

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Отчет по лабораторной работе №2.8  
Работа с функциями в языке Python  
по дисциплине «Технологии программирования»**

Выполнил студент группы ИВТ-б-о-20-1

Ищенко Т.С. « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверила Воронкин Р.А. \_\_\_\_\_

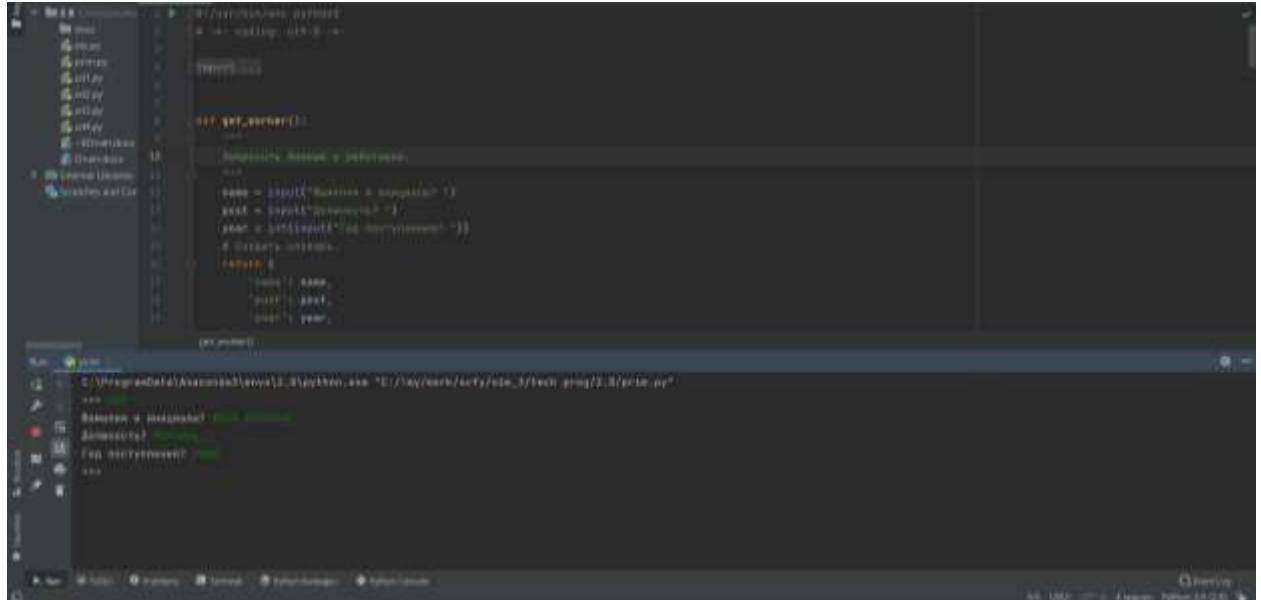
(подпись)

Ставрополь 2021

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

Ход работы:

### 1. Произвёл выполнение примера



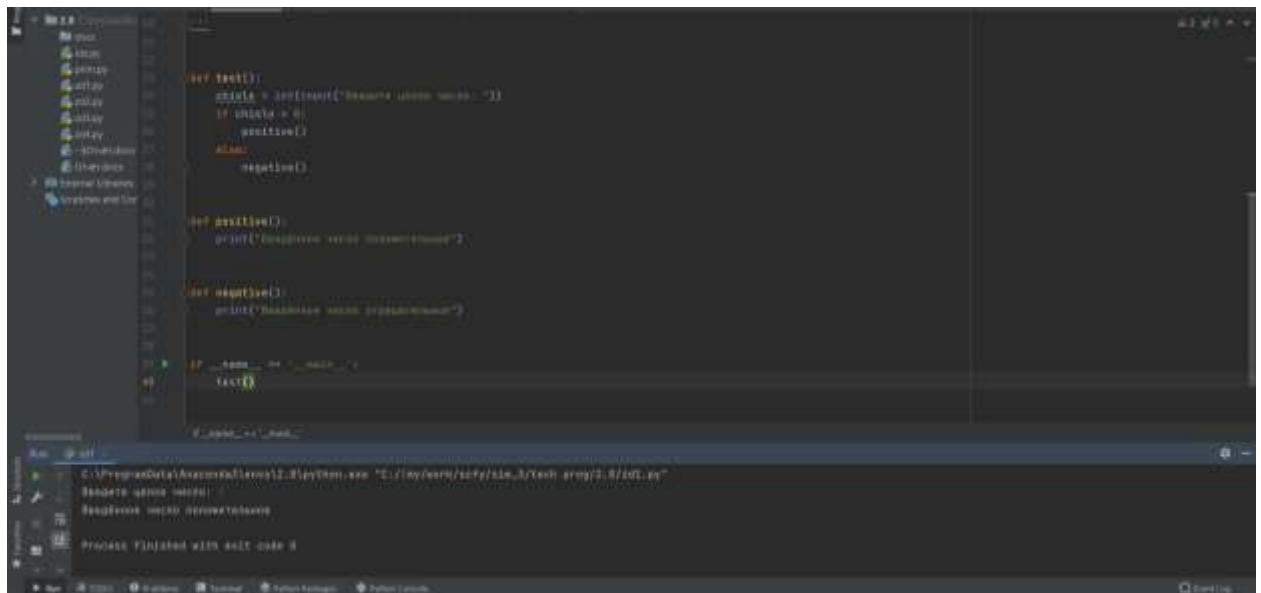
```
def get_worker():  
    """Возвращает данные о работнике."""  
    name = input("Введите имя работника: ")  
    sex = input("Введите пол: ")  
    age = input("Введите возраст работника: ")  
    # Создаём словарь.  
    return {  
        "name": name,  
        "sex": sex,  
        "age": age,  
    }  
get_worker()
```

Output:

```
Введите имя работника: Иван Иванович  
Введите пол: мужской  
Введите возраст работника: 35  
{'name': 'Иван Иванович', 'sex': 'мужской', 'age': 35}
```

Рисунок 1 – Результат выполнения примера

### 2. Выполнил первое задание



```
def test():  
    value = input("Введите число: ")  
    if value < 0:  
        positive()  
    else:  
        negative()  
  
def positive():  
    print("Ваше число положительное")  
  
def negative():  
    print("Ваше число отрицательное")  
  
if __name__ == "__main__":  
    test()
```

Output:

```
Введите число: -5  
Ваше число отрицательное  
Process finished with exit code 0
```

