МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №2.16

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-20-1
Примаков В. Д. « »_____20__г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

ВЫПОЛНЕНИЕ

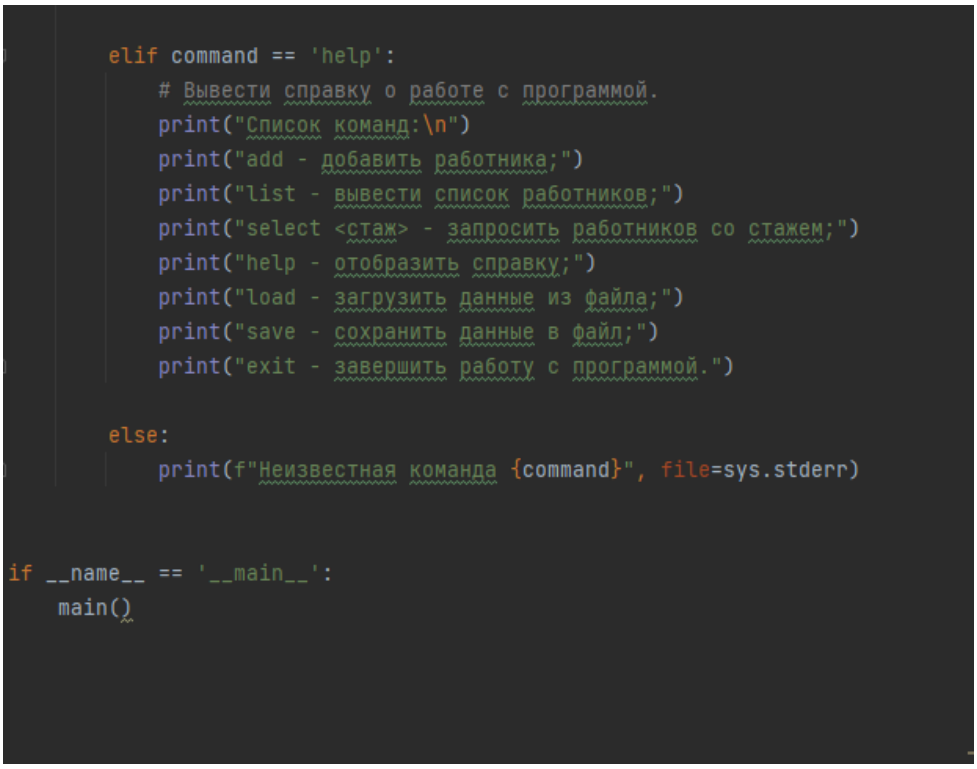
Пример 1:

```python
#!/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(
```

```
                "№",
                "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.")


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
```

```python
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()                                    ⚠ 1  ✓ 165  ^

            # 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()
```

Индивидуальное задание (Вместе со сложным):

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

import sys
import json
import jsonschema
from jsonschema import validate

schema = {
    "type": "object",
    "properties": {
        "price": {"type": "number"},
        "name": {"type": "string"},
        "shop": {"type": "string"}
    },
}


def get_goods():
    """
    Запросить данные о товаре.
    """

    name = input("Название товара: ")
    shop = input("Название магазина: ")
    price = float(input("Стоимость: "))

    # Создать словарь.
    return {
        'name': name,
        'shop': shop,
        'price': price,
    }


def display_goods(goods):
    """
    Отобразить список товаров.
    """
    # Проверить, что список товаров не пуст.
```

```
    if goods:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Название",
                "Магазин",
                "Цена"
            )
        )
        print(line)
        # Вывести данные о всех товарах.
        for idx, good in enumerate(goods, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    good.get('name', ''),
                    good.get('shop', ''),
                    good.get('price', 0)
                )
            )
        print(line)

    else:
        print("Список товаров пуст.")


def select_goods(goods, shop):
    """
    Выбрать товары магазина.
    """
```

```
    # Счетчик записей.                                    ⚠ 8  ✓ 179  ^  v
    count = 0

    # Сформировать список товаров.
    result = []

    for good in goods:
        if shop == good.get('shop', shop):
            count += 1
            result.append(good)

