МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №2.16 по дисциплине основы программной инженерии

Выполнил:
Шальнев Владимир Сергеевич,
2 курс, группа ПИЖ-б-о-20-1,

Проверил:
Доцент кафедры
прикладной математики и
компьютерной безопасности,
Воронкин Р.А.

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2022 г.

Выполнение:

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


import json
import sys
from datetime import date


def get_worker():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))
    # Создать словарь.
    return {
        'name': name,
        'post': post,
        'year': year,
    }


def display_workers(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "No",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)
    # Вывести данные о всех сотрудниках.
    for idx, worker in enumerate(staff, 1):
        print(
            '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                idx,
                worker.get('name', ''),
                worker.get('post', ''),
                worker.get('year', 0)
            )
        )
        print(line)
    else:
        print("Список работников пуст.")
```

```python
def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    # Получить текущую дату.
    today = date.today()
    # Сформировать список работников.
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)
    # Возвратить список выбранных работников.
    return result


def save_workers(file_name, staff):
    """
    Сохранить всех работников в файл JSON.
    """
    # Открыть файл с заданным именем для записи.
    with open(file_name, "w", encoding="utf-8") as fout:
        # Выполнить сериализацию данных в формат JSON.
        # Для поддержки кирилицы установим ensure_ascii=False
        json.dump(staff, fout, ensure_ascii=False, indent=4)


def load_workers(file_name):
    """
    Загрузить всех работников из файла JSON.
    """
    # Открыть файл с заданным именем для чтения.
    with open(file_name, "r", encoding="utf-8") as fin:
        return json.load(fin)


def main():
    """
    Главная функция программы.
    """
    # Список работников.
    workers = []
    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()
        # Выполнить действие в соответствие с командой.
        if command == "exit":
            break
        elif command == "add":
            # Запросить данные о работнике.
            worker = get_worker()
            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))
        elif command == "list":
            # Отобразить всех работников.
            display_workers(workers)
        elif command.startswith("select"):
            # Разбить команду на части для выделения стажа.
            parts = command.split(maxsplit=1)
            # Получить требуемый стаж.
```

```
            period = int(parts[1])
            # Выбрать работников с заданным стажем.
            selected = select_workers(workers, period)
            # Отобразить выбранных работников.
            display_workers(selected)
        elif command.startswith("save "):
            # Разбить команду на части для выделения имени файла.
            parts = command.split(maxsplit=1)
            # Получить имя файла.
            file_name = parts[1]
            # Сохранить данные в файл с заданным именем.
            save_workers(file_name, workers)
        elif command.startswith("load "):
            # Разбить команду на части для выделения имени файла.
            parts = command.split(maxsplit=1)
            # Получить имя файла.
            file_name = parts[1]
            # Сохранить данные в файл с заданным именем.
            workers = load_workers(file_name)
        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            print("help - отобразить справку;")
            print("load - загрузить данные из файла;")
            print("save - сохранить данные в файл;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)


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

Пример 1

```
>>> add
Фамилия и инициалы? Проверка ТС
Должность? Учитель
Год поступления? 2020
>>> add
Фамилия и инициалы? Тест РП
Должность? Студент
Год поступления? 2000
>>> list
+------+--------------------------------+----------------------+----------+
|  No  |            Ф.И.О.              |      Должность       |   Год    |
+------+--------------------------------+----------------------+----------+
|    1 | Проверка ТС                    | Учитель              |     2020 |
+------+--------------------------------+----------------------+----------+
|    2 | Тест РП                        | Студент              |     2000 |
+------+--------------------------------+----------------------+----------+
Список работников пуст.
>>> save data.json
>>> exit

Process finished with exit code 0
```

Результат работы примера 1

```
>>> load data.json
>>> list
+------+--------------------------------+---------------------+----------+
|  No  |            Ф.И.О.             |      Должность      |   Год    |
+------+--------------------------------+---------------------+----------+
|    1 | Проверка ТС                    | Учитель             |     2020 |
+------+--------------------------------+---------------------+----------+
|    2 | Тест РП                        | Студент             |     2000 |
+------+--------------------------------+---------------------+----------+
>>> exit

Process finished with exit code 0
```

