

#### Задание 4 (семинар):

Функция получает на вход текст вида: «2-й четверг ноября», «3-я среда мая» и т.п.  
Преобразуйте его в дату в текущем году  
Логируйте ошибки если текст не соответствует формату

```
import logging
from datetime import datetime, timedelta
import calendar
import re

# Настройка логирования
logging.basicConfig(filename='date_conversion_errors.log', level=logging.ERROR,
                    format='%(asctime)s: %(levelname)s: %(message)s')

# Сопоставление русских и английских названий дней недели и месяцев
weekday_translation = {
    'понедельник': 'Monday',
    'вторник': 'Tuesday',
    'среда': 'Wednesday',
    'четверг': 'Thursday',
    'пятница': 'Friday',
    'суббота': 'Saturday',
    'воскресенье': 'Sunday'
}

month_translation = {
    'января': 'January',
    'февраля': 'February',
    'марта': 'March',
    'апреля': 'April',
    'мая': 'May',
```

```
'июня': 'June',  
'июля': 'July',  
'августа': 'August',  
'сентября': 'September',  
'октября': 'October',  
'ноября': 'November',  
'декабря': 'December'  
}
```

```
def convert_to_date(text):
```

```
    # Регулярное выражение для парсинга входного текста
```

```
    pattern = re.compile(r"(\d+)-(й|я) (\w+) (\w+)")
```

```
    match = pattern.match(text)
```

```
    # Если формат не соответствует, логируем ошибку
```

```
    if not match:
```

```
        logging.error(f"Некорректный формат: {text}")
```

```
        return None
```

```
    # Извлечение данных из входного текста
```

```
    week_number, week_suffix, weekday_name, month_name = match.groups()
```

```
    try:
```

```
        # Конвертация номера недели в целое число
```

```
        week_number = int(week_number)
```

```
    # Получение текущего года
```

```
    current_year = datetime.now().year
```

```
    # Перевод названий на английский для работы с модулем datetime
```

```
    month_name_eng = month_translation[month_name.lower()]
```

```

weekday_name_eng = weekday_translation[weekday_name.lower()]

# Получение номера месяца
month_number = list(calendar.month_name).index(month_name_eng)

# Получение номера дня недели
weekday_number = list(calendar.day_name).index(weekday_name_eng)
except (ValueError, KeyError) as e:

# Логирование ошибки при конвертации данных
logging.error(f"Ошибка при конвертации: {text} - {e}")
return None

# Проверка корректности номера недели
if week_number < 1 or week_number > 5:
    logging.error(f"Неверный номер недели: {week_number}")
    return None

# Находим первый день месяца
first_day_of_month = datetime(current_year, month_number, 1)
first_weekday_of_month = first_day_of_month.weekday()

# Рассчитываем смещение до нужного дня недели
days_to_add = (weekday_number - first_weekday_of_month + 7) % 7
first_occurrence = first_day_of_month + timedelta(days=days_to_add)

# Находим нужный день
target_date = first_occurrence + timedelta(weeks=week_number-1)

# Проверка, что дата находится в правильном месяце
if target_date.month != month_number:

```

```
logging.error(f'Недопустимая дата: {text}')
return None
```

```
return target_date
```

# Пример использования функции с выводом успешного результата

```
date_texts = ["2-й четверг ноября", "3-я среда мая", "5-й понедельник января",
"$$$ ", "среда ноября", "третья среда мая", "6-й понедельник января", "2-й четвёрг
ноябрь", "3-я среда июля"]
```

```
for text in date_texts:
```


```
    date = convert_to_date(text)
```

```
    if date:
```

```
        print(f'Дата для '{text}': {date.strftime('%Y-%m-%d')}')
```

```
    else:
```

```
        print(f'Не удалось конвертировать дату для '{text}')
```

 date\_conversion\_errors.log – Блокнот

Файл Правка Формат Вид Справка

```
2024-08-01 15:49:09,016:ERROR:Некорректный формат: $$$
2024-08-01 15:49:09,016:ERROR:Некорректный формат: среда ноября
2024-08-01 15:49:09,017:ERROR:Некорректный формат: третья среда мая
2024-08-01 15:49:09,017:ERROR:Неверный номер недели: 6
2024-08-01 15:49:09,017:ERROR:Ошибка при конвертации: 2-й четвёрг ноябрь - 'ноябрь'
```

Без кода к 4 заданию не хотел писать решение пятого задания

5 задание:

Добавьте возможность запуска из командной строки. При этом значение любого параметра можно опустить. В этом случае берётся первый в месяце день недели, текущий день недели и/или текущий месяц.

```
import logging
```

```
from datetime import datetime, timedelta
```

```
import calendar
import re
import argparse
```

```
# Настройка логирования
```

```
logging.basicConfig(filename='date_conversion_errors.log', level=logging.ERROR,
                    format='%(asctime)s: %(levelname)s: %(message)s')
```

```
# Сопоставление русских и английских названий дней недели и месяцев
```

```
weekday_translation = {
    'понедельник': 'Monday',
    'вторник': 'Tuesday',
    'среда': 'Wednesday',
    'четверг': 'Thursday',
    'пятница': 'Friday',
    'суббота': 'Saturday',
    'воскресенье': 'Sunday'
}
```

```
month_translation = {
    'января': 'January',
    'февраля': 'February',
    'марта': 'March',
    'апреля': 'April',
    'мая': 'May',
    'июня': 'June',
    'июля': 'July',
    'августа': 'August',
    'сентября': 'September',
    'октября': 'October',
    'ноября': 'November',
}
```

```
'декабря': 'December'  
}
```

```
def convert_to_date(text):
```

```
    # Регулярное выражение для парсинга входного текста
```

```
    pattern = re.compile(r"(\d*)-?(й|я)? ?(\w+)? ?(\w+)?")
```

```
    match = pattern.match(text)
```

```
    # Если формат не соответствует, логируем ошибку
```

```
    if not match:
```

```
        logging.error(f"Некорректный формат: {text}")
```

```
        return None
```

```
    # Извлечение данных из входного текста
```

```
    week_number, week_suffix, weekday_name, month_name = match.groups()
```

```
    # Получение текущих значений
```

```
    current_date = datetime.now()
```

```
    current_weekday_name = current_date.strftime('%A').lower()
```

```
    current_month_name = current_date.strftime('%B').lower()
```

```
    # Установка значений по умолчанию
```

```
    week_number = int(week_number) if week_number else 1
```

```
    weekday_name = weekday_name if weekday_name else current_weekday_name
```

```
    month_name = month_name if month_name else current_month_name
```

```
    try:
```

```
        # Перевод названий на английский для работы с модулем datetime
```

```
        month_name_eng = month_translation[month_name.lower()]
```

```
        weekday_name_eng = weekday_translation[weekday_name.lower()]
```

```
        # Получение номера месяца
```

```
        month_number = list(calendar.month_name).index(month_name_eng)
```

```
# Получение номера дня недели
weekday_number = list(calendar.day_name).index(weekday_name_eng)
except (ValueError, KeyError) as e:
    # Логирование ошибки при конвертации данных
    logging.error(f"Ошибка при конвертации: {text} - {e}")
    return None
```

```
# Проверка корректности номера недели
if week_number < 1 or week_number > 5:
    logging.error(f"Неверный номер недели: {week_number}")
    return None
```

```
# Находим первый день месяца
first_day_of_month = datetime(current_date.year, month_number, 1)
first_weekday_of_month = first_day_of_month.weekday()
```

```
# Рассчитываем смещение до нужного дня недели
days_to_add = (weekday_number - first_weekday_of_month + 7) % 7
first_occurrence = first_day_of_month + timedelta(days=days_to_add)
```

```
# Находим нужный день
target_date = first_occurrence + timedelta(weeks=week_number-1)
```

```
# Проверка, что дата находится в правильном месяце
if target_date.month != month_number:
    logging.error(f"Недопустимая дата: {text}")
    return None
```

```
return target_date
```

```
def main():
```

```
parser = argparse.ArgumentParser(description="Конвертация текстового  
описания даты в дату текущего года.")  
args = parser.parse_args()
```

```
while True:  
    user_input = input("Введите желаемую дату для анализа (или нажмите Enter  
для выхода): ").strip()  
    if not user_input:  
        break  
    date = convert_to_date(user_input)  
    if date:  
        print(f'Дата для '{user_input}': {date.strftime('%Y-%m-%d')}')  
    else:  
        print(f'Не удалось конвертировать дату для '{user_input}')
```

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

Для проверки использую следующие фразы:

среда ноября

2-й четверг в ноябре

6-й понедельник января

0-й вторник февраля

2-й четвёрг ноябрь

3-я среда МАй

2-й ноября

четверг ноября


2-й четверг

ноября



;;;

```
Введите желаемую дату для анализа (или нажмите Enter для выхода): среда ноября
я
Дата для 'среда ноября': 2024-11-06
Введите желаемую дату для анализа (или нажмите Enter для выхода): 2-й четверг
в ноябре
Не удалось конвертировать дату для '2-й четверг в ноябре'
Введите желаемую дату для анализа (или нажмите Enter для выхода): 6-й понедель
ьник января
Не удалось конвертировать дату для '6-й понедельник января'
Введите желаемую дату для анализа (или нажмите Enter для выхода): 0-й вторник
февраля
Не удалось конвертировать дату для '0-й вторник февраля'
Введите желаемую дату для анализа (или нажмите Enter для выхода): 2-й четвёрг
ноябрь
Не удалось конвертировать дату для '2-й четвёрг ноябрь'
Введите желаемую дату для анализа (или нажмите Enter для выхода): 3-я срада м
Ай
Не удалось конвертировать дату для '3-я срада МАЙ'
Введите желаемую дату для анализа (или нажмите Enter для выхода): 2-й ноября
Не удалось конвертировать дату для '2-й ноября'
Введите желаемую дату для анализа (или нажмите Enter для выхода): четверг ноябр
Дата для 'четверг ноября': 2024-11-07
Введите желаемую дату для анализа (или нажмите Enter для выхода): 2-й четверг
Не удалось конвертировать дату для '2-й четверг'
Введите желаемую дату для анализа (или нажмите Enter для выхода): ноября
Не удалось конвертировать дату для 'ноября'
Введите желаемую дату для анализа (или нажмите Enter для выхода): ;;;
```

