Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

# ОТЧЁТ
# ПО ЛАБОРАТОРНОЙ РАБОТЕ №5
## дисциплины «Анализ данных»
## Вариант 28

Выполнил:
Репкин Александр Павлович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной
техники и автоматизированных
систем», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р.А., канд. техн. наук,
доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

**Тема**: Работа с файловой системой в Python 3 с использованием модуля pathlib.

**Цель**: приобрести навыки работы с файловой системой в Python 3.x с использованием модуля pathlib.

**Порядок выполнения работы**:

1. Выполнено первое индивидуальное задание – модифицировано задание из лабораторной работы №3 дисциплины “Анализ Данных”. При помощи модуля pathlib добавлена возможность хранения файла данных в домашнем каталоге пользователя. Теперь, пользователю доступна работа с двумя каталогами – домашним и текущим (В котором находится файл Individual_1.py).

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import argparse
import json
import os.path
import pathlib


def new_human(people, name, surname, telephone, happy_birthday):
    """Добавить данные о человеке."""
    people.append({
        "name": name,
        "surname": surname,
        "telephone": telephone,
        "birthday": happy_birthday
    })
    return people


def display_people(people):
    """Отобразить список людей."""
    # Проверка, что в списке есть люди.
    if people:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 5, '-' * 25, '-' * 25, '-' * 25, '-' * 18)
        print(line)
        print('| {:^5} | {:^24} | {:^25} | {:^25} | {:^18} |'.format(
            "№", "Name", "Surname", "Telephone", "Birthday"))
        print(line)
        for number, human in enumerate(people, 1):
            print('| {:^5} | {:<24} | {:<25} | {:<25} | {:<18} |'.format(number, human.get(
                "name", ""), human.get("surname", ""),
                human.get("telephone", ""),
                human.get("birthday", "")))
        print(line)
```

Рисунок 1. Код индивидуального задания №1

```
(Data_Analysis) C:\Users\yabuz\GitHub\Data Analysis\Data_Analysis_5\Программы и файлы>python Individual_1.py display abra-codabra.json
There are no people in list!

(Data_Analysis) C:\Users\yabuz\GitHub\Data Analysis\Data_Analysis_5\Программы и файлы>python Individual_1.py display abra-codabra.json --own
There are no people in list!

(Data_Analysis) C:\Users\yabuz\GitHub\Data Analysis\Data_Analysis_5\Программы и файлы>python Individual_1.py add -n Boo -s Hoo -t 89188888888
 -b 01.01.2001 abra-codabra.json --own

(Data_Analysis) C:\Users\yabuz\GitHub\Data Analysis\Data_Analysis_5\Программы и файлы>python Individual_1.py display abra-codabra.json
There are no people in list!