Результат работы примера 1

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


import sys
import os
from datetime import datetime
import json
import jsonschema
from jsonschema import validate


def main():
    trains = []
    while True:
        command = get_command()
        if command == 'exit':
            break
        elif command == 'add':
            trains.append(add())
            if len(trains) > 1:
                trains.sort(key=lambda item: item.get('destination', ''))
        elif command == 'list':
            print_list(trains)
        elif command.startswith('select '):
            select(command, trains)
        elif command.startswith("save "):
            parts = command.split(maxsplit=1)
            file_name = parts[1]
            if not save_workers(file_name, trains):
                print("Несоответствующий формат файла", file=sys.stderr)
        elif command.startswith("load "):
            parts = command.split(maxsplit=1)
            file_name = parts[1]
            trains = load_workers(file_name)
        elif command == 'help':
            print_help()
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)


def get_command():
    return input(">>> ").lower()


def add():
    destination = input("Название пункта назначения? ")
    number = int(input("Номер поезда? "))
    time = input("Время отправления ЧЧ:ММ? ")
```

```
    time = datetime.strptime(time, '%H:%M')
    train = {
        'destination': destination,
        'number': number,
        'time': time,
    }
    return train


def print_list(trains):
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 28,
        '-' * 14,
        '-' * 19
    )
    print(line)
    print(
        '| {:^4} | {:^28} | {:^14} | {:^19} |'.format(
            "No",
            "Название пункта назначения",
            "Номер поезда",
            "Время отправления"
        )
    )
    print(line)
    for idx, train in enumerate(trains, 1):
        print(
            '| {:>4} | {:<28} | {:<14} | {:>19} |'.format(
                idx,
                train.get('destination', ''),
                train.get('number', ''),
                train.get('time', 0).strftime("%H:%M")
            )
        )
    print(line)


def print_help():
    print("Список команд:\n")
    print("add - добавить отправление;")
    print("list - вывести список отправлений;")
    print("select <ЧЧ:ММ> - вывод на экран информации о "
          "поездах, отправляющихся после этого времени;")
    print("save <имя файла.json> - сохранить в файл")
    print("load <имя файла.json> - загрузить из файла")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")


def select(command, trains):
    count = 0
    parts = command.split(' ', maxsplit=1)
    time = datetime.strptime(parts[1], '%H:%M')
    for train in trains:
        if train.get("time") > time:
            count += 1
            print(
                '{:>4}: {} {}'.format(
                    count,
                    train.get('destination', ''),
                    train.get("number")
                )
            )
```

```
    if count == 0:
        print("Отправлений позже этого времени нет.")


def save_workers(file_name, staff):
    if file_name.split('.', maxsplit=1)[-1] != "json":
        return False
    try:
        list_of_files = os.listdir(os.path.split(os.getcwd())[0])
        index = list_of_files.index('.gitignore')
        flag = True
    except ValueError:
        flag = False
        print("Файл .gitignore не найден", file=sys.stderr)

    if flag:
        file = f"{os.path.split(os.getcwd())[0]}/.gitignore"
        with open(file, 'a', encoding="utf-8") as git_file:
            git_file.write(file_name)

    for i in staff:
        i['time'] = i['time'].time().strftime("%H:%M")
    with open(file_name, "w", encoding="utf-8") as fout:
        json.dump(staff, fout, ensure_ascii=False, indent=4)
    return True


def validate_json(json_data):
    schema = {
        "type": "array",
        "items": {
            "type": "object",
            "properties": {
                "destination": {"type": "string"},
                "number": {"type": "number"},
                "time": {"type": "string"},
            }
        },
    }

    try:
        validate(instance=json_data, schema=schema)
    except jsonschema.exceptions.ValidationError as err:
        return False
    return True


def load_workers(file_name):
    if file_name.split('.', maxsplit=1)[-1] != "json":
        return False

    with open(file_name, "r", encoding="utf-8") as fin:
        data = []
        try:
            data = json.load(fin)
            flag = validate json(data)
        except Exception as e:
            print("Некоректный файл", file=sys.stderr)
            flag = False
        if flag:
            for i in data:
                i['time'] = datetime.strptime(i['time'], '%H:%M')
            return data
        else:
```

```
            return []


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

Индивидуальная задача с модулем OS

```
>>> load d.json
>>> Файл не существует
load d.txt
Некоректный файл
>>> add
Название пункта назначения? Moscow
Номер поезда? 45
Время отправления ЧЧ:ММ? 23:00
>>> add
Название пункта назначения? Stavropol
Номер поезда? 27
Время отправления ЧЧ:ММ? 12:00
>>> list
+------+-------------------------------+----------------+----------------------+
|  No  |  Название пункта назначения   |  Номер поезда  |  Время отправления   |
+------+-------------------------------+----------------+----------------------+
|    1 | Moscow                        | 45             |                23:00 |
|    2 | Stavropol                     | 27             |                12:00 |
+------+-------------------------------+----------------+----------------------+
>>> save d.txt
>>> Несоответствующий формат файла
sace data.json
Неизвестная команда sace data.json
>>> save data.json
```