Рисунок 2 – Результат выполнения первого задания

### 3. Выполнил второе задание

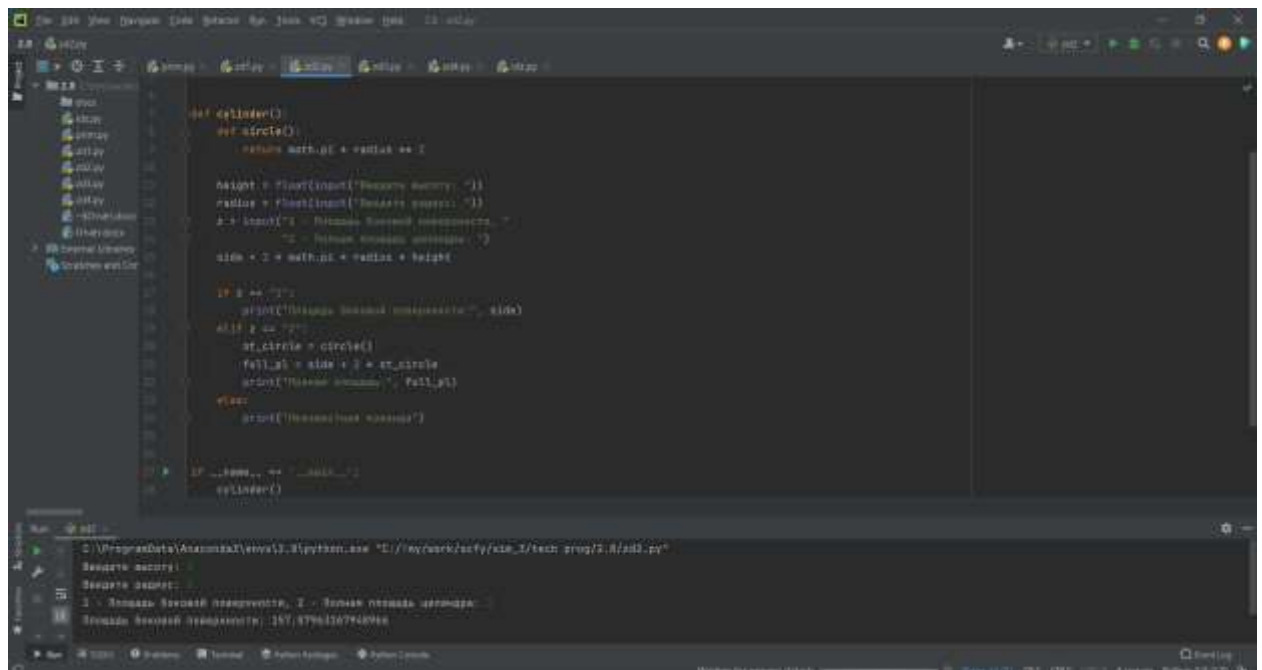


Рисунок 3 – Результат выполнения второго задания

#### 4. Выполнил третье задание

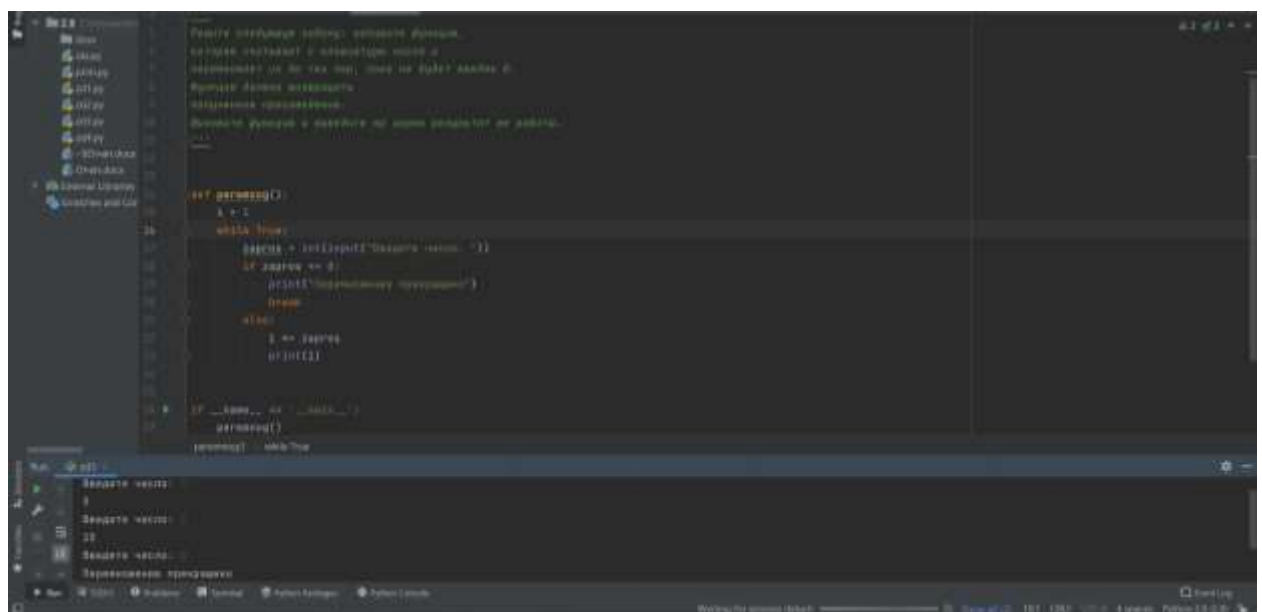


Рисунок 4 – Результат выполнения третьего задания

#### 5. Выполнил четвёртое задание

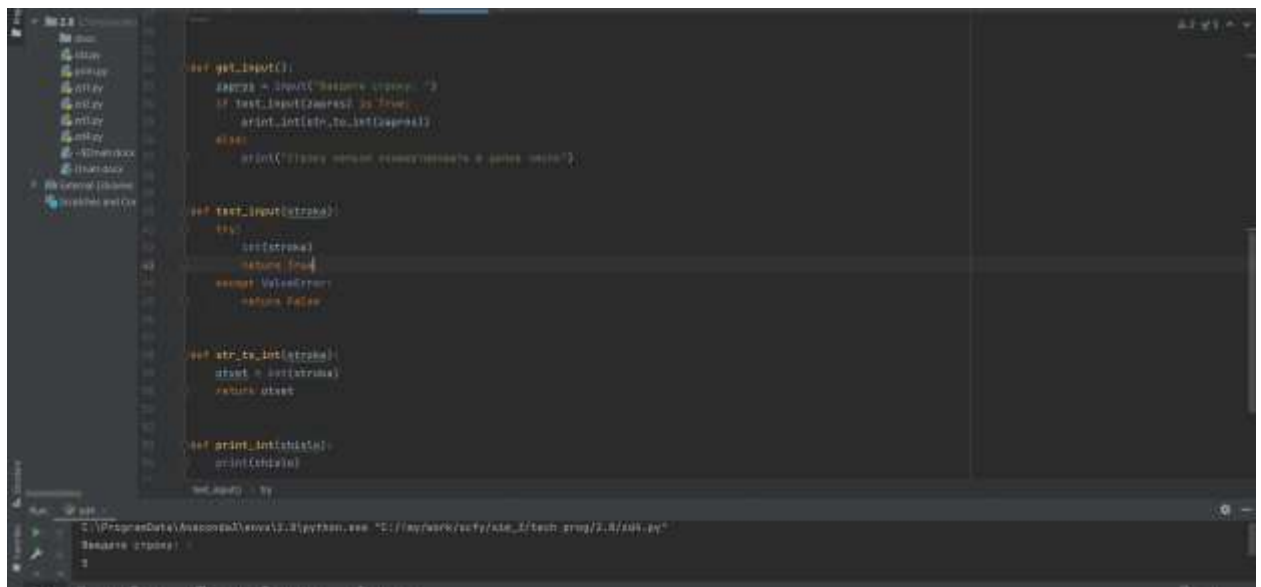


Рисунок 5 – Результат выполнения четвертого задания

## 6. Выполнил индивидуальное задание

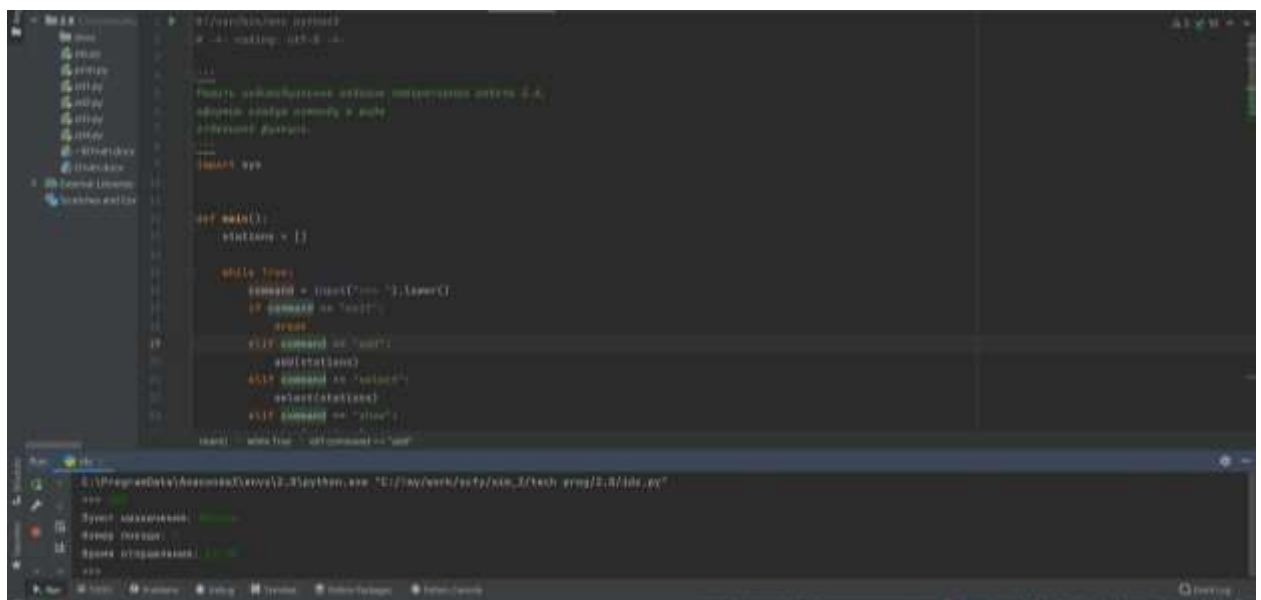


Рисунок 6 – Результат выполнения индивидуального задания

## 7. Провел проверку на PEP8 всех заданий

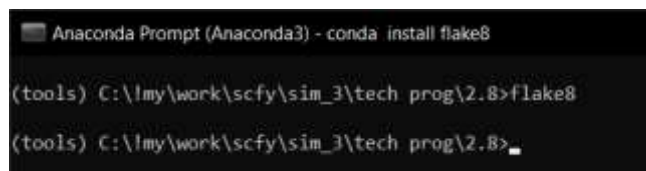


Рисунок 7 – Результат проверки на соответствие PEP8

Контрольные вопросы:

1. Каково назначение функций в языке программирования Python? - 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`? `def` – создаёт функцию, `return` – возвращает параметр из функции

