```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(
        "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.")

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()
```

Рисунок 4.1 – Пример номер 1

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

import pandas as pd
import json
from jsonschema import validate
from pathlib import Path


FILE_NAME = 'json_file.json'
SETTINGS_FILE = 'settings.json'


def make_table():
    with open(FILE_NAME, 'r') as f:
        trains = json.loads(f.read())
        num_lst = []
        end_point_lst = []
        for trn in trains:
            num_lst.append(trn['num'])
            end_point_lst.append(trn['name'])
        data = {'Номер поезда:': num_lst, 'Конечный пункт:': end_point_lst}
        df = pd.DataFrame(data=data)
        print(df)



def add_element():
    name = input('Конечный пункт: ')
    num = input('Номер поезда: ')
    tm = input('Время отправления: ')
    trains = {}
    trains['name'] = name
    trains['num'] = int(num)
    trains['tm'] = tm
```

```
    schema = {
        "type": "object",
        "properties": {
            "description": {"type": "string"},
            "status": {"type": "boolean"},
            "value_a": {"type": "number"},
            "value_b": {"type": "number"},
        },
    }
    validate(instance=trains, schema=schema)
    with open(FILE_NAME, 'a') as f:
        f.write(json.dumps(trains) + '\n')


def find_train(num):
    with open('json_file.json', 'r') as f:
        trains = f.readlines()
        for dcts in trains:
            dcts = json.loads(dcts)
            if dcts['num'] == int(num):
                print(
                    f'Конечный пункт: {dcts["name"]} \n'
                    f'Номер поезда: {dcts["num"]} \n'
                    f'Время отправления: {(dcts["tm"])}'
                )
                return
        print('Поезда с таким номером нет')


if __name__ == '__main__':
    print('LOADING...')
    with open(SETTINGS_FILE, 'r') as f:
        settings = json.loads(f.read())
        if settings['gitignore'] == False:
            path = Path(__file__).resolve()
            print(path.parents[1])
            par_path = path.parents[1]
            with open(str(par_path) + '\\.gitignore', 'a') as gig:
                gig.write('\n' + '*.json' + '\n')
    with open(SETTINGS_FILE, 'w') as f:
        f.write(json.dumps({'gitignore': True}))

    print('Hello!')

    flag = True
    while flag:
        print('1. Добавить новый поезд')
        print('2. Вывести информацию о поезде')
        print('3.Выход из программы')
        com = int(input('введите номер команды: '))
        if com == 1:
            add_element()
        elif com == 2:
            train_num = input('Введите номер поезда: ')
            find_train(train_num)
        elif com == 3:
            flag = False
```

Рисунок 4.2 – Код программы индивидуального задания

Контрольные вопросы:

1. Для чего используется JSON?

Формат JSON используется для упорядоченного хранения данных в процессе их обмена между веб-браузером или клиентской частью приложения и сервером или между разными серверами.

2. Какие типы значений используются в JSON?

- строка;
- число;
- логический;
- null;
- объект;
- массив.

3. Как организована работа со сложными данными в JSON?

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

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

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

Модуль json

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

json.dump() - сериализует obj как форматированный JSON поток в fp.

json.dumps() - сериализует obj в строку JSON-формата.

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

Модуль json

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

Необходимо использовать средства позволяющие декодировать в подходящую для кодировки кирилицу

10. Самостоятельно ознакомьтесь со спецификацией JSON Schema? Что такое схема данных?

Приведите схему данных для примера 1.