    # Проверка на отсутствие товаров или выбранного магазина.
    if count == 0:
        print("Такого магазина не существует либо нет товаров.")
    else:
        # Возвратить список выбранных товаров.
        return result


def save_goods(file_name, goods):
    """
    Сохранить все магазины 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(goods, fout, ensure_ascii=False, indent=4)


def load_goods(file_name):
    """
    Загрузить все магазины из файла JSON.
    """
    # Открыть файл с заданным именем для чтения.
    with open(file_name, "r", encoding="utf-8") as fin:
        f = json.load(fin)
    err_count = 0
    print("...валидация...")
    for idx, item in enumerate(f):
```

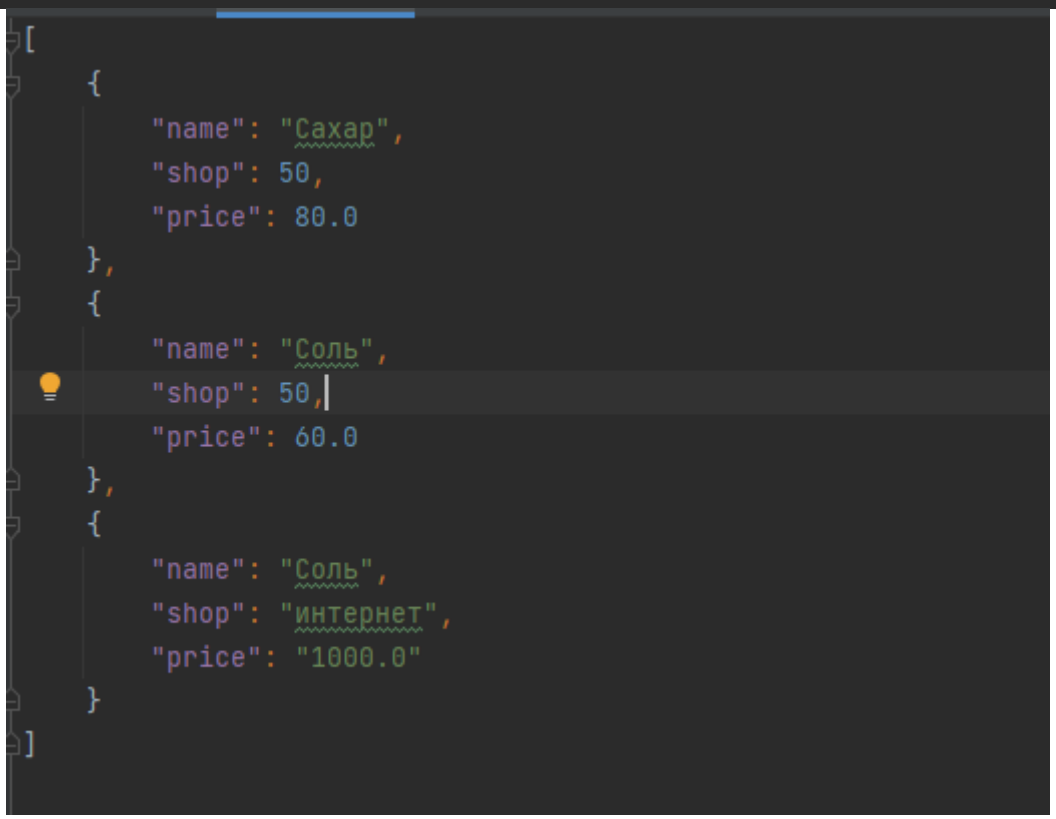
```
        try:                                                    ⚠ 8  ✓ 179  ^
            validate(item, schema)
            sys.stdout.write("Запись {}: ОК\n".format(idx))
        except jsonschema.exceptions.ValidationError as ve:
            sys.stderr.write("Запись {}: ОШИБКА\n".format(idx))
            sys.stderr.write(str(ve) + "\n")
            err_count += 1
    if err_count > 0:
        print("JSON-файл не прошел валидацию.\nФайл не будет загружен.")
    else:
        print("JSON-файл успешно загружен")
        return f


def main():
    """
    Главная функция программы.
    """

