Устимова Снежана, ПИ19-3

Лабораторная работа №2

Файловый менеджер

Для начала напишу небольшое руководство пользователя по своему менеджеру. Итак:

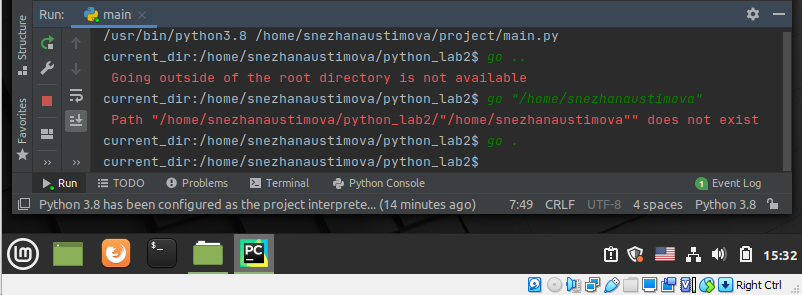
1. Создание папки с указанием имени – crfdir.
2. Удаление папки по имени – deldir.
3. Перемещение между папками – go.
4. Создание пустого файла с указанием имени – crfile.
5. Запись текста в файл – write.
6. Просмотр содержимого каталога – view.
7. Удаление файла по имени – delfile.
8. Копирование файла из одной директории в другую – cpfile.
9. Перемещение файла – mvfile.
10. Переименование – rename.

Как мы видим, ни одна команда не повторяет уже существующие.

В любом случае, все равно лучше рассмотреть все на примерах. Так как файловый менеджер должен быть кроссплатформенным, начнем его тестирование с **OS Linux**.

В указаниях к работе было сказано, чтобы пользователь мог сам настраивать рабочую директорию, которая не должна быть связана с расположением проекта. Настройки должны находиться в отдельном файле. Что самое главное, файловый менеджер должен блокировать пользователя от выхода за пределы рабочей папки. Пользователь должен воспринимать рабочую папку как корневую и все действия файлового менеджера должны локализоваться только в пределах этой папки.

В качестве корневой директории я установила папку /home/snezhanaustimova/python\_lab2, в то время как сам проект находился в директории /home/Snezhanaustimova/project.

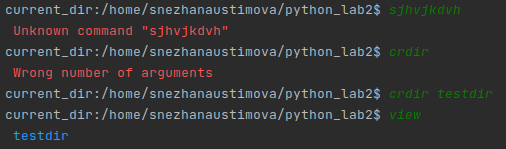


Как вы видим, пользователь не может выйти за пределы указанной папки. Условие выполнено.

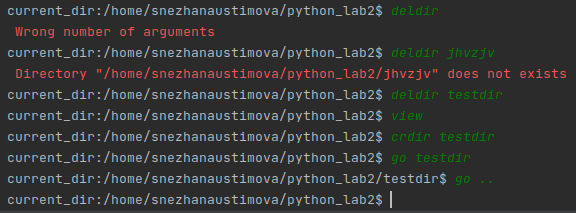
Забыла сказать, что прерывание программы (закрытие менеджера) производится командой exit.

Перейдем к командам.

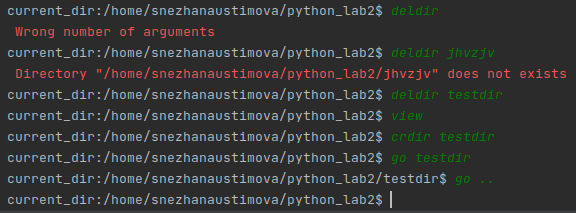
1. Создание папки. Должны передаваться обязательные параметры, без которых программа выдаст ошибку “Wrong number of arguments”. Помимо этого, я показала, как при написании какой-либо «белиберды» менеджер скажет нам, что такой команды не существует. Напомню, что view – это просмотр содержимого каталога.



2. Удаление директории. Если переданной папки нет, то выведется ошибка “directory does not exists”. Если папка не пустая, программа запросит подтверждение пользователя.



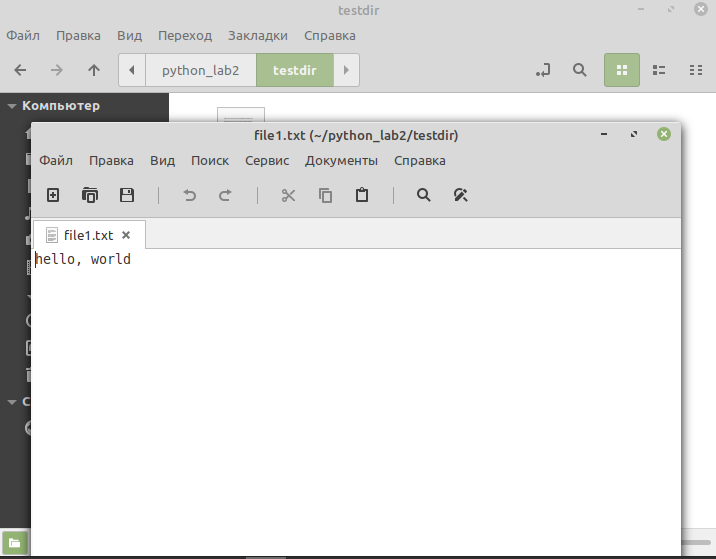
3. Переход из одной директории в другую:



4-5. Создание пустого файла и запись текста в файл.