(Data_Analysis) C:\Users\yabuz\GitHub\Data Analysis\Data_Analysis_5\Программы и файлы>python Individual_1.py display abra-codabra.json --own
+-------+--------------------------+---------------------------+---------------------------+--------------------+
|   №   |           Name           |          Surname          |         Telephone         |      Birthday      |
+-------+--------------------------+---------------------------+---------------------------+--------------------+
|   1   | Boo                      | Hoo                       | 89188888888               | 01.01.2001         |
+-------+--------------------------+---------------------------+---------------------------+--------------------+
```

Рисунок 2. Пример выполнения индивидуального задания №1

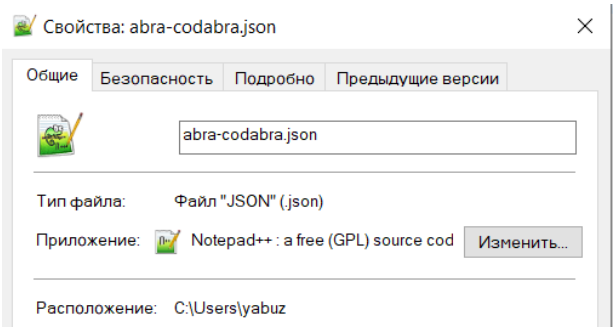


Рисунок 3. Получаемый файл в домашнем каталоге

2. Выполнено индивидуальное задание №2 – реализован аналог команды tree (Команда, выводящая содержимое каталогов в Linux) при помощи модуля pathlib. Добавленные ключи: -a (Вывод даже скрытых файлов), -d\-f (Вывод либо только каталогов, либо только файлов), -p число (Допустимая глубина), -t (Вывод полного пути до файла, а не только его имя и расширение).

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


import argparse
import sys
import pathlib

# Аналог команде tree в Linux.


def tree(directory, args, prefix="", level=0):
    """Рекурсивный вывод содержимого каталога."""
    # Проверка глубины рекурсии. Если задан -p и текущая глубина превышает его, то прекращается дальнейшее выполнение.
    if args.p is not None and level > args.p:
        return
    # Получение содержимого текущего каталога.
    contents = list(directory.iterdir())
    # Если не задан -a, то НЕ учитываются скрытые файлы (Те, у которых в начале точка. Пример - .idea).
    if not args.a:
        filtered_contents = []
        for file in contents:
            if not file.name.startswith("."):
                filtered_contents.append(file)
        contents = filtered_contents

    # Если задан -d, то выводятся только каталоги.
    if args.d:
        filtered_contents = []
        for file in contents:
            if file.is_dir():  # Проверка, каталог ли.
                filtered_contents.append(file)
        contents = filtered_contents

    # Если задан -f, выводятся только не каталоги.
    if args.f:
        filtered_contents = []
```

