

# Лабораторная работа №1

---

Выполнил: Галкин Климентий. 4851003/10002

## Цели и задачи

**Цель** - получение навыков управления файловой системой и реестром с использованием языка программирования Python.

**Задача** - Реализовать программу для работы с файловой системой и реестром операционной системы Windows. Взаимодействие с командой идет через аргументы командной строки. Функционал с файловой системой: создание, удаление, чтение файла, запись в файл, изменение названия файла, копирование файла в другую директорию. Функционал работы с реестром: создание и удаление ключа, установка значения в ключе.

## Ход работы

### Описание программы

Используемые модули:

1. os
2. shutil
3. winreg
4. struct

В ходе выполнения работы было реализовано два класса: **FileSystem** и **WinReg**. Кратко опишем методы каждого класса:

- **FileSystem**
  - `__init__` - "конструктор" класса. Переопределяет передаваемые в класс параметры
  - `file_write` - функция записи введенной строки в файл. Важно отметить, что файл открывается с атрибутом "a", означающий дополнение файла. При открытии файла с атрибутом "w" все старые данные будут перетираться.
  - `file_delete` - функция удаления файла
  - `file_create` - создание файла. Использование метода `open()` с атрибутом "x". Проверка на то, что файла с таким именем не существует
  - `file_rename` - переименование файла. В качестве параметра метод принимает новое имя файла. Так же работает при условии, что имя файла передано в виде `source_dir/file_name`. Происходит проверка на то, что новое имя файла - директория, что у нас нет прав на создание файла и что такой файл уже есть.

- file\_read - читает все содержимое из файла и выводит сразу на экран
- file\_copy - копирование файла с таким же именем, но в другую директорию.
- WinReg
  - \_\_init\_\_ - инициализация переменных класса, создание сокета на реестр
  - key\_create - создание ключа через открытый дескриптор. Ключ создается с полными доступами
  - key\_data\_add - функция добавления значения в ключ. В качестве параметра передается данные в виде **N:T:V**, где **N** - название значения, **T** - тип данных, **V** - значение, которое записывается в ключ. Данные форматируются для выбранного типа в конструкции match-case
  - key\_delete - удаление ключа
  - close - закрытие дескрипторов на реестр

Важно! Все дополнительные ключи создаются в разделе  
HKLM\SOFTWARE\Lab1\

main содержит в себе инициализацию аргументов командной строки. После парсинга аргументов, в зависимости от того, какие аргументы указываются, происходит поиск символов, то есть названий функций, через метод getattr(). Благодаря этому вызываются функции из объектов класса.

## PEP8

Чтобы не придавать форматированию кода большое значение, было принято решение использовать автоформатирование, как расширение в VSCode. Поэтому, если были бы обнаружены ошибки в форматировании, как например это было с длиной строк, то они сразу бы вывелись во вкладку **Problems**

## Результаты работы

Ниже представлены скриншоты работы программы:

- Вспомогательная информация при работе со скриптом

```
PS D:\Education\Second_Course\Python-projects> & C:/Users/klmg/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/FileSystem.py -h
usage: FileSystem [-h] [-f FILE] [-k KEY] [--create] [--delete] [--rename FILENAME] [--write] [--read] [--copy DIRNAME] [--create-key] [--delete-key] [--set-value N:T:V]

A Python script that allows you to perform operations on Windows files and registry

options:
  -h, --help            show this help message and exit
  -f FILE, --file FILE  Work with files
  -k KEY, --key KEY     Work with Windows registry

Work with FileSystem:
  --create              Create file
  --delete              Delete file
  --rename FILENAME     Change file name to FILENAME
  --write               Write your input in the end of file
  --read               Read all file and print strings from it
  --copy DIRNAME        Copy file to another directory DIRNAME

Work with Windows Registry:
  --create-key          Add a new key in "dir"
  --delete-key          Delete key from directory
  --set-value N:T:V     Add value with name N, type T and value V. Value is REG_SZ, REG_BINARY, REG_DWORD, REG_QWORD
```