 date\_conversion\_errors.log – Блокнот

Файл Правка Формат Вид Справка

```
2024-08-01 17:32:57,838:ERROR:Некорректный формат: среда ноября
2024-08-01 17:34:39,704:ERROR:Ошибка при конвертации: 2-й четверг в ноябре - 'в'
2024-08-01 17:34:52,242:ERROR:Неверный номер недели: 6
2024-08-01 17:35:00,483:ERROR:Неверный номер недели: 0
2024-08-01 17:35:06,979:ERROR:Ошибка при конвертации: 2-й четвёрг ноябрь - 'ноябрь'
2024-08-01 17:35:13,164:ERROR:Ошибка при конвертации: 3-я срада МАЙ - 'май'
2024-08-01 17:35:18,260:ERROR:Ошибка при конвертации: 2-й ноября - 'august'
2024-08-01 17:35:28,549:ERROR:Ошибка при конвертации: 2-й четверг - 'august'
2024-08-01 17:35:34,302:ERROR:Ошибка при конвертации: ноября - 'august'
2024-08-01 17:35:44,863:ERROR:Ошибка при конвертации: ;;; - 'august'
```

Задание №6 Напишите код, который запускается из командной строки и получает на вход путь до директории на ПК. Соберите информацию о содержимом в виде объектов namedtuple. Каждый объект хранит: ○ имя файла без расширения или название каталога, ○ расширение, если это файл, ○ флаг каталога, ○ название родительского каталога. В процессе сбора сохраните данные в текстовый файл используя логирование.

```
import os
import logging
from collections import namedtuple

# Настройка логирования
logging.basicConfig(filename='directory_contents.log', level=logging.DEBUG,
                    format='%(asctime)s: %(levelname)s: %(message)s')

# Определение структуры данных
FileInfo = namedtuple('FileInfo', ['name', 'extension', 'is_directory', 'parent_directory'])

def collect_directory_info(directory_path):
    """Собирает информацию о содержимом директории."""
    if not os.path.isdir(directory_path):
        logging.error(f"Указанный путь не является директорией: {directory_path}")
        print(f"Ошибка: '{directory_path}' не является директорией.")
        return

    logging.info(f"Начинаем обработку директории: {directory_path}")

    try:
        # Проход по всем каталогам и файлам в указанной директории
```

```

for root, dirs, files in os.walk(directory_path):
    logging.info(f"Обрабатываем каталог: {root}")

    for name in dirs:
        # Для каталогов
        dir_path = os.path.join(root, name)
        parent_dir = os.path.basename(root)
        file_info = FileInfo(
            name=name,
            extension=None,
            is_directory=True,
            parent_directory=parent_dir
        )
        logging.info(f"Каталог: {file_info}")

    for name in files:
        # Для файлов
        file_path = os.path.join(root, name)
        parent_dir = os.path.basename(root)
        name_without_ext, ext = os.path.splitext(name)
        file_info = FileInfo(
            name=name_without_ext,
            extension=ext[1:], # убираем точку из расширения
            is_directory=False,
            parent_directory=parent_dir
        )
        logging.info(f"Файл: {file_info}")

except PermissionError as e:
    logging.error(f"Ошибка доступа к директории '{directory_path}': {e}")
except FileNotFoundError as e:
    logging.error(f"Файл или директория не найдены при обработке '{directory_path}': {e}")

```

```

except OSError as e:
    logging.error(f'OS ошибка при обработке '{directory_path}': {e}')
except Exception as e:
    logging.error(f'Неизвестная ошибка при сборе информации из
' '{directory_path}': {e}')

def main():
    while True:
        # Запрос директорий у пользователя
        directories_input = input("Введите пути до директорий, разделенные
пробелом (или нажмите Enter для выхода): ")

        # Проверка на пустую строку
        if not directories_input.strip():
            print("Завершение программы.")
            break

        directories = directories_input.split()

        # Обработка каждой директории
        for directory in directories:
            if os.path.isdir(directory):
                print(f'Обрабатывается директория: {directory}')
                collect_directory_info(directory)
            else:
                # Если директория не существует, записываем ошибку в лог
                logging.error(f'Указанный путь не является директорией: {directory}')
                print(f'Ошибка: '{directory}' не является директорией.')

if __name__ == "__main__":
    main()