    # Список товаров.
    goods = []

    # 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 товаре.
            good = get_goods()

            # Добавить словарь в список.
            goods.append(good)
            # Отсортировать список в случае необходимости.
            if len(goods) > 1:
                goods.sort(key=lambda item: item.get('name', ''))
```

```
        elif command == 'list':
            # Отобразить все товары.
            display_goods(goods)

        elif command.startswith('select '):
            # Разбить команду на части для выделения стажа.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемые товары.
            shop = parts[1]

            # Выбрать товары ммагазина.
            selected = select_goods(goods, shop)
            # Отобразить выбранные товары.
            display_goods(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_goods(file_name, goods)

        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.
            goods = load_goods(file_name)

        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить товар;")
            print("list - вывести список товаров;")
            print("select <имя магазина> - запросить товары магазина;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)


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

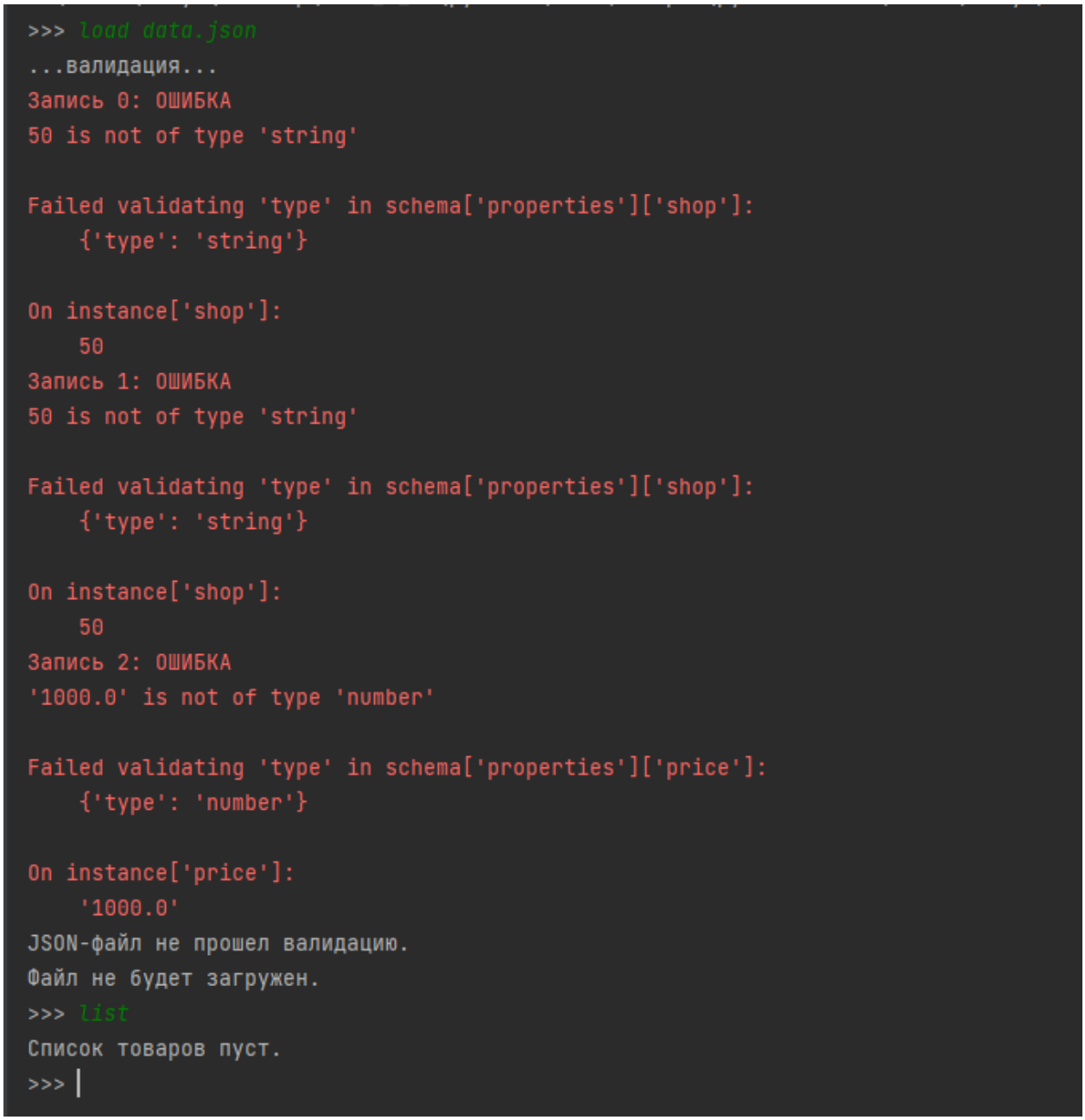
```
>>> help
Список команд:

add - добавить товар;
list - вывести список товаров;
select <имя магазина> - запросить товары магазина;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Название товара: Сахар
Название магазина: пятак
Стоимость: 80
>>> add
Название товара: Соль
Название магазина: пятак
Стоимость: 60
>>> add
Название товара: Соль
Название магазина: интернет
Стоимость: 1000
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |            Название            |       Магазин        |   Цена   |
+------+--------------------------------+----------------------+----------+
|    1 | Сахар                          | пятак                |     80.0 |
|    2 | Соль                           | пятак                |     60.0 |
|    3 | Соль                           | интернет             |   1000.0 |
+------+--------------------------------+----------------------+----------+
>>> select пятак
+------+--------------------------------+----------------------+----------+
|  №   |            Название            |       Магазин        |   Цена   |
+------+--------------------------------+----------------------+----------+
|    1 | Сахар                          | пятак                |     80.0 |
|    2 | Соль                           | пятак                |     60.0 |