- Пример работы с файловой системой

```
PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 123.txt --write
Print string to file: kausdhskdjghfjllfkjdjkl
PS D:\Education\Second_Course\Python-projects> ls

Karanor: D:\Education\Second_Course\Python-projects

Mode                LastWriteTime         Length Name
----                -
d-----          10.02.2023   20:24             .git
d-----          10.02.2023   19:35             .vscode
d-----          13.02.2023   10:36             media
d-----          10.02.2023   20:16             src
-a-----          10.02.2023   20:24             16 .gitignore
-a-----          13.02.2023   19:39             26 123.txt
-a-----          13.02.2023   10:38             4595 filesystem.md
-a-----          06.02.2023   12:30             38 README.md

PS D:\Education\Second_Course\Python-projects> cat .\123.txt
kausdhskdjghfjllfkjdjkl
PS D:\Education\Second_Course\Python-projects>

PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 123.txt --delete
File deleted
PS D:\Education\Second_Course\Python-projects> ls

Karanor: D:\Education\Second_Course\Python-projects

Mode                LastWriteTime         Length Name
----                -
d-----          10.02.2023   20:24             .git
d-----          10.02.2023   19:35             .vscode
d-----          13.02.2023   10:36             media
d-----          10.02.2023   20:16             src
-a-----          10.02.2023   20:24             16 .gitignore
-a-----          13.02.2023   10:39             4788 filesystem.md
-a-----          06.02.2023   12:30             38 README.md

PS D:\Education\Second_Course\Python-projects>

PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 123.txt --write
Print string to file: ;djgkfoijffpoas;k;l
PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 123.txt --read
Data from file:
===START OF FILE===
;djgkfoijffpoas;k;l
===END OF FILE===
PS D:\Education\Second_Course\Python-projects>

PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 123.txt --rename 456.txt
File copied successfully.
PS D:\Education\Second_Course\Python-projects> ls

Karanor: D:\Education\Second_Course\Python-projects

Mode                LastWriteTime         Length Name
----                -
d-----          10.02.2023   20:24             .git
d-----          10.02.2023   19:35             .vscode
d-----          13.02.2023   10:36             media
d-----          10.02.2023   20:16             src
-a-----          10.02.2023   20:24             16 .gitignore
-a-----          13.02.2023   10:41             19 456.txt
-a-----          13.02.2023   10:40             4773 filesystem.md
-a-----          06.02.2023   12:30             38 README.md

PS D:\Education\Second_Course\Python-projects>

PS D:\Education\Second_Course\Python-projects> cd src
PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/Education/Second_Course/Python-projects/src/Filesystem.py -f 456.txt --copy src/
PS D:\Education\Second_Course\Python-projects> ls .\src\

Karanor: D:\Education\Second_Course\Python-projects\src

Mode                LastWriteTime         Length Name
----                -
-a-----          13.02.2023   10:42             19 456.txt
-a-----          10.02.2023   20:20             6781 filesystem.py

PS D:\Education\Second_Course\Python-projects>
```

- Пример работы с реестром

The screenshot shows a Windows development environment with a Python script and the Windows Registry Editor.

