Министерство образования и науки Российской Федерации

Новосибирский государственный технический университет

Кафедра ТПИ

Расчетно-графическая работа

## **Разработка эмулятора командного процессора ОС Windows**

Факультет: ФПМИ

Группа: ПМ-91

Студенты: Затолоцкая Ю. А.

Преподаватель: Кобылянский В.Г.

Новосибирск

2021

# Содержание

[**Разработка эмулятора командного процессора ОС Windows** 1](#_Toc83221992)

[Содержание 2](#_Toc83221993)

[Введение 3](#_Toc83221994)

[Теоретическая часть 4](#_Toc83221995)

[Описание программы 5](#_Toc83221996)

[Результаты тестирования 6](#_Toc83221997)

[Заключение 14](#_Toc83221998)

[Источники 14](#_Toc83221999)

[Приложение 15](#_Toc83222000)

# Введение

Командный процессор - программа, обеспечивающая обработку и выполнение команд ОС. Эмуляторкомандного процессора должен создать у пользователя иллюзию работы в среде заданной ОС, т.е. пользователь должен, например, вводить команды Windows, а выполняться они должны в системе Linux. Реализация команд заданной ОС возможна с помощью команд Linux или с помощью процедур, написанных на алгоритмических языках.

Эмулятор должен обеспечивать интерфейс заданной ОС. Должны быть предусмотрены возможность обработки ключей команды и возможность исполнения командных файлов (сценариев). Количество команд – не менее 10, количество ключей в командах – не менее двух. Если ОС предусматривает возможность регистрации пользователя и ввода пароля, то эмулятор также должен реализовать эту функцию.

Эмулятор командного процессора с русскоязычным интерфейсом должен обеспечить ввод команд на русском языке.

Интерфейс эмулятора должен соответствовать стандартному интерфейсу командной строки с использованием текстового режима.

Для выполнения работы нами были выбраны следующие команды и ключи:

1. help [<command>] - Отображает список доступных команд или подробные справочные сведения об указанной команде. Если используется без параметров, выводятся списки и краткое описание каждой системной команды.

|  |  |
| --- | --- |
| <command> | Указывает команду, для которой отображаются подробные справочные сведения. |

1. mkdir <path> - Создает каталог или подкаталог.

|  |  |
| --- | --- |
| <path> | Указывает имя и расположение нового каталога. Максимальная длина любого отдельного пути определяется файловой системой. Это обязательный параметр. |

1. rmdir <path> - Удаляет каталог.

|  |  |
| --- | --- |
| <path> | Указывает расположение и имя каталога, который требуется удалить. |

1. del [/p] <name> - Удаляет один файл.

|  |  |
| --- | --- |
| <name> | Указывает имя файла. |
| /p | Запрашивает подтверждение перед удалением указанного файла. |

1. cd [<path>] - Изменяет текущий каталог.

cd [..]

|  |  |
| --- | --- |
| <path> | Указывает путь к каталогу, который требуется отобразить или изменить. |
| [..] | Указывает, что вы хотите перейти к родительской папке. |

1. dir /b - Отображает список файлов и подкаталогов каталога.

|  |  |
| --- | --- |
| /b | Отображает список несовпадающих каталогов и файлов без дополнительных сведений. |

1. exit - Выход из интерпретатора команд.
2. rename [<path>]<filename1> <filename2> - Переименовывает файлы или каталоги.

|  |  |
| --- | --- |
| [<path>]<filename1> | Указывает расположение и имя файла или набора файлов, которые требуется переименовать.. |
| <filename2> | Указывает новое имя для файла. |

1. ipconfig [/all] - Вывод деталей текущего соединения

|  |  |
| --- | --- |
| all | Отображение полной информации по всем адаптерам. |

1. type [<path>]<filename> - Отображает содержимое текстового файла

|  |  |
| --- | --- |
| [<path>]<filename> | Указывает расположение и имя файла, который требуется просмотреть. |

1. cmd.exe < <filename> - Запуск сценария. Командный процессор не будет ожидать ввода команд с клавиатуры, а считает их из файла <filename>.

В качестве базовой ОС нами был выбран ОС дистрибутив Linux Fedora 32 (Workstation Edition).

# Теоретическая часть

Основным интерфейсом для любой операционной системы (ОС) является командный интерфейс, представленный набором команд. Благодаря этому и осуществляется диалог между пользователем и ОС. Количество команд ОС может быть достаточно большим, каждая команда реализует одно действие над заданным ресурсом, а основным устройством управления при этом является клавиатура.