```

```
directory_contents.log - Блокнот
Файл Правка Формат Вид Справка
2024-08-01 17:53:49,469:INFO:Начинаем обработку директории: C:\Users\RGB-5\Documents\sinobi-1
2024-08-01 17:53:49,469:INFO:Обрабатываем каталог: C:\Users\RGB-5\Documents\sinobi-1
2024-08-01 17:53:49,470:INFO:Каталог: FileInfo(name='myenv', extension=None, is_directory=True, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Каталог: FileInfo(name='ninjutsu_game', extension=None, is_directory=True, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='date_conversion_errors', extension='log', is_directory=False, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='directory_contents', extension='log', is_directory=False, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='task4', extension='py', is_directory=False, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='task5', extension='py', is_directory=False, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='task6', extension='py', is_directory=False, parent_directory='sinobi-1')
2024-08-01 17:53:49,470:INFO:Обрабатываем каталог: C:\Users\RGB-5\Documents\sinobi-1\myenv
2024-08-01 17:53:49,470:INFO:Каталог: FileInfo(name='Include', extension=None, is_directory=True, parent_directory='myenv')
2024-08-01 17:53:49,470:INFO:Каталог: FileInfo(name='Lib', extension=None, is_directory=True, parent_directory='myenv')
2024-08-01 17:53:49,470:INFO:Каталог: FileInfo(name='Scripts', extension=None, is_directory=True, parent_directory='myenv')
2024-08-01 17:53:49,470:INFO:Файл: FileInfo(name='pyvenv', extension='cfg', is_directory=False, parent_directory='myenv')
2024-08-01 17:53:49,470:INFO:Обрабатываем каталог: C:\Users\RGB-5\Documents\sinobi-1\myenv\Include
2024-08-01 17:53:49,470:INFO:Обрабатываем каталог: C:\Users\RGB-5\Documents\sinobi-1\myenv\Lib
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='site-packages', extension=None, is_directory=True, parent_directory='Lib')
2024-08-01 17:53:49,471:INFO:Обрабатываем каталог: C:\Users\RGB-5\Documents\sinobi-1\myenv\Lib\site-packages
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='addodbapi', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='aiohttp', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='aiohttp-3.9.5.dist-info', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='aiosignal', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='aiosignal-1.3.1.dist-info', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='asgiref', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='asgiref-3.8.1.dist-info', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='attr', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='attrs', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='attrs-23.2.0.dist-info', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='bitarray', extension=None, is_directory=True, parent_directory='site-packages')
2024-08-01 17:53:49,471:INFO:Каталог: FileInfo(name='bitarray-2.9.2.dist-info', extension=None, is_directory=True, parent_directory='site-packages')
```

1\*

## Урок 5. Генераторы и генераторные выражения

Напишите функцию `get_file_info`, которая принимает на вход строку - абсолютный путь до файла. Функция возвращает кортеж из трёх элементов: путь, имя файла, расширение файла.

**Пример использования.**

На входе:

```
file_path = "C:/Users/User/Documents/example.txt"
```

На выходе:

```
('C:/Users/User/Documents/', 'example', '.txt')
```

**Задание:** Добавить логирование ошибок и полезной информации. Также реализуйте возможность запуска из командной строки с передачей параметров

```
import os
import logging
import time
from datetime import datetime
```

```
# Настройка логирования
```

```
logging.basicConfig(filename='file_info.log', level=logging.DEBUG,  
                    format='%(asctime)s:%(levelname)s:%(message)s')
```

```
# Настройка логирования пользовательских действий
```

```
user_log = logging.getLogger('user_log')
```

```
user_log.setLevel(logging.DEBUG)
```

```
fh = logging.FileHandler('user_activity.log')
```

```
formatter = logging.Formatter('%(asctime)s:%(message)s')
```

```
fh.setFormatter(formatter)
```

```
user_log.addHandler(fh)
```

```
def get_file_info(file_path):
```

```
    """Получает информацию о файле: путь, имя и расширение."""
```

```
    try:
```

```
        # Проверка существования пути
```

```
        if not os.path.exists(file_path):
```

```
            raise FileNotFoundError(f"Путь не существует: {file_path}")
```

```
        # Получаем имя файла с расширением
```

```
        file_name_with_ext = os.path.basename(file_path)
```

```
        # Разделяем имя файла на имя и расширение
```

```
        file_name, file_extension = os.path.splitext(file_name_with_ext)
```

```
        # Получаем путь к директории
```

```
        directory_path = os.path.dirname(file_path)
```

```
        # Логируем успешную обработку
```

```
        logging.info(f"Успешно обработан файл: {file_path}")
```

```
        # Возвращаем кортеж с результатами
```

```

    return (directory_path, file_name, file_extension)

except Exception as e:
    logging.error(f'Ошибка при обработке пути '{file_path}': {e}')
    return (None, None, None)

def main():
    file_paths = []
    start_time = time.time()
    user_log.info("Начало работы программы")

    while True:
        path = input("Введите путь к файлу (или 'stop' для завершения ввода): ")
        user_log.info(f'Ввод пользователя: {path}')

        if path.lower() == 'stop':
            break

        file_paths.append(path)

    total_files = len(file_paths)
    user_log.info(f'Количество введенных путей: {total_files}')

    for path in file_paths:
        start_request_time = time.time()
        print(f'Обрабатывается путь: {path}')
        file_info = get_file_info(path)
        end_request_time = time.time()

        if file_info[0] is not None:
            print(f'Путь: {file_info[0]}')