Результат работы индивидуальной задачи с модулем OS

```
dmypy.json

# Pyre type checker
.pyre/
data.json
```

Изменение файла .gitignore

```
F:\pythonProject\lab_2_4\venv\Scripts\python.exe F:/pythonProject/lab_2_4/task/task1.py
>>> load data.json
>>> list
+------+-------------------------------+----------------+----------------------+
|  No  |  Название пункта назначения   |  Номер поезда  |  Время отправления   |
+------+-------------------------------+----------------+----------------------+
|    1 | Moscow                        | 45             |                23:00 |
|    2 | Stavropol                     | 27             |                12:00 |
+------+-------------------------------+----------------+----------------------+
>>>
```

Результат работы индивидуальной задачи с модулем OS

Ответы на вопросы:

1. Для чего используется JSON?

Для сериализации сложных структур.

2. Какие типы значений используются в JSON?

запись — это неупорядоченное множество пар ключ:значение, заключённое в фигурные скобки «{ }». Ключ описывается строкой, между ним и значением стоит символ «:». Пары ключ-значение отделяются друг от друга запятыми.

массив (одномерный) — это упорядоченное множество значений. Массив заключается в квадратные скобки «[ ]». Значения разделяются запятыми. Массив может быть пустым, т.е. не содержать ни одного значения. Значения в пределах одного массива могут иметь разныйтип.

число (целое или вещественное).

литералы true (логическое значение «истина»), false (логическое значение «ложь») и null.

строка — это упорядоченное множество из нуля или более символов юникода, заключённое в двойные кавычки. Символы могут быть указаны с использованием escape-последовательностей, начинающихся с обратной косой черты «\» (поддерживаются варианты ', ", \, \/, \t, \n, \r, \f и \b), или записаны шестнадцатеричным кодом в кодировке Unicode в виде \uFFFF.

3. Как организована работа со сложными данными в JSON?

По аналогии со словарями в Python.

4. Самостоятельно ознакомьтесь с форматом данных JSON5? В чем отличие этого формата от формата данных JSON?

JSON5 — предложенное расширение формата json в соответствии с синтаксисом ECMAScript 5, вызванное тем, что json используется не только для общения между программами, но и создаётся/редактируется вручную[6]. Файл JSON5 всегда является корректным кодом ECMAScript 5. JSON5 обратно совместим с JSON. Для некоторых языков программирования уже существуют парсеры json5[7].

Некоторые нововведения:

Поддерживаются как однострочные //, так и многострочные /* */ комментарии.

Записи и списки могут иметь запятую после последнего элемента (удобно при копировании элементов).

Ключи записей могут быть без кавычек, если они являются валидными идентификаторами ECMAScript 5.

Строки могут заключаться как в одинарные, так и в двойные кавычки.

Числа могут быть в шестнадцатеричном виде, начинаться или заканчиваться десятичной точкой, включать Infinity, -Infinity, NaN и -NaN, начинаться со знака +.

5. Какие средства языка программирования Python могут быть использованы для работы с данными в формате JSON5?

https://json5.org/

6. Какие средства предоставляет язык Python для сериализации данных в формате JSON?

Модуль json, который позволяет использовать методы dump и dumps

7. В чем отличие функций json.dump() и json.dumps()?

json.dump() # конвертировать python объект в json и записать в файл

json.dumps() # тоже самое, но в строку

8. Какие средства предоставляет язык Python для десериализации данных из формата JSON?

json.load() # прочитать json из файла и конвертировать в python объект

json.loads() # тоже самое, но из строки с json (s на конце от string/строка)

9. Какие средства необходимо использовать для работы с данными формата JSON, содержащими кирилицу?

ensure_ascii=False

10. Самостоятельно ознакомьтесь со спецификацией JSON Schema? Что такое схема данных? Приведите схему данных для примера 1.

```
schema = {
    "type": "array",
    "items": {
        "type": "object",
        "properties": {
            "name": {"type": "string"},
            "post": {"type": "string"},
            "year": {"type": "number"},
        }
    },
}
```