Команда набирается латиницей завершается нажатием клавиши <ENTER>. Формат командной строки следующий:

имя\_команды [ключи] [аргументы]

Здесьимя\_командыуказывает действие; аргументы *-* именаобъектов, над которыми выполняется действие, а ключи (опции) – уточняют действие команды. Ключи и аргументы не являются обязательными для всех команд, т.е. возможны команды, не имеющие параметров (например, cd). Все элементы командной строки разделяются пробелами, ключи начинаются с символа ‘/‘.

Команды бывают двух типов: внутренние и внешние. Внутренние команды выполняются непосредственно командным интерпретатором, а внешние команды реализуются программами, поставляемыми вместе с ОС в виде отдельных файлов. Командный интерпретатор распознает элементы командной строки, выделяя последовательности символов между пробелами, и пытается найти функцию, имя которой совпадает с именем команды, в своей внутренней базе. Если такая функция там зарегистрирована, то она исполняется с аргументами и ключами, выделенными из командной строки, иначе на экране появится сообщение «<command>: команда не найдена».

# Описание программы

Программа имеет следующие переменные и методы:

- переменная str – строка в формате "имя\_команды [ключи] [аргументы]"

- переменная option – символ ' ', разделяющий имя\_команды, ключи, аргументы

- переменная argument – массив аргументов

- метод parseCommand(command, option, argument) – разбивает на составные части строку

- метод runCommand(command, option, argument) – исполняет команду

Программа зациклена, чтобы была возможность продолжать работу после выполнения одной команды. Выход из цикла и завершение работы программы осуществляется вводом команды «exit».

В зависимости от того, какая команда была введена, будут выполняться следующие действия:

- при вводе команды «help» отображаются подробные справочные сведения об указанной в аргументе команде. Если используется без параметров, выводятся списки и краткое описание каждой системной команды. Если команды нет в списке, выводится строка "Данная команда не поддерживается. Воспользуйтесь параметром “<command> /?”."

- при вводе команды «mkdir» создается каталог или подкаталог. При вводе без аргумента или ввода аргумента в неверном формате выводится сообщение об ошибке.

- при вводе команды «type» осуществляется поиск в текущей директории файла с именем argument и отображается содержимое текстового файла.

- при вводе команды «rmdir» осуществляется поиск в текущей директории каталога с именем argument, затем каталог удаляется. Если директория не пуста, выводится строка "Папка не пуста." Если такой папки не существует, выводится строка "Не удается найти указанный файл."

- при вводе команды «del» осуществляет удаление файла с именем argument в текущей директории. Если введен ключ «/p», то запрашивается подтверждение перед удалением указанного файла. Если файл не найден, выводится строка "Не удается найти указанный файл."

- при вводе команды «cd» осуществляется поиск в текущей директории каталога с именем argument, затем происходит переход в этот каталог. Если путь не найден, выводится строка "Системе не удается найти указанный путь." Если аргумент пуст - отображает имена текущего диска и каталога. Аргумент «..» обозначает переход в родительский каталог.

- при вводе команды «dir /b» отображается список файлов и подкаталогов каталога.

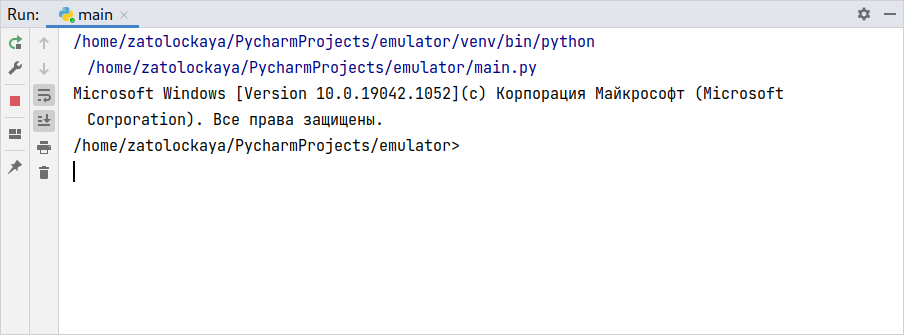
- при вводе команды «rename» осуществляется поиск в текущей директории каталога с именем argument[0], затем этот файл переименовывается на argument[1]. Если файл не найден, выводится строка "Не удается найти указанный файл."

- при вводе команды «ipconfig» выводятся детали текущего соединения. Если введен ключ «/all», то отображается полная информация по всем адаптерам.