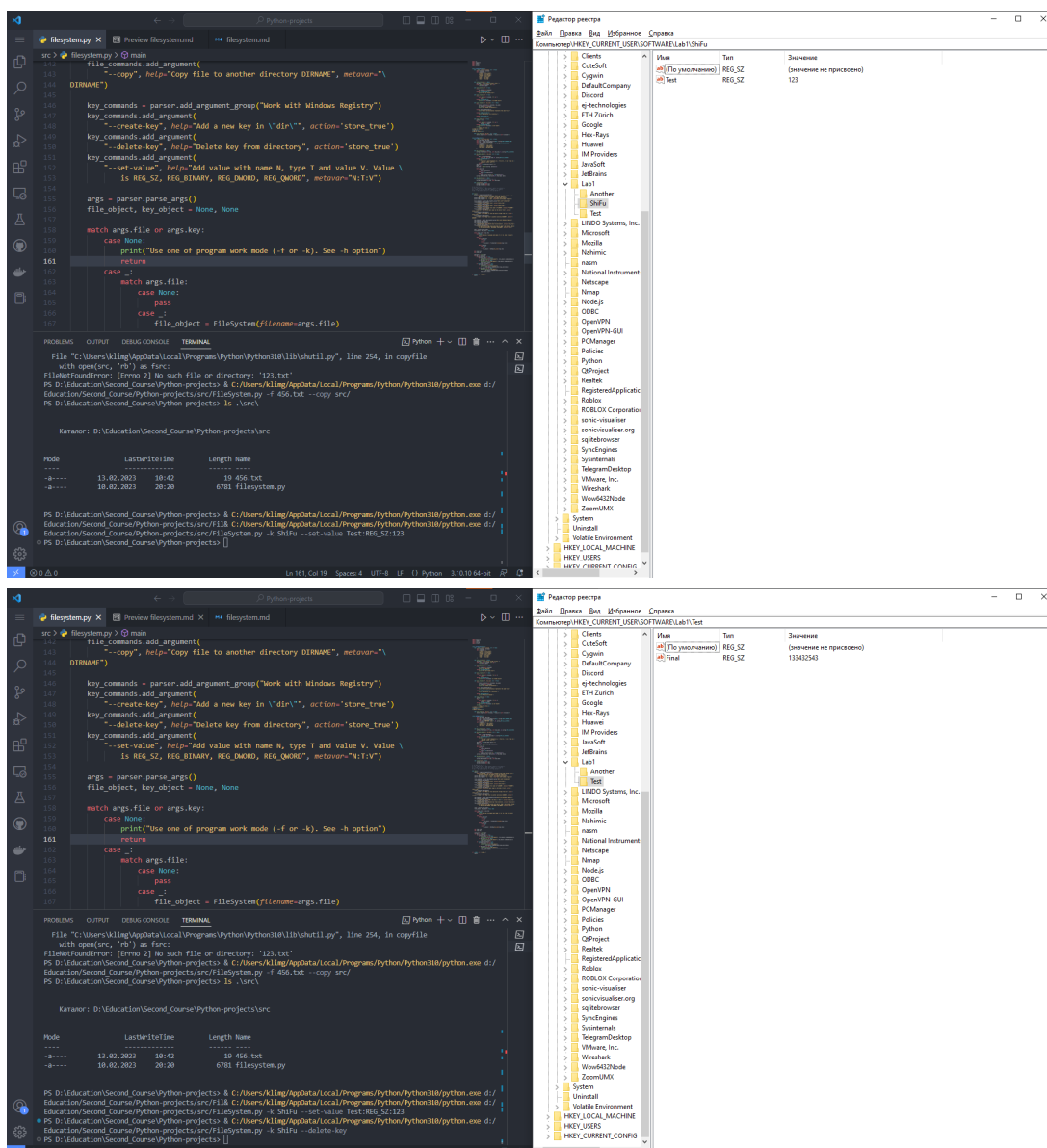
**Python Script (filesystem.py):**

```
src> cd filesystem
src> file_commands.add_argument(
    "--copy", help="Copy file to another directory DIRNAME", metavar="DIRNAME")
143
144
145 key_commands = parser.add_argument_group("Work with Windows Registry")
146 key_commands.add_argument(
    "--create-key", help="Add a new key in \"dir\", action='store_true')
147 key_commands.add_argument(
    "--delete-key", help="Delete key from directory", action='store_true')
148 key_commands.add_argument(
    "--set-value", help="Add value with name N, type T and value V. Value \
149 is REG_SZ, REG_BINARY, REG_DWORD, REG_QWORD", metavar="N:T:V")
150
151 args = parser.parse_args()
152 file_object, key_object = None, None
153 match args.file or args.key:
154     case None:
155         print("Use one of program work mode (-f or -k). See -h option")
156     return
157     case _:
158         match args.file:
159             case None:
160                 pass
161             case _:
162                 file_object = filesystem(filename=args.file)
163
164 PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
165 File "C:\Users\kling\AppData\Local\Programs\Python\Python310\lib\shutil.py", line 254, in copyfile
166 with open(src, 'rb') as fsrc:
167     fsrc.readinto(fdst)
168 FileNotFoundError: [Errno 2] No such file or directory: '123.txt'
169 PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/
170 Education/Second_Course/Python-projects/src/Filesystem.py -f 456.txt --copy src/
171 PS D:\Education\Second_Course\Python-projects> ls .\src\
172
173 Karanor: D:\Education\Second_Course\Python-projects\src
174
175 Mode                LastWriteTime         Length Name
176 ----                -
177 -a-----          13.02.2023   10:42             19 456.txt
178 -a-----          10.02.2023   20:20             6781 filesystem.py
179
180 PS D:\Education\Second_Course\Python-projects> & C:/Users/kling/AppData/Local/Programs/Python/Python310/python.exe d:/
181 Education/Second_Course/Python-projects/src/Filesystem.py -k Shifu --create-key
182 PS D:\Education\Second_Course\Python-projects>
```