Рисунок 4. Код индивидуального задания №2

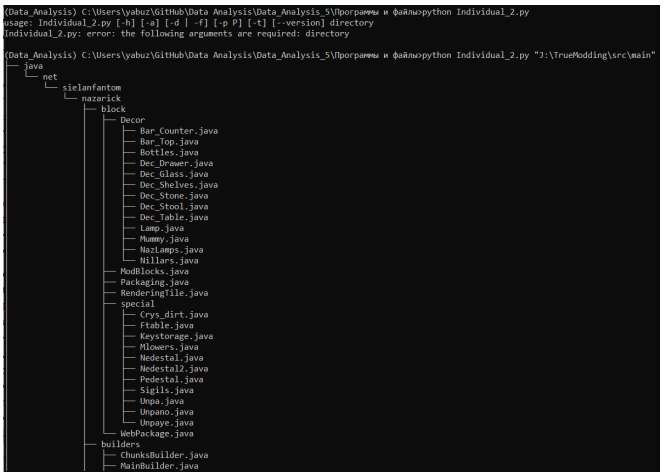


Рисунок 5. Пример выполнения индивидуального задания №2, без ключей

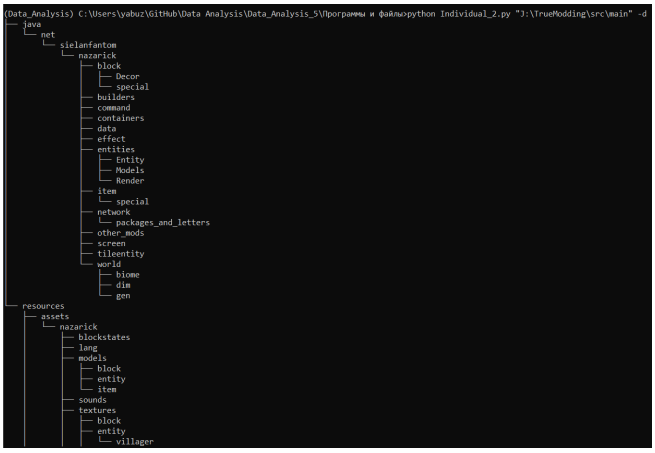


Рисунок 6. Пример выполнения индивидуального задания №2, с -d

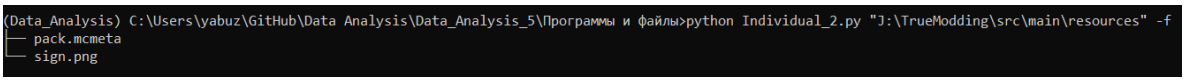


Рисунок 7. Пример выполнения индивидуального задания №2, с -f

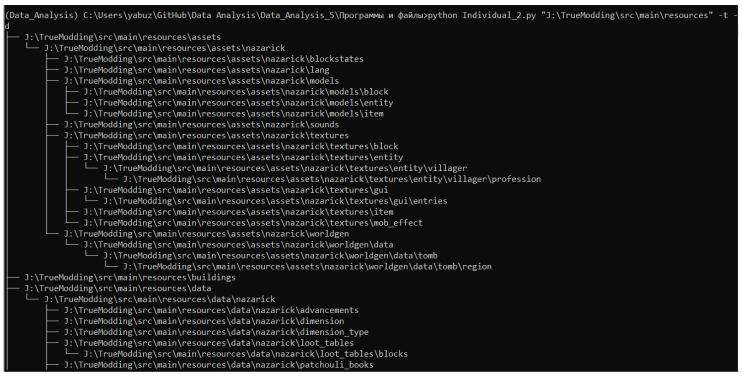


Рисунок 8. Пример выполнения индивидуального задания №2, с -t

3. Ответы на вопросы.

1) Какие существовали средства для работы с файловой системой до Python 3.4?

**Ответ:** до Python 3.4 использовались модули os и os.path для работы с файловой системой.

2) Что регламентирует PEP 428?

**Ответ:** PEP 428 регламентирует "Представление дерева каталогов в стандартной библиотеке Python".

3) Как осуществляется создание путей средствами модуля pathlib?

**Ответ:** для создания путей средствами модуля pathlib используется метод Path() с указанием нужного пути.

4) Как получить путь дочернего элемента файловой системы с помощью модуля pathlib?

**Ответ:** для получения пути дочернего элемента файловой системы с помощью модуля pathlib используется метод resolve() или оператор /.

5) Как получить путь к родительским элементам файловой системы с помощью модуля pathlib?

**Ответ:** для получения пути к родительским элементам файловой системы с помощью модуля pathlib используется атрибут parent.

6) Как выполняются операции с файлами с помощью модуля pathlib?

**Ответ:** операции с файлами с помощью модуля pathlib выполняются путём создания объектов Path, которые можно использовать для навигации по файловой системе, проверки существования файлов/директорий, создания/удаления файлов и директорий, т.д..

7) Как можно выделить компоненты пути файловой системы с помощью модуля pathlib?

**Ответ:** для выделения компонентов пути файловой системы с помощью модуля pathlib можно использовать различные атрибуты объектов Path, такие как: name, suffix, parent и т.д., чтобы получить имя файла, его расширение, родительский каталог и прочее.

8) Как выполнить перемещение и удаление файлов с помощью модуля pathlib?

**Ответ:** для перемещения файлов с помощью модуля pathlib можно использовать метод rename() или replace(), а для удаления – метод unlink().

9) Как выполнить подсчет файлов в файловой системе?

**Ответ:** для подсчета файлов можно использовать рекурсивную функцию, которая пройдется по всем каталогам и файлам.

10) Как отобразить дерево каталогов файловой системы?

**Ответ:** для отображения дерева каталогов файловой системы можно использовать рекурсивную функцию, которая пройдется по всем каталогам и файлам, выводя их структуру.

11) Как создать уникальное имя файла?

**Ответ:** для создания уникального имени файла можно использовать функции модуля tempfile или генерировать уникальные имена на основе времени, случайных чисел или других уникальных идентификаторов.

12) Каковы отличия в использовании модуля pathlib для различных операционных систем?

**Ответ:** основное отличие в использовании модуля pathlib для различных ОС заключается в разделителе каталогов: для Windows используется обратный слеш \, а для Unix-подобных систем – /. Кроме того, есть различия в поддерживаемых атрибутах файловых систем и их названиях, но в большинстве случаев модуль pathlib абстрагируется от этих различий, обеспечивая удобный интерфейс для работы с путями вне зависимости от ОС.

**Вывод**: в ходе выполнения лабораторной работы, приобретены навыки работы с файловой системой в Python 3.x с использованием модуля pathlib.