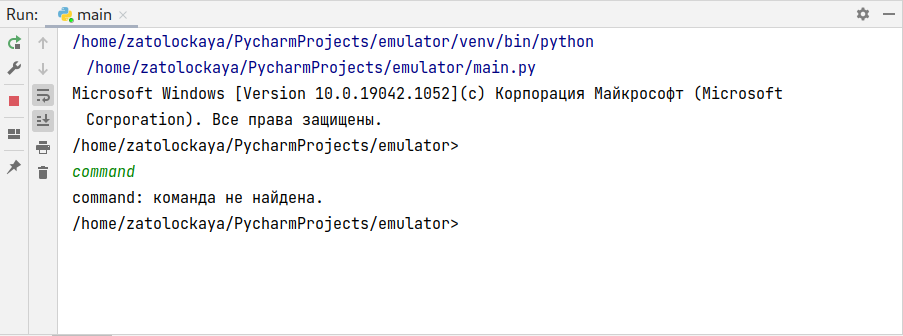
- при вводе команды «cmd.exe <» открывается файл сценария argument[1] для чтения. Затем в цикле считываются строки с командами, вызываются функции parseCommand(str, option, argument), runCommand(command, option, argument). Цикл завершается, когда достигнут конец файла. Если файл не найден, выводится строка "Не удается найти указанный файл."

# Результаты тестирования

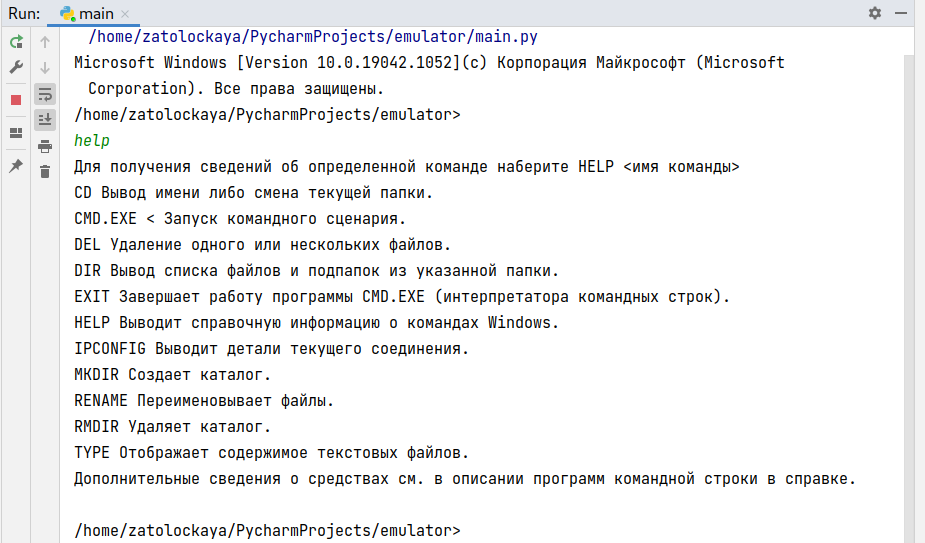
Стартовое окно:



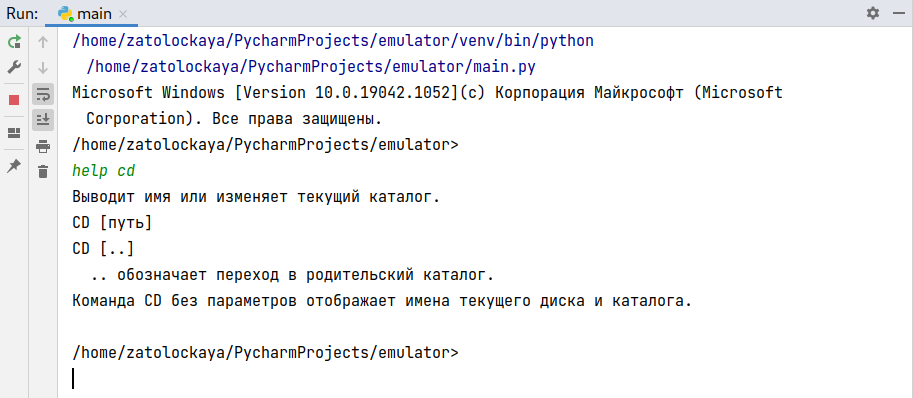
В случае ввода несуществующей команды:



С помощью команды «**help**» выведем все доступные команды:

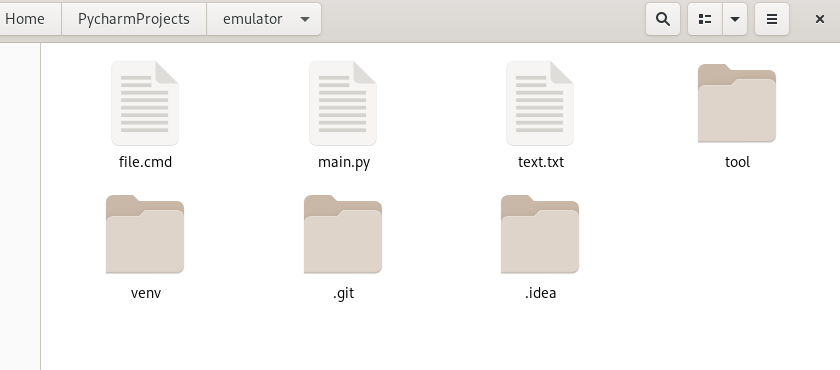
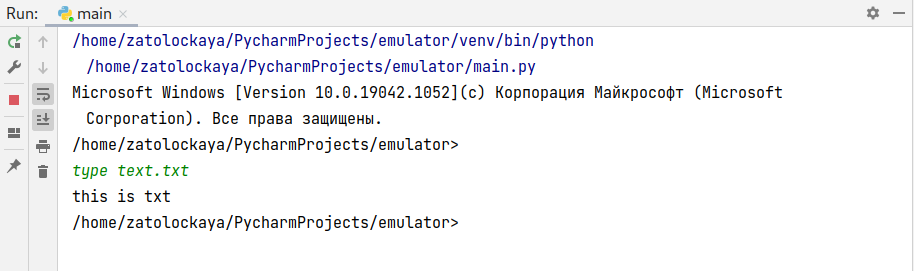


Попробуем вывести «**help**» по одной из команд отдельно:



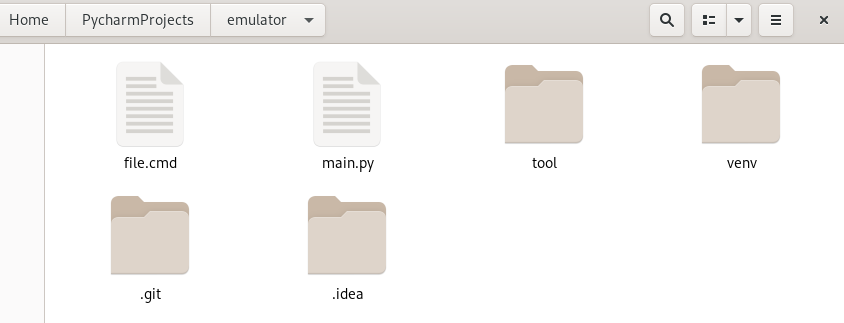
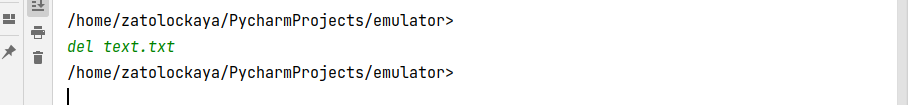
**Работа с текстовыми файлами.**

Попробуем с помощью команды «**type**» вывести содержимое существующего файла на экран:



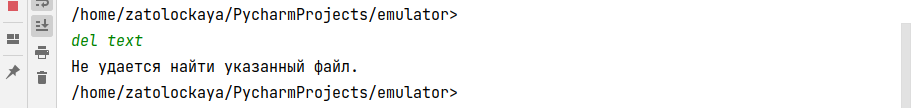
Мы видим, что текст файла выведен верно.

Удалим файл text.txt с помощью команды «**del**»:



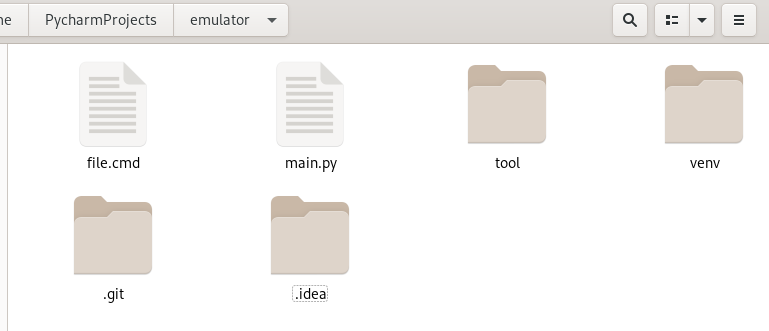
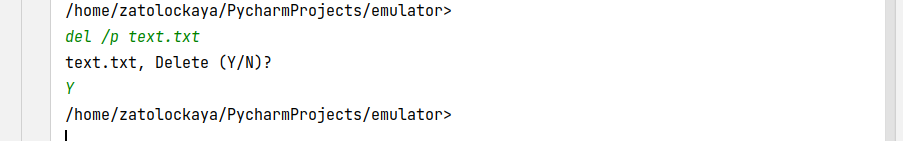
Файл успешно удален.

Попробуем удалить файл, которого нет в текущем каталоге:



Такое действие невозможно.

