Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

**Разработка приложений с интерфейсом командной строки (CLI) в Python3**

|  |
| --- |
| Выполнил:  Боженко Александр Иванович  2 курс, группа ИТС-б-о-21-1,  11.03.02 «Инфокоммуникационные технологии и системы связи», направленность (профиль)  «Инфокоммуникационные системы и сети», очная форма обучения  (подпись) |
| Руководитель практики:  Воронкин Р.А, канд. техн. наук, доцент кафедры инфокоммуникаций  (подпись) |

Отчет защищен с оценкой Дата защиты

Ставрополь, 2022 г.

**Цель работы**: приобретение навыков по работе с данными формата

JSON с помощью языка программирования Python версии 3.x. Ссылка на репозиторий https://github.com/danilusikov0913/YPlr6

**Ход работы:**

**1. Пример**

Для примера 1 лабораторной работы 2.8 добавьте возможность сохранения списка в файл формата JSON и чтения данных из файла JSON.

Решение: введем следующие команды для работы с файлом формата JSON в интерактивном режиме:

* load - загрузить данные из файла, имя файла должно отделяться от команды load пробелом. Например: load data.json
* save - сохранить сделанные изменения в файл, имя файла должно отделяться от команды save пробелом. Например: save data.json Напишем программу для решения поставленной задачи.

Код задания 1

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*-

import argparse

import json

import os.path

import sys

def add\_shop(list\_race, name, namber, time):

    """

    Добавить данные магазина.

    """

    list\_race.append(

        {

            "name": name,

            "namber": namber,

            "time": time

        }

    )

    return list\_race

def display\_shop(list\_race):

    """

    Отобразить список.

    """

    if list\_race:

        # Заголовок таблицы.

        line = '+-{}-+-{}-+-{}-+-{}-+'.format(

            '-' \* 6,

            '-' \* 20,

            '-' \* 30,

            '-' \* 20

        )

        print(line)

        print(

            '| {:^6} | {:^20} | {:^30} | {:^20} |'.format(

                "No",

                "пункт назначения",

                "номер",

                "время"

            )

        )

        print(line)

        for idx, listrace in enumerate(list\_race, 1):

            print(

                '| {:>6} | {:<20} | {:<30} | {:>20} |'.format(

                    idx,

                    listrace.get('name', ''),

                    listrace.get('namber', ''),

                    listrace.get('time', 0)

                )

            )

        print(line)

    else:

        print("Список рейсов пуст.")

def select\_product(list\_race, race\_sear):

    """

    Выбрать.

    """

    search\_race = []

    for race\_sear\_itme in list\_race:

        if race\_sear == race\_sear\_itme['name']:

            search\_race.append(race\_sear\_itme)

    return search\_race

def save\_race(file\_name, list\_race):

    """

    Сохранить все в JSON.

    """

    with open(file\_name, "w", encoding="utf-8") as fout:

        json.dump(list\_race, fout, ensure\_ascii=False, indent=4)

def load\_list\_race(file\_name):

    """

    Загрузить все из файла JSON.

    """

    with open(file\_name, "r", encoding="utf-8") as fin:

        return json.load(fin)

def main(command\_line=None):

    file\_parser = argparse.ArgumentParser(add\_help=False)

    file\_parser.add\_argument(

        "filename",

        action="store",

        help="The data file name"

    )

    parser = argparse.ArgumentParser("races")

    parser.add\_argument(

        "--version",

        action="version",

        version="%(prog)s 0.1.0"

    )

    subparsers = parser.add\_subparsers(dest="command")

    add = subparsers.add\_parser(

        "add",

        parents=[file\_parser],

        help="Add a new race"

    )

    add.add\_argument(

        "-nm",

        "--name",

        action="store",

        required=True,

        help="The race's name"

    )

    add.add\_argument(

        "-nb",

        "--namber",

        action="store",

        help="The namber"

    )

    add.add\_argument(

        "-t",

        "--time",

        action="store",

        type=int,

        required=True,

        help="time"

    )

    \_ = subparsers.add\_parser(

        "display",

        parents=[file\_parser],

        help="Display all races"

    )

    select = subparsers.add\_parser(

        "select",

        parents=[file\_parser],

        help="Select the product"

    )

    select.add\_argument(

        "-SS",

        "--name\_sear",

        action="store",

        type=str,

        required=True,

        help="The name race"

    )

    args = parser.parse\_args(command\_line)

    is\_dirty = False

    if os.path.exists(args.filename):

        race = load\_list\_race(args.filename)

    else:

        race = []

    if args.command == "add":

        race = add\_shop(

            race,

            args.name,

            args.namber,

            args.time

        )

        is\_dirty = True

    elif args.command == "display":

        display\_shop(race)

    elif args.command == "select":

        selected = select\_product(race, args.race\_sear)

        display\_shop(selected)

    if is\_dirty:

        save\_race(args.filename, race)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

результат программы

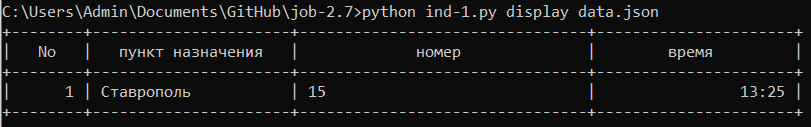


Рис 1.

**Контрольные вопросы:**

* 1. Для чего используется JSON?

JSON представляет собой хорошую альтернативу XML и требует куда меньше форматирования контента. Это информативное руководство поможет вам быстрее разобраться с данными, которые вы можете использовать с JSON и основной структурой с синтаксисом этого же формата.

* 1. Какие типы значений используются в JSON?

Запись, массив, число, литералы, строка

* 1. Как организована работа со сложными данными в JSON?
  2. 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. Файл JSON5 всегда является корректным кодом ECMAScript 5. JSON5 обратно совместим с JSON

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

|  |  |
| --- | --- |
| [JSON5](https://json5.org/) расширяет формат обмена данными [JSON,](http://www.json.org/) чтобы сделать его немного более удобным в качестве языка конфигурации: | |
|  | * Комментарии в стиле JavaScript (как однострочные, так и многострочные) являются законными. * Ключи объектов могут быть без кавычек, если они являются законными идентификаторами ECMAScript * Объекты и массивы могут заканчиваться запятыми. * Строки могут заключаться в одинарные кавычки, и допускаются многострочные строковые литералы. |

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

Модуль **json** предоставляет удобный метод dump() для записи данных в файл. Существует также метод dumps() для записи данных в обычную строку. Типы данных Python кодируются в формат JSON в соответствии с интуитивно понятными правилами преобразования

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

dump отличается от dumps тем, что dump записывает объект Python в

файл JSON, а dumps сериализует объект Python и хранит его в виде строки.

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

В модуле **json** определены методы load() и loads(), предназначенные для преобразования кодированных в формате JSON данных в объекты Python.

Подобно операции *сериализации*, также существует таблица преобразования типов, определяющая правила для обратного *декодирования* данных.

* 1. Какие средства необходимо использовать для работы с данными формата JSON, содержащими кириллицу?

Параметр ensure\_ascii

**Вывод:** в ходе лабораторной работы приобретены навыки по работе с данными формата JSON с помощью языка программирования Python версии

3.x.