```

```
    print(f'Имя файла: {file_info[1]}')
    print(f'Расширение: {file_info[2]}')
else:
    print(f'Не удалось получить информацию о файле: {path}')

user_log.info(f'Ответ программы для {path}: {file_info}')
    user_log.info(f'Время обработки запроса: {end_request_time -
start_request_time:.2f} секунд")

end_time = time.time()
total_time = end_time - start_time
user_log.info(f'Общее время работы программы: {total_time:.2f} секунд")
print(f'Общее количество введенных путей: {total_files}')
print(f'Общее время работы программы: {total_time:.2f} секунд")

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



```
file_info.log - Блокнот
Файл Правка Формат Вид Справка
2024-08-04 16:07:42,828:INFO:Начало работы программы
2024-08-04 16:07:46,803:INFO:Ввод пользователя: D:\task\example1.txt
2024-08-04 16:07:50,574:INFO:Ввод пользователя: D:\task\
2024-08-04 16:07:52,932:INFO:Ввод пользователя: D:\task\missing_file.txt
2024-08-04 16:07:56,239:INFO:Ввод пользователя: D:\non_existent_directory\
2024-08-04 16:08:03,269:INFO:Ввод пользователя: D:\task\
2024-08-04 16:08:06,538:INFO:Ввод пользователя: D:\task\invalid_file.txt
2024-08-04 16:08:11,819:INFO:Ввод пользователя: D:\task\no_extension_file
2024-08-04 16:08:16,870:INFO:Ввод пользователя:
2024-08-04 16:08:23,150:INFO:Ввод пользователя: stop
2024-08-04 16:08:23,150:INFO:Количество введенных путей: 8
2024-08-04 16:08:23,151:INFO:Успешно обработан файл: D:\task\example1.txt
2024-08-04 16:08:23,151:INFO:Ответ программы для D:\task\example1.txt : ('D:\\task', 'example1', '.txt ')
2024-08-04 16:08:23,151:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,151:INFO:Успешно обработан файл: D:\task\
2024-08-04 16:08:23,151:INFO:Ответ программы для D:\task\ : ('D:\\task', '', '')
2024-08-04 16:08:23,152:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,152:ERROR:Ошибка при обработке пути 'D:\task\missing_file.txt': Путь не существует: D:\task\missing_file.txt
2024-08-04 16:08:23,152:INFO:Ответ программы для D:\task\missing_file.txt: (None, None, None)
2024-08-04 16:08:23,153:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,153:ERROR:Ошибка при обработке пути 'D:\non_existent_directory\': Путь не существует: D:\non_existent_directory\
2024-08-04 16:08:23,153:INFO:Ответ программы для D:\non_existent_directory\ : (None, None, None)
2024-08-04 16:08:23,153:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,153:INFO:Успешно обработан файл: D:\task\
2024-08-04 16:08:23,153:INFO:Ответ программы для D:\task\ : ('D:\\task', '', '')
2024-08-04 16:08:23,154:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,154:ERROR:Ошибка при обработке пути 'D:\task\invalid_file.txt': Путь не существует: D:\task\invalid_file.txt
2024-08-04 16:08:23,154:INFO:Ответ программы для D:\task\invalid_file.txt: (None, None, None)
2024-08-04 16:08:23,154:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,154:ERROR:Ошибка при обработке пути 'D:\task\no_extension_file': Путь не существует: D:\task\no_extension_file
2024-08-04 16:08:23,154:INFO:Ответ программы для D:\task\no_extension_file: (None, None, None)
2024-08-04 16:08:23,154:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,154:ERROR:Ошибка при обработке пути '': Путь не существует:
2024-08-04 16:08:23,154:INFO:Ответ программы для : (None, None, None)
2024-08-04 16:08:23,154:INFO:Время обработки запроса: 0.00 секунд
2024-08-04 16:08:23,154:INFO:Общее время работы программы: 40.33 секунд
```

2\*

## Урок 6 модули. Задание 1.

### Проверка корректности даты

Вы работаете над разработкой программы для проверки корректности даты, введенной пользователем. На вход будет подаваться дата в формате "день.месяц.год". Ваша задача - создать программу, которая проверяет, является ли введенная дата корректной или нет.

Ваша программа должна предоставить ответ "True" (дата корректна) или "False" (дата некорректна) в зависимости от результата проверки.