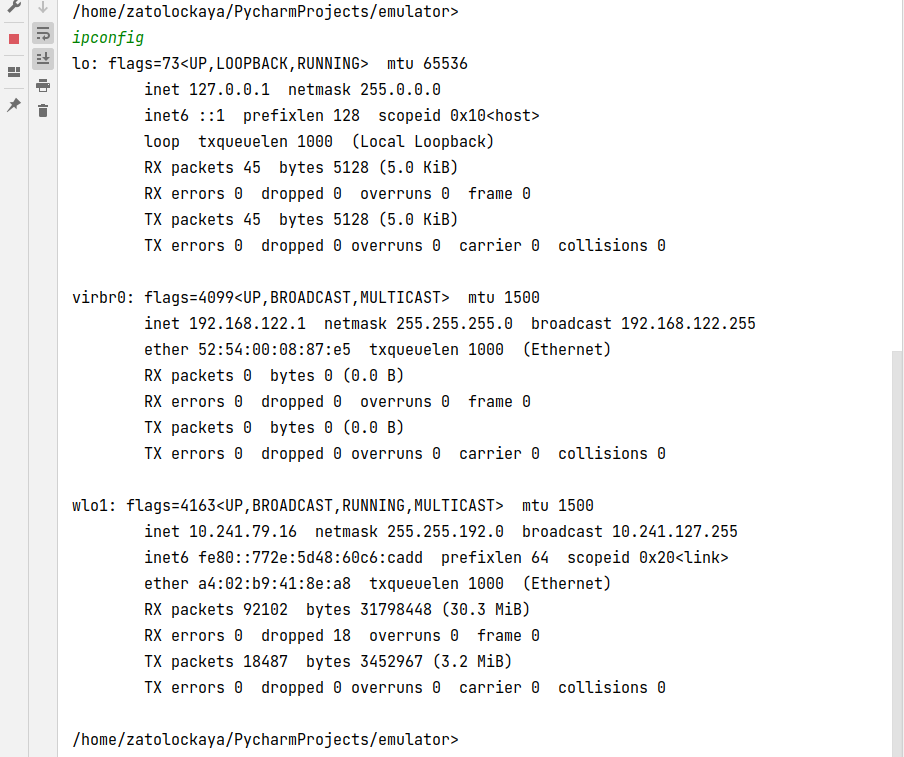
Попробуем удалить файл с параметром «**/p**»

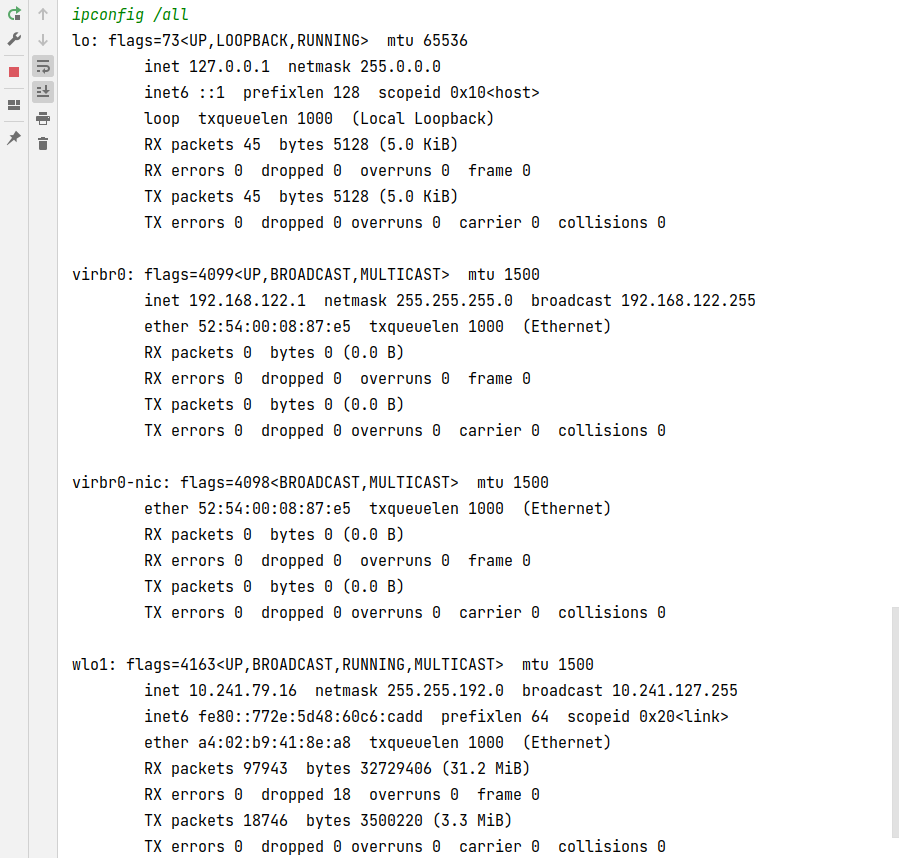


Файл успешно удален.