3. Каково назначение локальных и глобальных переменных при написании функций в Python? - К глобальной переменной можно обратиться из локальной области видимости. К локальной переменной нельзя обратиться из глобальной области видимости, потому что локальная переменная существует только в момент выполнения тела функции.

4. Как вернуть несколько значений из функции Python? - Просто перечислить их через запятую в `return`

5. Какие существуют способы передачи значений в функцию?

— Любая функция может обратиться к глобальной переменной.

— В функцию можно передать значение при вызове: `function(значение)`

6. Как задать значение аргументов функции по умолчанию? - При определении функции, в скобках указать переменные и их значения: `function(параметр=значение)`

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. Ключевое слово `lambda` может появляться там, где синтаксис языка Python не позволяет использовать инструкцию `def` – внутри литералов или в вызовах функций.

8. Как осуществляется документирование кода согласно PEP257? - PEP 257 описывает соглашения, связанные со строками документации python, рассказывает о том, как нужно документировать python код. Строки документации - строковые литералы, которые являются первым оператором в

модуле, функции, классе или определении метода. Такая строка документации становится специальным атрибутом `__doc__` этого объекта. Все модули должны, как правило, иметь строки документации, и все функции, и классы, экспортируемые модулем также должны иметь строки документации. Публичные методы (в том числе `__init__`) также должны иметь строки документации. Пакет модулей может быть документирован в `__init__.py`.

9. В чем особенность однострочных и многострочных форм строк документации? - Однострочная строка документации не должна быть "подписью" параметров функции / метода (которые могут быть получены с помощью интроспекции). 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

Вывод: в результате выполнения лабораторной работы были приобретены навыки по работе с функциями при написании программ на языке Питон.