

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ ФГАОУ ВО  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**Дисциплина «Языки программирования»**

**Отчет по практической работе № 2.17**

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

Выполнил: студент группы ИТС-б-о-22-1  
Наумов Никита Викторович

---

(подпись)

Проверил: к.т.н., доцент  
Кафедры инфокоммуникаций  
Воронкин Р.А.

---

(подпись)

Ставрополь, 2023

**Цель работы:** приобретение построения приложений с интерфейсом командной строки с помощью языка программирования Python версии 3.x.

### Практическая часть:

#### 1. Проработал примеры лабораторной работы.

```
G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python primer1.py add data.json
usage: workers add [-h] -n NAME [-p POST] -y YEAR filename
workers add: error: the following arguments are required: -n/--name, -y/--year

G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python primer1.py add data.json --name="Пирожков Артур" --post="Слесарь" --year=2015

G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python primer1.py add data.json --name="Данилевский Игорь" --post="Строитель" --year=1999

G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python primer1.py display data.json
```

No	Ф.И.О.	Должность	Год
1	Пирожков Артур	Слесарь	2015
2	Данилевский Игорь	Строитель	1999

```
G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python primer1.py select data.json --period=10
```

No	Ф.И.О.	Должность	Год
1	Данилевский Игорь	Строитель	1999

Рисунок 5. Результат работы примера

2. Задание: для своего варианта лабораторной работы 2.16 необходимо дополнительно реализовать интерфейс командной строки (CLI).

```
G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python individual1.py add ind1.json --no=1 --time="21:50:30" --name="Stavropol" --no=2 --time="21:50:30"

G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python individual1.py add ind1.json --no=1 --time="21:00:00" --name="Moscow" --no=2 --time="21:50:30" --name="Stavropol" --no=3 --time="12:50:30" --name="Chelaba"

G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python individual1.py display ind1.json
```

No	Название	Время
1	Moscow	21:00:00
2	Stavropol	21:50:30
3	Chelaba	12:50:30

Рисунок 2. Результат выполнения индивидуального задания

```
G:\Лабы\1 семестр 2 курса\Языки программирования\лаб_раб 7\Lab2_17-main\prog>python individual1.py select ind1.json --no=1 --time="21:00:00"
Список поездов пуст.
```

Рисунок 3. Результат поиска поезда по номеру.

3. 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.

```
PS C:\Users\super\Desktop\Dina\ВУЗ\программирование на python\Lab2_17\prog> python individual2.py -c display ind1.json
```

No	Название	Время
1	One	12:21:00
2	Two	12:21:00
3	Three	21:21:00

```
PS C:\Users\super\Desktop\Dina\ВУЗ\программирование на python\Lab2_17\prog>
```

Рисунок 7. Результат выполнения индивидуального задания повышенной сложности

### Контрольные вопросы:

1. В чем отличие терминала и консоли?

Терминал (от лат. terminus — граница) — устройство или ПО, выступающее посредником между человеком и вычислительной системой. Обычно данный термин используется, когда точка доступа к системе вынесена в отдельное физическое устройство и предоставляет свой пользовательский интерфейс на основе внутреннего интерфейса (например, сетевых протоколов).

Консоль console — исторически реализация терминала с клавиатурой и текстовым дисплеем. В настоящее время это слово часто используется как синоним сеанса работы или окна оболочки командной строки. В том же смысле иногда применяется и слово “терминал”.

2. Что такое консольное приложение?

Консольное приложение console application — вид ПО, разработанный с расчётом на работу внутри оболочки командной строки, т.е. опирающийся на текстовый ввод-вывод.

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

Python 3 поддерживает несколько различных способов обработки аргументов командной строки.

Встроенный способ – использовать модуль sys . С точки зрения имен и использования, он имеет

прямое отношение к библиотеке C (libc). Второй способ – это модуль getopt , который

обрабатывает как короткие, так и длинные параметры, включая оценку значений параметров.

Кроме того, существуют два других общих метода. Это модуль `argparse`, производный от

модуля `optparse`, доступного до Python 2.7. Другой метод – использование модуля `docopt`,

доступного на GitHub. У каждого из этих способов есть свои плюсы и минусы, поэтому стоит

оценить каждый, чтобы увидеть, какой из них лучше всего соответствует вашим потребностям.

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`?

Как вы могли заметить ранее, модуль `sys` разбивает строку командной строки только на отдельные фасы. Модуль `getopt` в Python идет немного дальше и расширяет разделение входной строки проверкой параметров. Основанный на функции C `getopt`, он позволяет использовать как короткие, так и длинные варианты, включая присвоение значений.

6. Какие особенности построение CLI с использованием модуля

argparse ?

Для начала рассмотрим, что интересного предлагает argparse :

- анализ аргументов `sys.argv` ;
- конвертирование строковых аргументов в объекты Вашей программы и работа с ними;
- форматирование и вывод информативных подсказок.

Одним из аргументов противников включения argparse в Python был довод о том, что в стандартных модулях и без этого содержится две библиотеки для семантической обработки (парсинга) параметров командной строки. Однако, как заявляют разработчики argparse , библиотеки `getopt` и `optparse` уступают argparse по нескольким причинам:

- обладая всей полнотой действий с обычными параметрами командной строки, они не умеют обрабатывать позиционные аргументы (positional arguments). Позиционные аргументы — это аргументы, влияющие на работу программы, в зависимости от порядка, в котором они в эту программу передаются. Простейший пример — программа `cp`, имеющая минимум 2 таких аргумента («`cp source destination`»).
- argparse дает на выходе более качественные сообщения о подсказке при минимуме затрат (в этом плане при работе с `optparse` часто можно наблюдать некоторую избыточность кода);
- argparse дает возможность программисту устанавливать для себя, какие символы являются параметрами, а какие нет. В отличие от него, `optparse` считает опции с синтаксисом наподобие `"-pf, -file, +rgb, /f` и т.п. «внутренне противоречивыми» и «не поддерживается `optpars` 'ом и никогда не будет»;
- argparse даст Вам возможность использовать несколько значений переменных у одного аргумента командной строки (nargs);
- argparse поддерживает субкоманды (subcommands). Это когда основной парсер отправляет к другому (субпарсеру), в зависимости от аргументов на входе.

**Вывод:** в результате выполнения работы были приобретены знания о построении приложений с интерфейсом командной строки с помощью языка программирования Python версии 3.x.