Запустим команду «**ipconfig**»

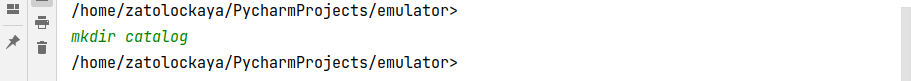
Данные выведены верно.

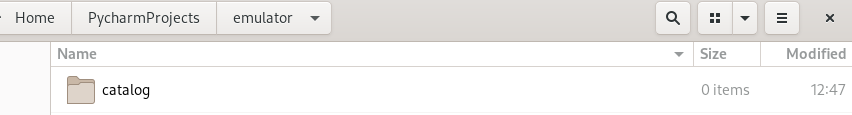
Запустим команду «**ipconfig**» с параметром «**/all**»

Отобразилась полная информация по всем адаптерам.

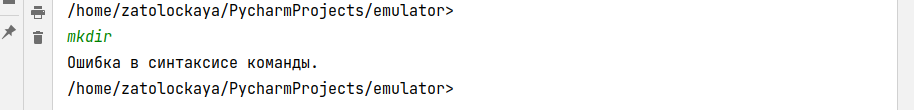
**Работа с каталогами.**

Создадим каталог «каталог» при помощи команды «**mkdir**»:

Каталог успешно создан:

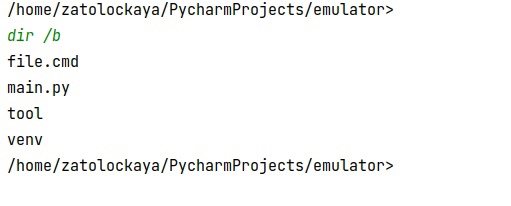


Проверим что будет, если ввести команду «**mkdir**» без аргумента:



Такое действие невозможно, так как нельзя создать безымянный каталог.

С помощью команды «**dir /b**» выведем содержимое каталога «emulator»:



Команда верно выводит список файлов в текущем каталоге.

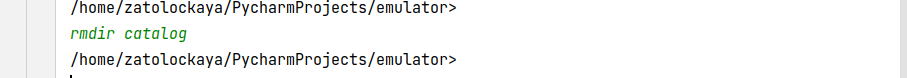
При помощи команды «**cd**» можно перейти в любую доступную папку в текущем каталоге:

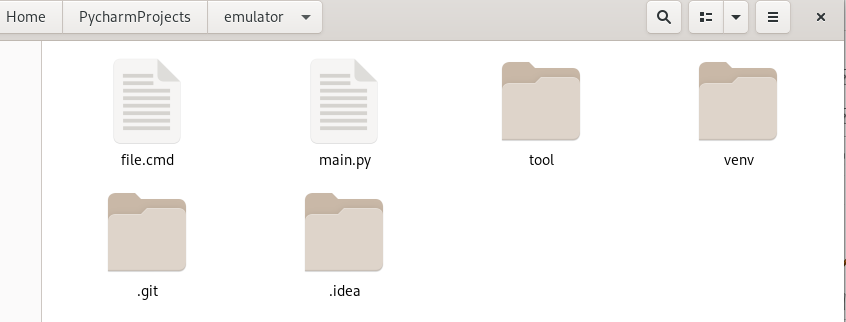
Команда «**cd**» без параметров выводит имя текущего каталога.



Если нужного каталога нет, то программа выдаст предупреждение о том, что такое действие невозможно.

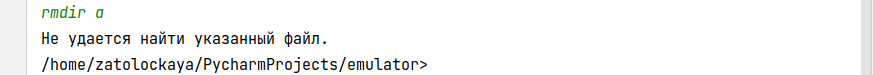
Команда «**rmdir**» удаляет указанный каталог:

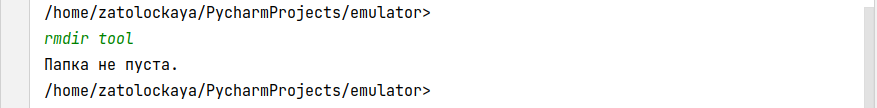




Каталог был успешно удален.

При попытке удалить несуществующий каталог выводится ошибка.

