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 №2.8
## дисциплины «Основы кроссплатформенного программирования»

Выполнила:
Иващенко Олеся Игорьевна
1 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., доцент кафедры инфокоммуникаций

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2022 г.

**Тема:** Работа с функциями в языке Python

**Цель:** приобретение навыков по работе с функциями при написании программ с помощью языка программирования Python версии 3.x.

## Ход работы:

## Вариант №10

## Индивидуальные задания

### Задание 1

Решить индивидуальное задание лабораторной работы 2.6, оформив каждую команду в виде отдельной функции.

my1.py

```
# Название  начального пункта маршрута
# Название конечного пункта
# Номер маршрута (фильтр)

list_route = []
spisok_new = []
list_route = []


def table():
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 15,
            '-' * 30,
            '-' * 20
        )
    return line

def table_name():
    line = '| {:^4} | {:^15} | {:^30} | {:^20} | '.format(
            "№",
            "Начало маршрута",
            "Конец маршрута",
            "№ Маршрута"
            )
    return line

def table_znach(list_routee):
    post = []
    for idx, spisok_new in enumerate(list_routee, 1):
        post.append('| {:>4} | {:<15} | {:<30} | {:<20} | '.format(
```

Рисунок 1. Задание 1 (1)


Рисунок 2. Задание 1 (2)

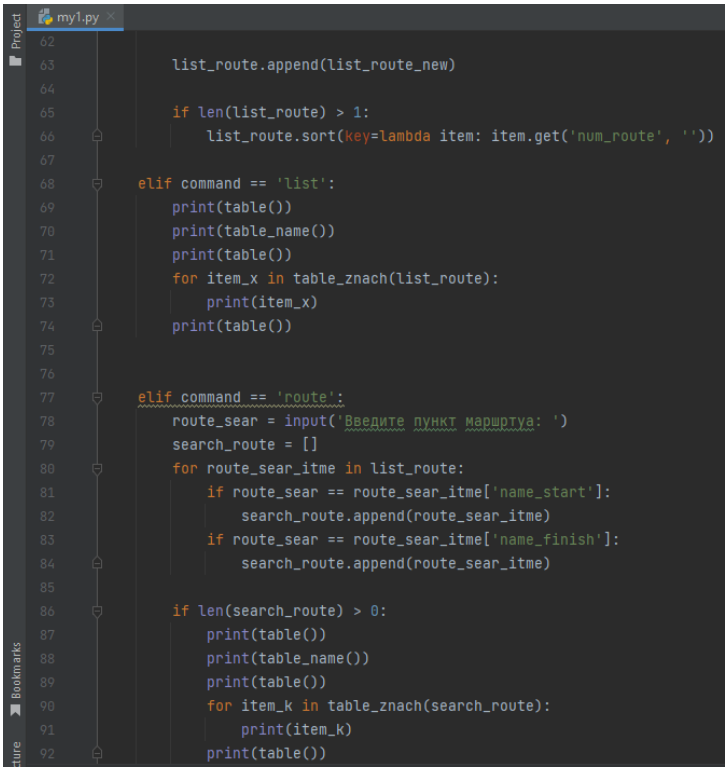


Рисунок 3. Задание 1 (3)

```
        else:
            print('Таких маршрутов не найдено', file=sys.stderr)


    elif command == 'help':
        print('Список команд:\n')
        print('add - добавить маршрут.')
        print('list - вывести список маршрутов.')
        print('route <Пункуты маршрутов> - запросить Начало или конечные пункты маршрутов.')
        print('help - Справочник.')
        print('exit - Завершить работу программы.')
    else:
        print(f'Команда <{command}> не существует.', file=sys.stderr)
        print('Введите <help> для просмотра доступных команд')
```

Рисунок 4. Задание 1 (4)

```
Run:  my1
C:\Users\User\PycharmProjects\pythonProject4\venv\Scripts\python.exe C:/Users/User/Pychar
>>> add
Начало маршрута: Владимира Нургалиева
Конец маршрута: ж/к Олимпийский
Номер маршрута: 48
>>> list
+------+-----------------+------------------------------+----------------------+
|  №   | Начало маршрута |         Конец маршрута       |      № Маршрута      |
+------+-----------------+------------------------------+----------------------+
|    1 | Владимира Нургалиева | ж/к Олимпийский              | 48                   |
+------+-----------------+------------------------------+----------------------+
>>> route
Введите пункт маршртуа: ж/к Олимпийский
+------+-----------------+------------------------------+----------------------+
|  №   | Начало маршрута |         Конец маршрута       |      № Маршрута      |
+------+-----------------+------------------------------+----------------------+
|    1 | Владимира Нургалиева | ж/к Олимпийский              | 48                   |
+------+-----------------+------------------------------+----------------------+
>>> help
Список команд:

add - добавить маршрут.
list - вывести список маршрутов.
route <Пункуты маршрутов> - запросить Начало или конечные пункты маршрутов.
help - Справочник.
exit - Завершить работу программы.
>>> exit

Process finished with exit code 0
```

Рисунок 5. Задание 1 (5)

## Ответы на контрольные вопросы:

1. Каково назначение функций в языке программирования Python?

