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 и алгоритмизации»

Выполнил студент группы ИВТ-б-о-20-1
Пушкин Н.С. « »_____20__г.
Подпись студента_____
Работа защищена« »_____20__г.

Проверил 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. После изучения теоретической части методических указаний, разобрал пример, описанный в документе.

```
def get_worker():
    """
    Запросить данные о работнике.
    """
    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))
    # Создать словарь.
    return {
        'name': name,
        'post': post,
        'year': year,
    }


def display_workers(workers):
    """
    Отобразить список работников.
    """
```

Рисунок 10.1 – Разбор примера

2. Затем приступил к выполнению общих заданий.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


def test():
    message = int(input('Введите целое число:'))
    if message > 0:
        positive(message)
    else:
        negative(message)


def negative(message):
    print(f'Число {message} отрицательное')


def positive(message):
    print(f"Число {message} положительное")


if __name__ == '__main__':
    test()
```

Рисунок 10.2 – Код первого общего задания

```
C:\Users\zligo\git\demo10\venv\Script
Введите целое число:50
Число 50 положительное

Process finished with exit code 0
```

Рисунок 10.3 – Результат выполнения кода

```python
def cylinder():
    radius = int(input('Введите радиус цилиндра: '))
    height = int(input('Введите высоту цилиндра: '))

    def circle():
        print('Площадь полной поверхности цилиндра: ',
              2 * 3.14 * radius * height + 2 * 3.14 * radius ** 2)

    message = input('Какую площать вы хотите получить: площадь бокой'
                    ' поверхности или полную площадь цилиндра?\n'
                    'Площадь бокой поверхности команда - неполная\n'
                    'Площадь полной поверхности цилинда команда - полная\n'
                    '>>>')
    if message.lower() == 'неполная':
        print('Площадь боковой повехности: ', 2 * 3.14 * radius * height)
    elif message.lower() == 'полная':
        circle()
    else:
        print('Неизвестная команда')


if __name__ == '__main__':
    cylinder()
```

Рисунок 10.4 – Код второго общего задания

```
Введите радиус цилиндра: 5
Введите высоту цилиндра: 5
Какую площать вы хотите получить: площадь бокой поверхности или полную площадь цилиндра?
Площадь бокой поверхности команда - неполная
Площадь полной поверхности цилинда команда - полная
>>>Полная
Площадь полной поверхности цилиндра:  314.0
```

Рисунок 10.5 – Результат выполнения кода

```python
def main():
    summ = 1
    while True:
        message = int(input("Введите число: "))
        summ *= message
        if summ == 0:
            print('Произведение равно 0')
            break
        else:
            print(f'Полчившееся прозведение: {summ}')


if __name__ == '__main__':
    main()
```

Рисунок 10.6 – Код третьего общего задания

```
Введите число: 2
Полчившееся прозведение: 2
Введите число: 4
Полчившееся прозведение: 8
Введите число: 5
Полчившееся прозведение: 40
Введите число: 0
Произведение равно 0

Process finished with exit code 0
```

Рисунок 10.7 – Результат выполнения кода

```python
def test_input(message):
    try:
        str_to_int(message)
    except ValueError:
        print('Нельзя преобразовать в число')


def str_to_int(message):
    i = int(message)
    print_int(i)


def print_int(i):
    print(i)


def get_input():
    message = input('Введите строку: ')
    test_input(message)


if __name__ == '__main__':
    get_input()
```

Рисунок 10.8 – Код четвёртого общего задания

```
Введите строку: 50
50
```

Рисунок 10.9 – Выполнение кода без ошибки

```
Введите строку: 20ф
Нельзя преобразовать в число

Process finished with exit code 0
```

Рисунок 10.10 – Выполнение кода с ошибкой

3. После разборки общих заданий, приступил к выполнению индивидуального задания для моего варианта.

```
def main():
    humans = []
    print('Список комманд: \n exit - Завершить работу'
          ' \n add - Добавить человека \n'
          ' list - Показать список людей'
          ' \n select - Выбрать знак зодиака по дате рождения')
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 20,
        '-' * 15,
        '-' * 16
    )
    while True:
        com = input('Введите команду: ').lower()
        if com == 'exit':
            break
        elif com == "add":
            add(humans)
        elif com == 'list':
            table(line, humans)
        elif com == 'select':
            select(line, humans)
        else:
            print(f"Неизвестная команда {com}", file=sys.stderr)


if __name__ == '__main__':
    main()
```

Рисунок 10.11 – Код индивидуального задания

Контрольные вопросы

1. Каково назначение функций в языке программирования Python?

Внедрение функций позволяет решить проблему дублирования кода в разных местах программы. Благодаря им можно исполнять один и тот же участок кода не сразу, а только тогда, когда он понадобится.

2. Каково назначение операторов def, return?

В языке программирования Python функции определяются с помощью оператора def. Ключевое слово def сообщает интерпретатору, что перед ним определение функции. За def следует имя функции. **Оператор return** завершает выполнение функции, в которой он задан, и возвращает управление в вызывающую функцию, в точку, непосредственно следующую за вызовом.

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

Локальные переменные видны только в локальной области видимости, которой может выступать отдельно взятая функция. Глобальные переменные видны во всей программе. "Видны" – значит, известны, доступны. К ним можно обратиться по имени и получить связанное с ними значение. 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. При выходе из нее, локальные переменные исчезают. Компьютерная память, которая под них отводилась, освобождается. Когда функция будет снова вызвана, локальные переменные будут созданы заново

4. Как вернуть несколько значений из функций?

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

5. Какие существуют способы передачи значений в функцию?

В программировании функции могут не только возвращать данные, но также принимать их, что реализуется с помощью так называемых параметров, которые указываются в скобках в заголовке функции. Количество параметров может быть любым. Параметры представляют собой локальные переменные, которым присваиваются значения в момент вызова функции. Конкретные значения, которые передаются в функцию при ее вызове, будем называть аргументами. Следует иметь в виду, что встречается иная терминология.

Например, формальные параметры и фактические параметры. В Python же обычно все называют аргументами. Обратим внимание еще на один момент. Количество аргументов и параметров совпадает. Нельзя передать три аргумента, если функция принимает только два. Нельзя передать один аргумент, если функция требует два обязательных. В рассмотренном примере они обязательные. Однако в Python у функций бывают параметры, которым уже присвоено значение по-умолчанию. В таком случае, при вызове можно не передавать соответствующие этим параметрам аргументы. Хотя можно и передать. Тогда значение по умолчанию заменится на переданное.

6. Как задать значение аргументов функции по умолчанию?

Нужно указать значения для параметров при описании функции

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 PRP257?

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

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

Однострочные строки документации предназначены для действительно очевидных случаев. Они должны умещаться на одной строке.

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 ходе выполнения лабораторной работы были поучены навыки по работе с функциями в Python.