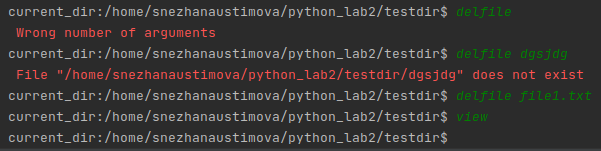


Посмотрим, что у нас вышло:

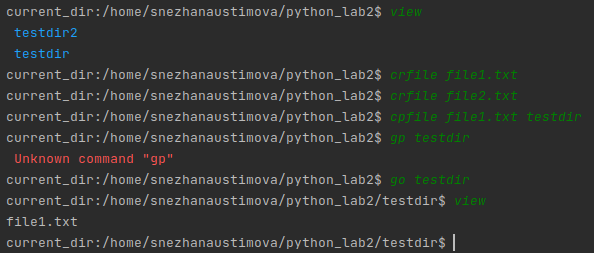


Все работает!

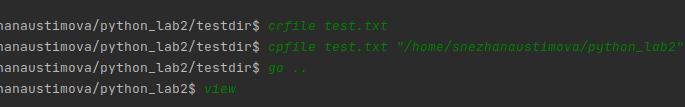
6-7. Просмотр содержимого директории и удаление файла.



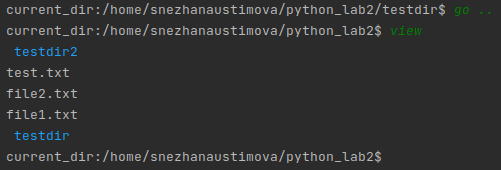
8. Копирование файла. Создадим 2 файла file1.txt и file2.txt. Скопируем первый в директорию testdir, которая, как мы помним, до этого была пустой.

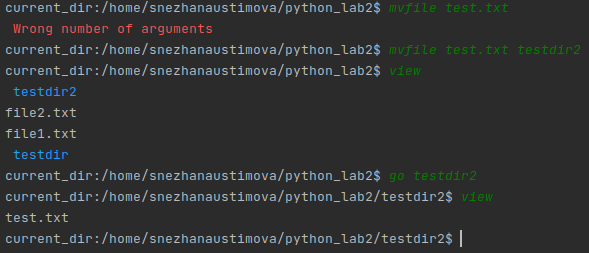


Так же убедимся, что мы можем скопировать файл в директорию, переданную в кавычках. Я забыла заскринить, но если мы введем путь, содержащий пробелы без кавычек, программа выведет ошибку о неверном формате ввода директории (в коде это можно посмотреть).

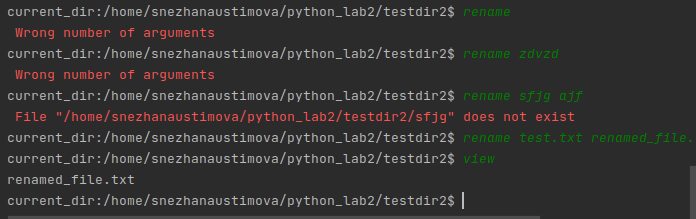


9. Перемещение файлов. Все аналогично с cpfile.



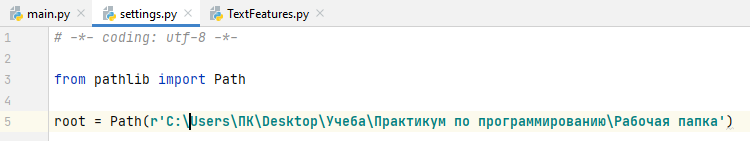


10. Переименование файла.

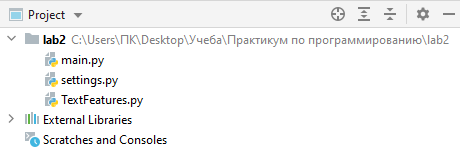


**OS Windows**

Корневая папка:



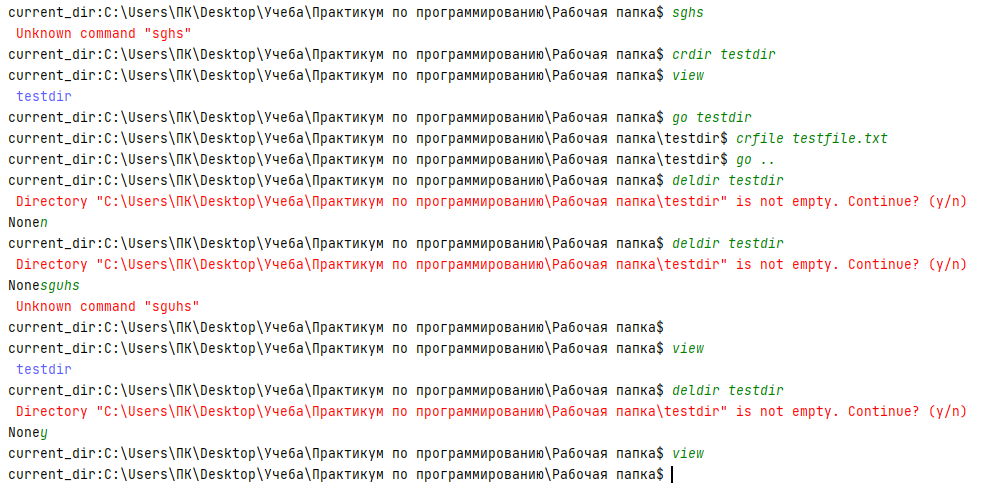
Расположение самого проекта:



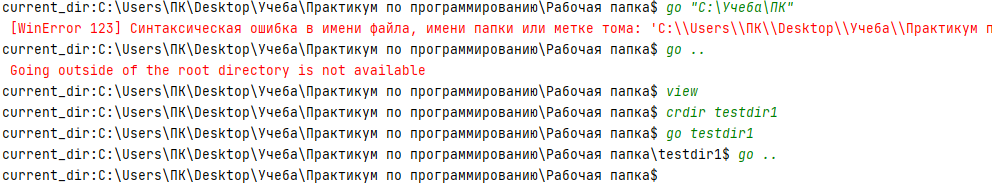
Они не находятся отдельно друг от друга. Условие соблюдено.

Начнем.

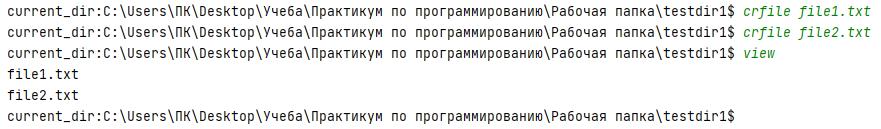
1-2. Создание папки с указанием имени и удаление папки по имени. Заодно демонстрирую то, что забыла показать на линуксе – запрос подтверждения на удаление, если папка не пустая.



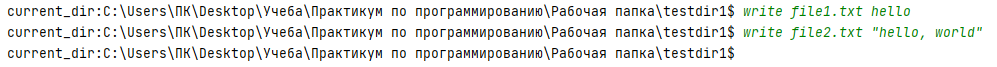
3. Перемещение между папками. Его видно на предыдущем скриншоте, но покажу, что мы не можем выходить за пределы корневой директории.

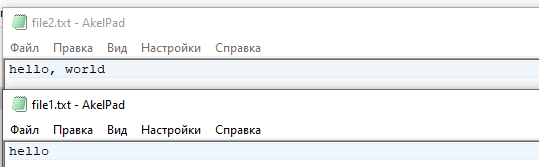


4. Создание пустого файла с указанием имени – crfile.



5. Запись текста в файл – write.





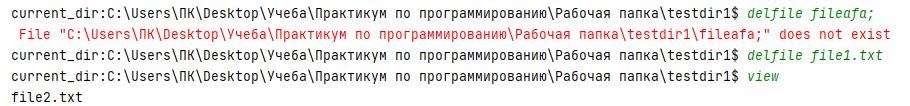
6. Просмотр содержимого каталога – view.



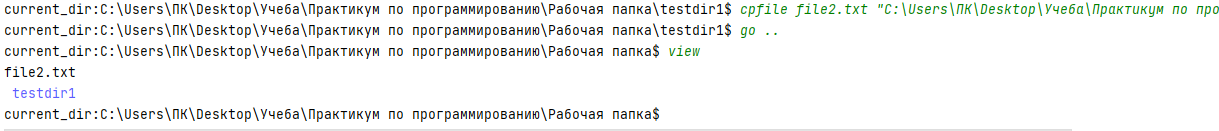
Команда view не имеет аргументов:



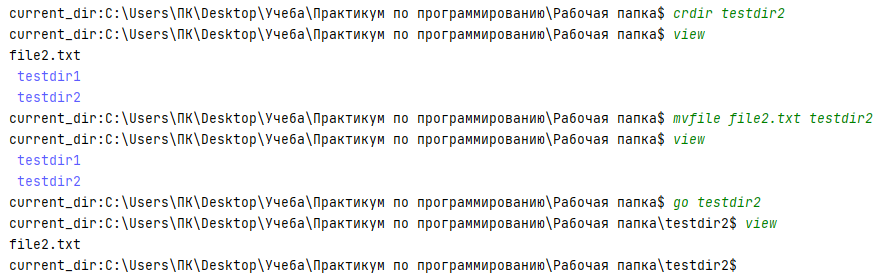
7. Удаление файла по имени – delfile.



8. Копирование файла из одной директории в другую – cpfile.



9. Перемещение файла – mvfile.



10. Переименование – rename.