Функция в программировании представляет собой обособленный участок кода, который можно вызывать, обратившись к нему по имени, которым он был назван. При вызове происходит выполнение команд тела функции.

Функции можно сравнить с небольшими программками, которые сами по себе, т. е. автономно, не исполняются, а встраиваются в обычную программу.

2. Каково назначение операторов def и return?

Оператор def, выполняемый внутри определения функции, определяет локальную функцию, которая может быть возвращена или передана. Свободные переменные, используемые во вложенной функции, могут обращаться к локальным переменным функции, содержащей def.

Оператор return возвращает значение из функции. return без аргумента возвращает None. Функции, у которых return не определен, также возвращает None.

3. Каково назначение локальных и глобальных переменных при написании функций в Python?

В Python переменная, объявленная вне функции или в глобальной области видимости, называется глобальной переменной. К глобальной переменной можно получить доступ как внутри, так и вне функции.

Переменная, объявленная внутри тела функции или в локальной области видимости, называется локальной переменной.

4. Как вернуть несколько значений из функции Python?

В Питоне позволительно возвращать из функции несколько объектов, перечислив их через запятую после команды return.

5. Какие существуют способы передачи значений в функцию?

По умолчанию аргументы могут передаваться в функцию Python либо по положению, либо явно по ключевому слову. Для производительности и удобочитаемости имеет смысл ограничить способ передачи аргументов. где символы / и * являются НЕ обязательными. Эти символы указывают тип аргумента в зависимости от того, как они могут быть переданы в функцию:

только по позиции, по позиции или по ключевому слову только по ключевому слову.

6. Как задать значение аргументов функции по умолчанию?

Значения параметров по умолчанию создаются при определении функции, а НЕ каждый раз, когда она вызывается в коде программы. Это означает, что эти выражение вычисляется один раз, и что для каждого вызова используется одно и то же предварительно вычисленное значение. Если функция изменяет объект (например, путем добавления элемента в список, словарь), значение по умолчанию фактически изменяется.

7. Каково назначение lambda-выражений в языке Python?

Python поддерживает интересный синтаксис, позволяющий определять небольшие однострочные функции на лету. Позаимствованные из Lisp, так называемые lambda-функции могут быть использованы везде, где требуется функция. lambda – это выражение, а не инструкция. По этой причине ключевое слово lambda может появляться там, где синтаксис языка Python не позволяет использовать инструкцию def , – внутри литералов или в вызовах функций, например.

8. Как осуществляется документирование кода согласно PEP257?

PEP 257 описывает соглашения, связанные со строками документации python, рассказывает о том, как нужно документировать python код. Цель этого PEP - стандартизировать структуру строк документации: что они должны в себя включать, и как это написать (не касаясь вопроса синтаксиса строк документации). Этот PEP описывает соглашения, а не правила или синтаксис. При нарушении этих соглашений, самое худшее, чего можно ожидать – некоторых неодобрительных взглядов. Но некоторые программы (например, docutils), знают о соглашениях, поэтому следование им даст вам лучшие результаты. Строки документации - строковые литералы, которые являются первым оператором в модуле, функции, классе или определении метода.

9. В чем особенность однострочных и многострочных форм строк документации?

```
def kos_root():
    """Return the pathname of the KOS root directory."""
    global _kos_root
    if _kos_root: return _kos_root
```

Используйте тройные кавычки, даже если документация умещается на одной строке. Потом будет проще её дополнить.

Однострочная строка документации не должна быть "подписью" параметров функции / метода (которые могут быть получены с помощью интроспекции). Не делайте:

```
def function(a, b):
    """function(a, b) -> list"""
```

Этот тип строк документации подходит только для C функций (таких, как встроенные модули), где интроспекция не представляется возможной. Тем не менее, возвращаемое значение не может быть определено путем интроспекции. Предпочтительный вариант для такой строки документации будет что-то вроде:

```
def function(a, b):
    """Do X and return a list."""
```

Рисунок 1. Однострочные

Многострочные строки документации состоят из однострочной строки документации с последующей пустой строкой, а затем более подробным описанием. Первая строка может быть использована автоматическими средствами индексации, поэтому важно, чтобы она находилась на одной строке и была отделена от остальной документации пустой строкой. Первая строка может быть на той же строке, где и открывающие кавычки, или на следующей строке. Вся документация должна иметь такой же отступ, как кавычки на первой строке (см. пример ниже).

Рисунок 2. Многострочные

**Вывод:** В ходе выполнения лабораторной работы приобретены навыки по работе с функциями при написании программ с помощью языка программирования Python версии 3.x.