**Windows Registry Editor:**

The Registry Editor shows the following structure:

- Computer\HKEY\_CURRENT\_USER\SOFTWARE\Lab1\Shifu
  - Shifu
    - Test
    - LPMS Systems, Inc.
    - Microsoft
    - Mozilla
    - Nabisco
    - nam
    - National Instrument
    - NatScope
    - Nmap
    - Nvidia
    - COBIC
    - OpenVPN
    - OpenVPN-GUI
    - PCManager
    - Polices
    - Python
    - QbProject
    - Rabbit
    - RegisteredApplication
    - Robins
    - ROBLOX Corporation
    - sonic-visualiser
    - sonicvisualiser.org
    - spthruviewer
    - SymEngine
    - SystemTools
    - TelegramDesktop
    - VMware, Inc.
    - WinShare
    - WinSCP
    - ZoomJMX



## Вывод

В ходе лабораторной работы были изучены принципы взаимодействия с файловой системой и реестром ОС Windows через метод модулей языка Python. Написана программа, позволяющая взаимодействовать с файлами и реестром.

## Ответы на контрольные вопросы

Чем компилируемый язык программирования отличается от интерпретируемого?  
Приведите примеры языков программирования для обоих типов

Компиляция - преобразование исходного кода в исполняемый файл, перевод в байт-код. Пример: C/C++ Интерпретация - "построчное" выполнение кода. Пример: Python

Про C/C++ и Python однако слышал, что эти языки имеют обе реализации, то есть Python можно так же сделать компилируемым.

Каким образом могут быть считаны входные данные для консольного приложения помимо использования модуля argparse?

Модуль sys хранит в себе объект argv - `sys.argv`

В чем разница открытия файла конструкцией:

with может хранить в себе несколько рабочих контекстов, сразу происходит закрытие контекстов, то есть не нужно вызывать функцию close. Под капотом вызывается `__exit__`

## Приложение

```
import os
import argparse
import shutil
import winreg
import struct

class FileSystem(object):
    """
    Windows File System class. There are some funcitons in it:\n
    ``create``: file_create,
    ``delete``: file_delete,
    ``rename``: file_rename,
    ``write``: file_write,
    ``read``: file_read,
    ``copy``: file_copy
    """
    def __init__(self, filename: str) -> None:
        self.filename = filename
        self.commands = {
            'create': 'file_create',
            'delete': 'file_delete',
            'rename': 'file_rename',
            'write': 'file_write',
            'read': 'file_read',
            'copy': 'file_copy'
        }

    def file_write(self) -> None:
        """
        Write input buffer to file
        """
        new_data = input("Print string to file: ")
        with open(self.filename, "a") as f:
            f.write(new_data)

    def file_delete(self) -> None:
```

```

    """
    Delete file by file name
    """
    try:
        os.remove(self.filename)
        print("File deleted")
    except FileNotFoundError:
        print("File not found error")

def file_create(self) -> None:
    """
    Create an empty file
    """
    try:
        with open(self.filename, "x") as f:
            pass
    except FileExistsError:
        print(f"{self.filename} is already exist")

def file_rename(self, new_name: str) -> None:
    """
    Rename file from arguments to file with name ``new_name``
    """
    try:
        shutil.copyfile(self.filename, new_name)
        os.remove(self.filename)
        print("File copied successfully.")

    except shutil.SameFileError:
        print("Source and destination represents the same file.")

    except IsADirectoryError:
        print("Destination is a directory.")

    except PermissionError:
        print("Permission denied.")

def file_read(self) -> None:
    """
    Read all data from file and print it
    """
    buffer = str()
    try:
        with open(self.filename, "r") as f:
            buffer = f.read()
    except FileNotFoundError:
        print(f"{self.filename} is not found")
    print(
        f"Data from file:\n\
===START OF FILE===\
\n{buffer}\n\