```
import logging
from datetime import datetime
import time
```

```
# Настройка логирования
```

```
logging.basicConfig(filename='date_validation.log', level=logging.DEBUG,
                    format='%(asctime)s:%(levelname)s:%(message)s')
```

```
def is_valid_date(date_str):
    if not date_str:
```

```

logging.warning("Пустая строка введена.")
return False

if len(date_str) != 10:
    logging.error(f"Некорректная длина строки для даты: {date_str}")
    return False

if date_str.count('.') != 2:
    logging.error(f"Некорректный формат даты (ожидалось 'день.месяц.год'): {date_str}")
    return False

day, month, year = date_str.split('.')
if not (day.isdigit() and month.isdigit() and year.isdigit()):
    logging.error(f"Дата содержит недопустимые символы: {date_str}")
    return False

try:
    # Попытка преобразовать строку в объект datetime
    datetime.strptime(date_str, "%d.%m.%Y")
    logging.info(f"Дата {date_str} корректна.")
    return True
except ValueError as e:
    logging.error(f"Ошибка проверки даты {date_str}: {e}")
    return False

def main():
    logging.info("Программа запущена.")
    start_program_time = time.time() # Время начала работы программы
    count = 0

    while True:
        date_input = input("Введите дату в формате 'день.месяц.год' (или оставьте пустым для выхода): ")
        if date_input.strip() == "":
            logging.info("Пользователь завершил программу.")
            print("Выход из программы.")
            break

    logging.info("////////////////////")

```

```
start_time = time.time() # Время начала анализа даты
logging.info(f"Начало анализа даты: {date_input}")

result = is_valid_date(date_input)

end_time = time.time() # Время окончания анализа даты
elapsed_time = end_time - start_time
    logging.info(f"Окончание анализа даты: {date_input}. Время анализа:
{elapsed_time:.4f} секунд.")
    logging.info("////////////////////")

count += 1
print(f"Дата {date_input} корректна: {result}")

end_program_time = time.time() # Время окончания работы программы
total_elapsed_time = end_program_time - start_program_time
    logging.info(f"Программа завершена. Обработано {count} дат. Общее время
работы программы: {total_elapsed_time:.4f} секунд.")

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

```

2024-08-04 15:47:39,792:INFO:Программа запущена.
2024-08-04 15:48:04,712:INFO:Программа запущена.
2024-08-04 15:48:13,859:INFO://///////////////
2024-08-04 15:48:13,860:INFO:Начало анализа даты: 31.12.2020
2024-08-04 15:48:13,863:INFO:Дата 31.12.2020 корректна.
2024-08-04 15:48:13,863:INFO:Окончание анализа даты: 31.12.2020. Время анализа: 0.0037 секунд.
2024-08-04 15:48:13,863:INFO://///////////////
2024-08-04 15:48:18,116:INFO://///////////////
2024-08-04 15:48:18,116:INFO:Начало анализа даты: 01.01.2021
2024-08-04 15:48:18,116:INFO:Дата 01.01.2021 корректна.
2024-08-04 15:48:18,116:INFO:Окончание анализа даты: 01.01.2021. Время анализа: 0.0000 секунд.
2024-08-04 15:48:18,116:INFO://///////////////
2024-08-04 15:48:22,428:INFO://///////////////
2024-08-04 15:48:22,428:INFO:Начало анализа даты: 32.01.2020
2024-08-04 15:48:22,428:ERROR:Ошибка проверки даты 32.01.2020: time data '32.01.2020' does not match format '%d.%m.%Y'
2024-08-04 15:48:22,428:INFO:Окончание анализа даты: 32.01.2020. Время анализа: 0.0000 секунд.
2024-08-04 15:48:22,428:INFO://///////////////
2024-08-04 15:48:26,525:INFO://///////////////
2024-08-04 15:48:26,525:INFO:Начало анализа даты: 01-01-2020
2024-08-04 15:48:26,525:ERROR:Некорректный формат даты (ожидалось 'день.месяц.год'): 01-01-2020
2024-08-04 15:48:26,525:INFO:Окончание анализа даты: 01-01-2020. Время анализа: 0.0000 секунд.
2024-08-04 15:48:26,525:INFO://///////////////
2024-08-04 15:48:30,509:INFO://///////////////
2024-08-04 15:48:30,509:INFO:Начало анализа даты: 01.01.20
2024-08-04 15:48:30,509:ERROR:Некорректная длина строки для даты: 01.01.20
2024-08-04 15:48:30,509:INFO:Окончание анализа даты: 01.01.20. Время анализа: 0.0000 секунд.
2024-08-04 15:48:30,509:INFO://///////////////
2024-08-04 15:48:34,014:INFO://///////////////
2024-08-04 15:48:34,014:INFO:Начало анализа даты: abcd.ef.ghij
2024-08-04 15:48:34,014:ERROR:Некорректная длина строки для даты: abcd.ef.ghij
2024-08-04 15:48:34,014:INFO:Окончание анализа даты: abcd.ef.ghij. Время анализа: 0.0000 секунд.
2024-08-04 15:48:34,014:INFO://///////////////
2024-08-04 15:48:37,086:INFO://///////////////
2024-08-04 15:48:37,086:INFO:Начало анализа даты: 01.13.2020
2024-08-04 15:48:37,086:ERROR:Ошибка проверки даты 01.13.2020: time data '01.13.2020' does not match format '%d.%m.%Y'
2024-08-04 15:48:37,086:INFO:Окончание анализа даты: 01.13.2020. Время анализа: 0.0000 секунд.
2024-08-04 15:48:37,086:INFO://///////////////
2024-08-04 15:48:41,774:INFO://///////////////
2024-08-04 15:48:41,774:INFO:Начало анализа даты: 29.02.2021
2024-08-04 15:48:41,775:ERROR:Ошибка проверки даты 29.02.2021: day is out of range for month
2024-08-04 15:48:41,775:INFO:Окончание анализа даты: 29.02.2021. Время анализа: 0.0005 секунд.
2024-08-04 15:48:41,775:INFO://///////////////
2024-08-04 15:48:46,751:INFO://///////////////
2024-08-04 15:48:46,751:INFO:Начало анализа даты: 00.12.2020
2024-08-04 15:48:46,751:ERROR:Ошибка проверки даты 00.12.2020: time data '00.12.2020' does not match format '%d.%m.%Y'
2024-08-04 15:48:46,751:INFO:Окончание анализа даты: 00.12.2020. Время анализа: 0.0005 секунд.
2024-08-04 15:48:46,751:INFO://///////////////
2024-08-04 15:48:47,503:INFO:Пользователь завершил программу.
2024-08-04 15:48:47,503:INFO:Программа завершена. Обработано 9 дат. Общее время работы программы: 42.7914 секунд.