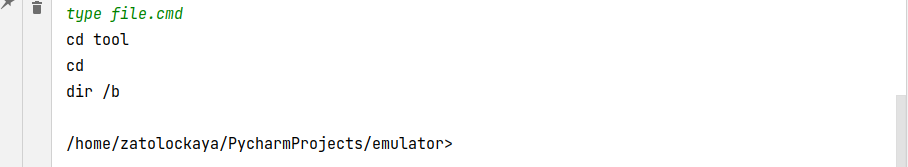
При попытке удалить непустую папку выводится ошибка.



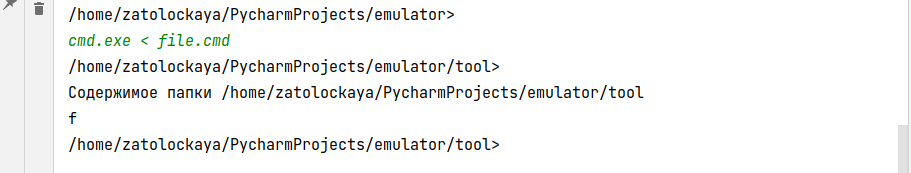
**Работа со сценариями.**

Запустим файл file.cmd

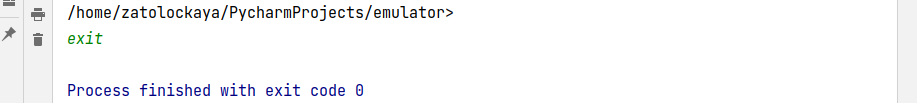
Его содержимое:



Запустим файл с помощью команды «**cmd.exe** **<**»

Файл был запущен успешно.

Для завершения работы программы введем команду «**exit**»:

Программа успешно завершила выполнение.

# Заключение

Во время выполнения работы мы разобрались в принципах работы командного процессора. Разработка эмулятора командного процессора ОС Windows, работающего на Linux, помогло нам разобраться в отличиях этих операционных систем, например, отличия в файловых системах.

Создавая программу, мы опирались на знания, полученные во время изучения курса «Операционные системы, среды и оболочки», и опыт работы с командными интерпретаторами Linux и Windows. Мы старались приблизить нашу программу по функционалу к командному интерфейсу Windows. На наш взгляд, нам удалось выполнить поставленную задачу.

Для реализации был выбран язык Python.

В итоге мы создали программу – эмулятор командного процессора Windows, работающий на Linux. Реализовали 11 команд, для 3-х из них были реализованы ключи.

# Источники

* Операционные системы, среды и оболочки : [учебное пособие] / В. Г. Кобылянский ; Новосиб. гос. техн. ун-т Новосибирск : Изд-во НГТУ , 2018
* Операционные системы, среды и оболочки : [методические указания к выполнению РГЗ] / В. Г. Кобылянский ; Новосиб. гос. техн. ун-т Новосибирск : Изд-во НГТУ , 2021
* <https://docs.microsoft.com/ru-ru/windows-server/administration/windows-commands/windows-commands> — Официальный справочник по командам Windows.

# Приложение

import os  
  
def runCommand(command, option, argument):  
 if command == **"mkdir"**:   
 if argument[0] == **' '**:  
 print(**"Ошибка в синтаксисе команды."**)  
 else:  
 try:  
 os.mkdir(argument[0])  
 except FileNotFoundError:  
 print(**"Ошибка в синтаксисе команды."**)  
 elif command == **"ipconfig"**:  
 if option == **"/all"**:  
 os.system(**"ifconfig -a"**)  
 else:  
 os.system(**"ifconfig"**)  
 elif command == **"exit"**:  
 return  
 elif command == **"cmd.exe"**:  
 if argument[0] == **'<'**:  
 try:  
 f = open(argument[1], **'r'**)  
 while True:  
 str = f.readline()  
 if not str:  
 break  
 option = **' '**  
argument = []  
 command, option, argument = parseCommand(str, option, argument)  
 runCommand(command, option, argument)  
 f.close()  
 except FileNotFoundError:  
 print(**"Не удается найти указанный файл."**)  
  
 elif command == **"type"**:  
 try:  
 os.system(**"cat "** + argument[0])  
 except FileNotFoundError:  
 print(**"Не удается найти указанный файл."**)  
 elif command == **"rmdir"**:   
 try:  
 os.rmdir(argument[0])  
 except FileNotFoundError:  
 print(**"Не удается найти указанный файл."**)  
 except OSError:  
 print(**"Папка не пуста."**)  
 elif command == **"del"**:  
 try:  
 if option == **"/p"**:  
 print(argument[0] + **", Delete (Y/N)?"**)  
 answer = input()  
 if answer == **'Y'**:  
 os.remove(argument[0])  
 else:  
 return  
 else:  
 os.remove(argument[0])  
 except FileNotFoundError:  
 print(**"Не удается найти указанный файл."**)  
 except IsADirectoryError:  
 print(**"Файл является директорией."**)  
  
