

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

**Отчет по лабораторной работе № 2.17**

**«Работа с данными формата JSON в языке  
Python»**

**по дисциплине «Основы программной инженерии»**

Выполнила:  
Коновалова Виктория  
Николаевна, 2 курс, группа  
ПИЖ-б-о-21-1, Проверил:  
Доцент кафедры инфокоммуникаций,  
Воронкин Р.А.

Ставрополь, 2023 г.

## Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.

Проработайте примеры лабораторной работы. Создайте для них отдельные модули языка Python. Зафиксируйте изменения в репозитории.

```
primer.py x primer1.json x README.md x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse
5  import json
6  import os.path
7  import sys
8  from datetime import date
9
10  1 usage
11  def add_worker(staff, name, post, year):
12      """
13      Добавить данные о работнике.
14      """
15      staff.append(
16          {
17              "name": name,
18              "post": post,
19              "year": year
```

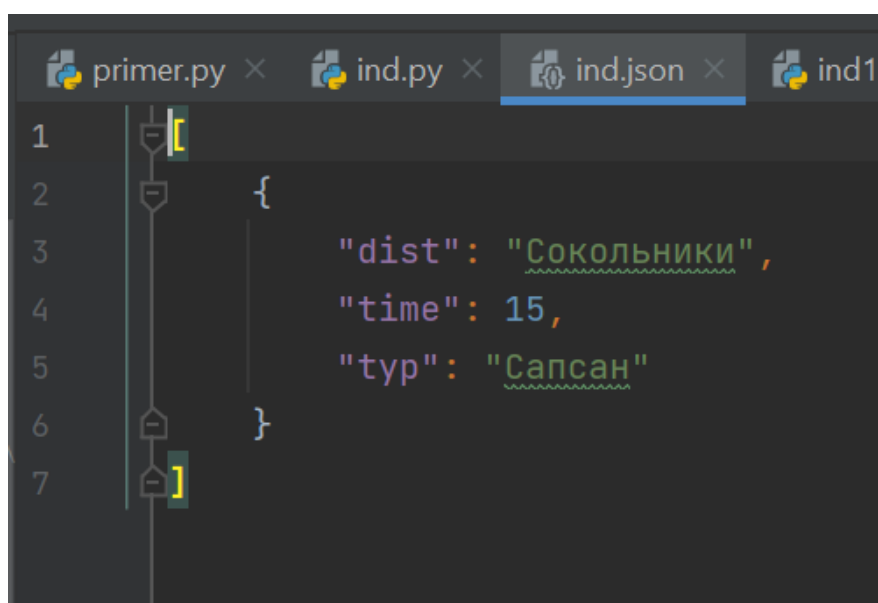
```
primer.py x primer1.json x README.md x
1  [
2      {
3          "name": "Иван Иванов",
4          "post": "Сварщик",
5          "year": 2011
6      },
7      {
8          "name": "Сидоров Сидор",
9          "post": "Главный инженер",
10         "year": 2012
11     }
12 ]
```

## Индивидуальные задания

Для своего варианта лабораторной работы 2.16 необходимо дополнительно реализовать интерфейс командной строки (CLI).



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse
5  import json
6  import os
7
8
9  def get_train(staff, dist, time, typ):
10     """
11     Запросить данные о поезде.
12     """
13
14     staff.append({
15         "dist": dist,
16         "time": time,
17         "typ": typ,
18     })
19
20 get_train()
```



```
1  [
2      {
3          "dist": "Сокольники",
4          "time": 15,
5          "typ": "Сапсан"
6      }
7  ]
```

## Задание повышенной сложности

Самостоятельно изучите работу с пакетом click для построения интерфейса командной строки (CLI). Для своего варианта лабораторной работы 2.16 необходимо реализовать интерфейс командной строки с использованием пакета click

```
ind_hard.py x ind_hard.json x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import argparse
5  import json
6  import os
7  import click
8
9
10  1 usage
11  def get_train(staff, dist, time, typ, file_name):
12      """
13      Запросить данные о поезде.
14      """
15      staff.append({
16          "dist": dist,
17          "time": time,
```

```
ind_hard.py x ind_hard.json x
1  [
2      {
3          "dist": "Черкизово",
4          "time": "Ласточка",
5          "typ": "12:30"
6      }
7  ]
```

No	Пункт назначения	время поезда	Тип поезда
1	Черкизово	Ласточка	12:30

## **Вопросы для защиты работы**

### **1. В чем отличие терминала и консоли?**

Таким образом, основное отличие между консолью и терминалом заключается в том, что консоль является аппаратным устройством, а терминал - программой. Консоль используется для ввода и вывода информации в реальном времени, в то время как терминал обычно используется для управления системой, запуска приложений, выполнения команд и т.д.

### **2. Что такое консольное приложение?**

Консольное приложение - это программное обеспечение, которое запускается в командной строке (консоли) операционной системы и работает в текстовом режиме. Оно не имеет графического интерфейса и взаимодействует с пользователем через ввод и вывод текстовых данных в командной строке.

### **3. Какие существуют средства языка программирования Python для построения приложений командной строки?**

Модуль `sys`, Модуль `getopt`, Модуль `argparse`, Модуль `click`

### **4. Какие особенности построение CLI с использованием модуля `sys`?**

Это базовый модуль, который с самого начала поставлялся с Python. Он использует подход, очень похожий на библиотеку C, с использованием `argc` и `argv` для доступа к аргументам. Модуль `sys` реализует аргументы командной строки в простой структуре списка с именем `sys.argv`.

Каждый элемент списка представляет собой единственный аргумент. Первый элемент в списке `sys.argv[0]` — это имя скрипта Python. Остальные элементы списка, от `sys.argv[1]` до `sys.argv[n]`, являются аргументами командной строки с 2 по n. В качестве разделителя между аргументами используется пробел. Значения аргументов, содержащие пробел, должны быть заключены в кавычки, чтобы их правильно проанализировал `sys`. Эквивалент `argc` — это просто количество элементов в списке. Чтобы

получить это значение,  
используйте оператор `len()` . Позже мы покажем это на примере кода.

## **5. Какие особенности построение CLI с использованием модуля `getopt`?**

Модуль `getopt` в Python предназначен для парсинга аргументов командной строки, переданных при запуске скрипта. Он предоставляет набор функций для анализа опций и их значений, которые передаются скрипту через командную строку.

Некоторые особенности построения CLI с использованием модуля `getopt`:

Определение опций и их значений. Модуль `getopt` предоставляет функцию `getopt()`, которая принимает три аргумента: список аргументов командной строки, строку, определяющую короткие опции, и список строк, определяющих длинные опции. Эта функция возвращает кортеж, содержащий два элемента: список опций и список аргументов.

## **6. Какие особенности построение CLI с использованием модуля `argparse`?**

Модуль `argparse` является более современным и удобным способом парсинга аргументов командной строки, чем модуль `getopt`. Он имеет следующие особенности:

Поддержка позиционных и именованных аргументов: `argparse` позволяет определять как позиционные аргументы, так и именованные (опциональные) аргументы, что упрощает интерфейс командной строки для пользователя.