```

3\*

## Урок 7.1 Функция группового переименования файлов

Напишите функцию группового переименования файлов в папке `test_folder` под названием `rename_files`. Она должна:

- принимать параметр желаемое конечное имя файлов `desired_name`. При переименовании в конце имени добавляется порядковый номер.
- принимать параметр количество цифр в порядковом номере `num_digits`.
- принимать параметр расширение исходного файла `source_ext`. Переименование должно работать только для этих файлов внутри каталога.
- принимать параметр расширение конечного файла `target_ext`.
- принимать диапазон сохраняемого оригинального имени. Например для диапазона [3, 6] берутся буквы с 3 по 6 из исходного имени файла. К ним прибавляется желаемое конечное имя, если оно передано. Далее счётчик файлов и расширение.
- Папка `test_folder` доступна из текущей директории.

```

import os
import logging
from collections import namedtuple

```

```

# Настройка логирования
logging.basicConfig(filename='directory_contents.log', level=logging.DEBUG,
                    format='%(asctime)s: %(levelname)s: %(message)s')

# Определение структуры данных
FileInfo = namedtuple('FileInfo', ['name', 'extension', 'is_directory', 'parent_directory'])

def collect_directory_info(directory_path):
    """Собирает информацию о содержимом директории."""
    if not os.path.isdir(directory_path):
        logging.error(f"Указанный путь не является директорией: {directory_path}")
        print(f"Ошибка: '{directory_path}' не является директорией.")
        return

    logging.info(f"Начинаем обработку директории: {directory_path}")

    try:
        # Проход по всем каталогам и файлам в указанной директории
        for root, dirs, files in os.walk(directory_path):
            logging.info(f"Обрабатываем каталог: {root}")

            for name in dirs:
                # Для каталогов
                dir_path = os.path.join(root, name)
                parent_dir = os.path.basename(root)
                file_info = FileInfo(
                    name=name,
                    extension=None,
                    is_directory=True,
                    parent_directory=parent_dir
                )
                logging.info(f"Каталог: {file_info}")

            for name in files:
                # Для файлов
                file_path = os.path.join(root, name)
                parent_dir = os.path.basename(root)
                name_without_ext, ext = os.path.splitext(name)
                file_info = FileInfo(
                    name=name_without_ext,
                    extension=ext[1:], # убираем точку из расширения

```

```

        is_directory=False,
        parent_directory=parent_dir
    )
    logging.info(f"Файл: {file_info}")

except PermissionError as e:
    logging.error(f"Ошибка доступа к директории '{directory_path}': {e}")
except FileNotFoundError as e:
    logging.error(f"Файл или директория не найдены при обработке '{directory_path}': {e}")
except OSError as e:
    logging.error(f"OS ошибка при обработке '{directory_path}': {e}")
except Exception as e:
    logging.error(f"Неизвестная ошибка при сборе информации из '{directory_path}': {e}")

def main():
    while True:
        # Запрос директорий у пользователя
        directories_input = input("Введите пути до директорий, разделенные пробелом (или нажмите Enter для выхода): ")

        # Проверка на пустую строку
        if not directories_input.strip():
            print("Завершение программы.")
            break

        directories = directories_input.split()

        # Обработка каждой директории
        for directory in directories:
            if os.path.isdir(directory):
                print(f"Обрабатывается директория: {directory}")
                collect_directory_info(directory)
            else:
                # Если директория не существует, записываем ошибку в лог
                logging.error(f"Указанный путь не является директорией: {directory}")
                print(f"Ошибка: '{directory}' не является директорией.")

if __name__ == "__main__":
    main()