 elif command == **"cd"**:   
 if argument[0] == **' '**:  
 print(os.getcwd() + **">"**)  
 else:  
 try:  
 os.chdir(argument[0])  
 except FileNotFoundError:  
 print(**"Системе не удается найти указанный путь."**)  
  
 elif command == **"dir"**:   
 if option == **'/b'**:  
   
 os.system(**'ls'**)  
  
 elif command == **"rename"**:  
 try:  
 os.rename(argument[0], argument[1])  
 except FileNotFoundError:  
 print(**"Не удается найти указанный файл."**)  
 elif command == **"help"**:  
 if argument[0] == **"mkdir"**:  
 print(**"""Создает каталог или подкаталог.**   
**mkdir <path>**  
**<path> Указывает имя и расположение нового каталога. """**)  
 elif argument[0] == **"ipconfig"**:  
 print(**"""Вывод деталей текущего соединения**  
 **mkdir <path>**  
 **<path> Указывает имя и расположение нового каталога. """**)  
 elif argument[0] == **"rmdir"**:  
 print(**"""Удаление каталога.**  
  
**RMDIR <path>**  
**<path> Указывает расположение и имя каталога, который требуется удалить.**   
  
 **"""**)  
 elif argument[0] == **"del"**:  
 print(**"""Удаление одного файла.**  
**DEL [/P] names**  
 **name Имя файла.**  
 **/P Запрос подтверждения перед удалением каждого файла.**  
 **"""**)  
 elif argument[0] == **"exit"**:  
 print(**"""Завершает программу CMD.EXE (интерпретатор команд) или текущий пакетный**  
**файл-сценарий.**  
**EXIT**   
 **"""**)  
 elif argument[0] == **"cd"**:  
 print(**"""Выводит имя или изменяет текущий каталог.**  
**CD [путь]**  
**CD [..]**  
 **.. обозначает переход в родительский каталог.**  
**Команда CD без параметров отображает имена текущего диска и каталога.**  
**"""**)  
 elif argument[0] == **"dir"**:  
 print(**"""Вывод списка файлов и подкаталогов в указанном каталоге.**  
**DIR [path][filename] /B**   
 **[path][filename]**  
 **Каталог или имена файлов для включения в список.**  
 **/B Вывод только имен файлов.**  
 **"""**)  
 elif argument[0] == **' '**:  
 print(**"""Для получения сведений об определенной команде наберите HELP <имя команды>**  
**CD Вывод имени либо смена текущей папки.**  
**CMD.EXE < Запуск командного сценария.**  
**DEL Удаление одного или нескольких файлов.**  
**DIR Вывод списка файлов и подпапок из указанной папки.**  
**EXIT Завершает работу программы CMD.EXE (интерпретатора командных строк).**  
**HELP Выводит справочную информацию о командах Windows.**  
**IPCONFIG Выводит детали текущего соединения.**  
**MKDIR Создает каталог.**  
**RENAME Переименовывает файлы.**  
**RMDIR Удаляет каталог.**  
**TYPE Отображает содержимое текстовых файлов.**  
**Дополнительные сведения о средствах см. в описании программ командной строки в справке.**  
 **"""**)  
 elif argument[0] == **"rename"**:  
 print(**"""Переименование одного или нескольких файлов.**  
  
**RENAME [путь]имя\_файла1 имя\_файла2.**  
  
  
  
 **"""**)   
 elif argument[0] == **"type"**:  
 print(**"""Вывод содержимого одного или нескольких текстовых файлов.**  
  
**TYPE [путь]имя\_файла**  
 **"""**)  
 else:  
 print(**""" Данная команда не поддерживается. Воспользуйтесь параметром** \"**"""** + argument[0] + **" /?**\"**."**)  
 else:  
 print(command + **": команда не найдена."**)  
  
def parseCommand(command, option, argument):  
 command = command.lower()  
 command = command.split()  
  
 while len(command) < 3:  
 command.append(**' '**)  
 for word in command[1:]:  
 if word[0] == **'/'**:  
 option = word  
 else:  
 argument.append(word)  
 if not argument:  
 argument = [**" "**, **" "**]  
 command = command[0]  
 return command, option, argument  
  
  
def main():  
 print(  
 **"Microsoft Windows [Version 10.0.19042.1052](c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены."**)  
  
 command = **' '**  
while command != **"exit"**:  
 print(os.getcwd() + **">"**)  
 str = input()  
 option = **' '**  
argument = []  
 command, option, argument = parseCommand(str, option, argument)  
 runCommand(command, option, argument)  
  
  
if \_\_name\_\_ == **"\_\_main\_\_"**:  
 main()