+------+--------------------------------+----------------------+----------+
>>> save data.json
>>> exit
```

```
[
    {
        "name": "Сахар",
        "shop": "пятак",
        "price": 80.0
    },
    {
        "name": "Соль",
        "shop": "пятак",
        "price": 60.0
    },
    {
        "name": "Соль",
        "shop": "интернет",
        "price": 1000.0
    }
]
```

```
>>> load data.json
...валидация...
Запись 0: OK
Запись 1: OK
Запись 2: OK
JSON-файл успешно загружен
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |           Название             |       Магазин        |  Цена    |
+------+--------------------------------+----------------------+----------+
|    1 | Сахар                          | пятак                |     80.0 |
|    2 | Соль                           | пятак                |     60.0 |
|    3 | Соль                           | интернет             |   1000.0 |
+------+--------------------------------+----------------------+----------+
>>>
```



```
[
    {
        "name": "Сахар",
        "shop": 50,
        "price": 80.0
    },
    {
        "name": "Соль",
        "shop": 50,
        "price": 60.0
    },
    {
        "name": "Соль",
        "shop": "интернет",
        "price": "1000.0"
    }
]
```



Ссылки на репозитории

GitHub - https://github.com/surai5a/laba_2_15

Ответы на контрольные вопросы

1. Применяется в веб-приложениях как для обмена данными между браузером и сервером (AJAX), так и между серверами (программные HTTP-сопряжения).
2. Значение JSON может быть одним из шести типов данных: строкой, числом, объектом, массивом, булевым значением или null.
3. При работе с большим набором данных предпочтительней использовать запись в многострочном формате с использованием пробелов между парами.
4. 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. Файл JSON5 всегда является корректным кодом ECMAScript 5. JSON5 обратно совместим с JSON. Для некоторых языков программирования уже существуют парсеры json5.

5. Для работы с JSON5 в python можно использовать библиотеку pyjson5.
6. Метод dump библиотеки json для python позволяет конвертировать python-объект в json и записать в файл. (dumps – для записи в строку)
7. Dump – запись в многострочном режиме, dump – запись в строку.
8. Метод load загружает данные json из многострочного режима, loads – из строки.
9. При записи объекта, содержащего кириллицу в файл json нужно использовать параметр ensure_ascii с флагом true. Тогда все символы будут записаны корректно.
10. Схема данных определяет разрешенные типы данных для определенных полей, их количество и т.п.