R; в Windows есть ключи /S, /P, /O. |
| type file.txt | cat file.txt | В Linux можно использовать cat, less, more; в Windows type ограничен простым выводом. |

## 2.2.3 Переменные окружения в Linux

Кратко перечислим основные отличия Linux от Windows:

* имена переменных в Linux **чувствительны к регистру** (PATH ≠ path), в Windows — обычно регистр не различают при обращении через %VAR%;
* в Linux переменная подставляется как $VAR, в Windows — %VAR%;
* в Linux переменные **экспортируются** в потомков (export), в Windows нужно **setx** для перманентной записи (и set для сессии).

3 Заключение

В ходе выполнения лабораторной работы я ознакомилась с основными командами командных оболочек Windows (cmd) и Linux (bash), научилась работать с файлами и каталогами, применять перенаправление потоков, конвейеры и атрибуты команд, создавать и запускать пакетные и bash-скрипты для автоматизации типовых задач. Дополнительно были исследованы правила наименования команд и их параметров, а также роль переменных окружения, что позволило лучше понять отличия подходов в разных операционных системах и приобрести практические навыки администрирования и работы с консолью.