```

Файл Правка Формат Вид Справка

```
2024-08-05 13:21:49,339:INFO:Запуск функции rename_files с параметрами: newname, 3, txt, md, [1, 10], D:\task
2024-08-05 13:21:49,340:INFO:Файл newname001 – копия (2).txt переименован в newname001newname001.md
2024-08-05 13:21:49,340:INFO:Файл newname001 – копия (3).txt переименован в newname001newname002.md
2024-08-05 13:21:49,341:INFO:Файл newname001 – копия (4).txt переименован в newname001newname003.md
2024-08-05 13:21:49,341:INFO:Файл newname001 – копия (5).txt переименован в newname001newname004.md
2024-08-05 13:21:49,342:INFO:Файл newname001 – копия.txt переименован в newname001newname005.md
2024-08-05 13:21:49,342:INFO:Файл newname001.txt переименован в newname001newname006.md
2024-08-05 13:21:49,342:INFO:Успешное переименование файлов в директории D:\task
2024-08-05 13:21:49,342:INFO:Время, затраченное на ввод примера: 88.55 секунд.
2024-08-05 13:22:00,036:ERROR:Произошла ошибка: Директория не существует: D:\wrong_task. Пожалуйста, попробуйте снова.
2024-08-05 13:22:00,037:INFO:Время, затраченное на ввод примера: 10.69 секунд.
2024-08-05 13:22:12,797:ERROR:Произошла ошибка: Некорректный ввод данных для количества цифр: invalid literal for int() with base 10: 'three'. Некорректный ввод данных. Пожалуйста, попробуйте снова.
2024-08-05 13:22:12,798:INFO:Время, затраченное на ввод примера: 12.76 секунд.
2024-08-05 13:22:35,944:INFO:Запуск функции rename_files с параметрами: newname, 3, docx, txt, [1, 10], D:\task
2024-08-05 13:22:35,944:ERROR:Ошибка в функции rename_files: Нет файлов с расширением docx в директории D:\task
2024-08-05 13:22:35,944:ERROR:Произошла ошибка: Нет файлов с расширением docx в директории D:\task. Пожалуйста, попробуйте снова.
2024-08-05 13:22:35,945:INFO:Время, затраченное на ввод примера: 23.15 секунд.
2024-08-05 13:23:11,539:INFO:Запуск функции rename_files с параметрами: newname, 3, md, txt, [100, 1000], D:\task
2024-08-05 13:23:11,540:ERROR:Ошибка при переименовании файла newname001newname001.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname001.md
2024-08-05 13:23:11,540:ERROR:Ошибка при переименовании файла newname001newname002.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname002.md
2024-08-05 13:23:11,540:ERROR:Ошибка при переименовании файла newname001newname003.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname003.md
2024-08-05 13:23:11,541:ERROR:Ошибка при переименовании файла newname001newname004.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname004.md
2024-08-05 13:23:11,541:ERROR:Ошибка при переименовании файла newname001newname005.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname005.md
2024-08-05 13:23:11,541:ERROR:Ошибка при переименовании файла newname001newname006.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname006.md
2024-08-05 13:23:11,542:INFO:Не удалось переименовать файлы в директории D:\task
2024-08-05 13:23:11,542:INFO:Время, затраченное на ввод примера: 35.60 секунд.
2024-08-05 13:24:17,962:INFO:Запуск функции rename_files с параметрами: newname, 3, md, md, [100, 1000], D:\task
2024-08-05 13:24:17,962:ERROR:Ошибка при переименовании файла newname001newname001.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname001.md
2024-08-05 13:24:17,963:ERROR:Ошибка при переименовании файла newname001newname002.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname002.md
2024-08-05 13:24:17,963:ERROR:Ошибка при переименовании файла newname001newname003.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname003.md
2024-08-05 13:24:17,963:ERROR:Ошибка при переименовании файла newname001newname004.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname004.md
2024-08-05 13:24:17,964:ERROR:Ошибка при переименовании файла newname001newname005.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname005.md
2024-08-05 13:24:17,964:ERROR:Ошибка при переименовании файла newname001newname006.md: Диапазон [100, 1000] выходит за пределы
длины имени файла: newname001newname006.md

2024-08-05 13:24:17,964:INFO:Не удалось переименовать файлы в директории D:\task
2024-08-05 13:24:17,964:INFO:Время, затраченное на ввод примера: 66.42 секунд.
2024-08-05 13:25:14,160:INFO:Запуск функции rename_files с параметрами: existingname, 1, txt, txt, [1, 10], D:\task
2024-08-05 13:25:14,160:INFO:Файл existingname1.txt переименован в existingnaexistingname1.txt
2024-08-05 13:25:14,161:INFO:Успешное переименование файлов в директории D:\task
2024-08-05 13:25:14,161:INFO:Время, затраченное на ввод примера: 56.20 секунд.
2024-08-05 13:25:17,840:ERROR:Произошла ошибка: Директория не существует: exit. Пожалуйста, попробуйте снова.
2024-08-05 13:25:17,840:INFO:Время, затраченное на ввод примера: 3.68 секунд.
2024-08-05 13:25:22,496:INFO:Программа завершена пользователем. Введено 8 примеров.
2024-08-05 13:25:22,496:INFO:Время, затраченное на ввод примера: 4.66 секунд.
2024-08-05 13:25:22,497:INFO:Пользователь ввел 8 примеров.
```