```

```

===END OF FILE===\n")

def file_copy(self, new_dir: str) -> None:
    """
    Copy file to another directory with name ``new_dir``
    """
    shutil.copyfile(self.filename, f"{new_dir}/{self.filename}")

class WinReg(object):
    def __init__(self, key_name: str) -> None:
        self.key_name = key_name
        self.dir = winreg.ConnectRegistry(None,
winreg.HKEY_CURRENT_USER)
        self.key = winreg.CreateKeyEx(
            self.dir, "SOFTWARE\\Lab1\\", 0, winreg.KEY_ALL_ACCESS)
        self.commands = {
            'create_key': 'key_create',
            'delete_key': 'key_delete',
            'set_value': 'key_data_add'
        }

    def key_create(self) -> None:
        """
        Create key in opened descriptor (HKLM\\SOFTWARE\\Lab1\\)
        """
        winreg.CreateKeyEx(self.key, self.key_name, 0,
winreg.KEY_ALL_ACCESS)

    def key_data_add(self, new_data: str) -> None:
        """
        Add data in to the key with different value types
        """
        try:
            key = winreg.OpenKeyEx(
                self.key, self.key_name, 0, winreg.KEY_ALL_ACCESS)
        except FileNotFoundError:
            print(
                "Не удастся найти указанный ключ. Возможно, он не
создан или \
                название введено неверно")
            return
        options = list(new_data.split(":"))
        data = 0
        key_type = getattr(winreg, options[1])

        match key_type:
            case 1:
                data = options[2]
            case 4 | 11:
                data = int(options[2])

```

```

        case 3:
            data = struct.pack('>i', int(options[2]))
            winreg.SetValueEx(key, options[0], 0, key_type, data)

def key_delete(self) -> None:
    """
    Delete key
    """
    winreg.DeleteKeyEx(self.key, self.key_name)

def close(self) -> None:
    winreg.CloseKey(self.dir)
    winreg.CloseKey(self.key)

def main():
    parser = argparse.ArgumentParser(
        prog="FileSystem", description="A Python script that allows
you to \
        perform operations on Windows files and registry")
    parser.add_argument("-f", "--file", help="Work with files")
    parser.add_argument("-k", "--key", help="Work with Windows
registry")

    file_commands = parser.add_argument_group("Work with FileSystem")
    file_commands.add_argument(
        "--create", help="Create file", action='store_true')
    file_commands.add_argument(
        "--delete", help="Delete file", action='store_true')
    file_commands.add_argument(
        "--rename", help="Change file name to FILENAME",
metavar="FILENAME")
    file_commands.add_argument(
        "--write", help="Write your input in the end of file",
action='\
store_true')
    file_commands.add_argument(
        "--read", help="Read all file and print strings from it",
action='\
store_true')
    file_commands.add_argument(
        "--copy", help="Copy file to another directory DIRNAME",
metavar="\
DIRNAME")

    key_commands = parser.add_argument_group("Work with Windows
Registry")
    key_commands.add_argument(
        "--create-key", help="Add a new key in \"dir\\",
action='store_true')
    key_commands.add_argument(

```



```

        "--delete-key", help="Delete key from directory",
        action='store_true')
    key_commands.add_argument(
        "--set-value", help="Add value with name N, type T and value
V. Value \
        is REG_SZ, REG_BINARY, REG_DWORD, REG_QWORD",
        metavar="N:T:V")

    args = parser.parse_args()
    file_object, key_object = None, None

    match args.file or args.key:
        case None:
            print("Use one of program work mode (-f or -k). See -h
option")
            return
        case _:
            match args.file:
                case None:
                    pass
                case _:
                    file_object = FileSystem(filename=args.file)

            match args.key:
                case None:
                    pass
                case _:
                    key_object = WinReg(key_name=args.key)

    del args.file
    del args.key

    args_dict = vars(args)
    for key in args_dict:
        if args_dict[key] is True:
            if file_object is not None:
                getattr(locals()["file_object"],
file_object.commands[key])()
            if key_object is not None:
                getattr(locals()["key_object"],
key_object.commands[key])()
            key_object.close()
        if type(args_dict[key]) is str:
            if file_object is not None:
                getattr(locals()["file_object"],
                    file_object.commands[key])(args_dict[key])
            if key_object is not None:
                getattr(locals()["key_object"],
                    key_object.commands[key])(args_dict[key])
            key_object.close()

```

```
if __name__ == "__main__":  
    main()
```