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

Кафедра инфокоммуникаций

Отчет по лабораторной работе №3

Работа с функциями в языке Python

По дисциплине «Теории программирования и алгоритмизации»

Выполнил студент группы ИВТ-б-о-20-1

Бобров Н. В. « »_____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.

Ход работы:

Ссылка на репозиторий: https://github.com/nbobrov8/laba10

1. Проработал пример из методички.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys
from datetime import date


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.
    """
    # Проверить, что список работников не пуст.
    if workers:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
        print(line)
```

Рисунок 1 – Код примера

2. Приступил к выполнению заданий.

3. Написал код для первого задания.

Условие: основная ветка программы, не считая заголовков функций, состоит из двух строки кода. Это вызов функции test() и инструкции if __name__ == '__main__' . В ней запрашивается на ввод целое число. Если оно положительное, то вызывается функция positive(), тело которой содержит команду вывода на экран слова "Положительное". Если число отрицательное, то вызывается функция negative(), ее тело содержит выражение вывода на экран слова "Отрицательное".

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

import sys


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 2 – Код первого задания

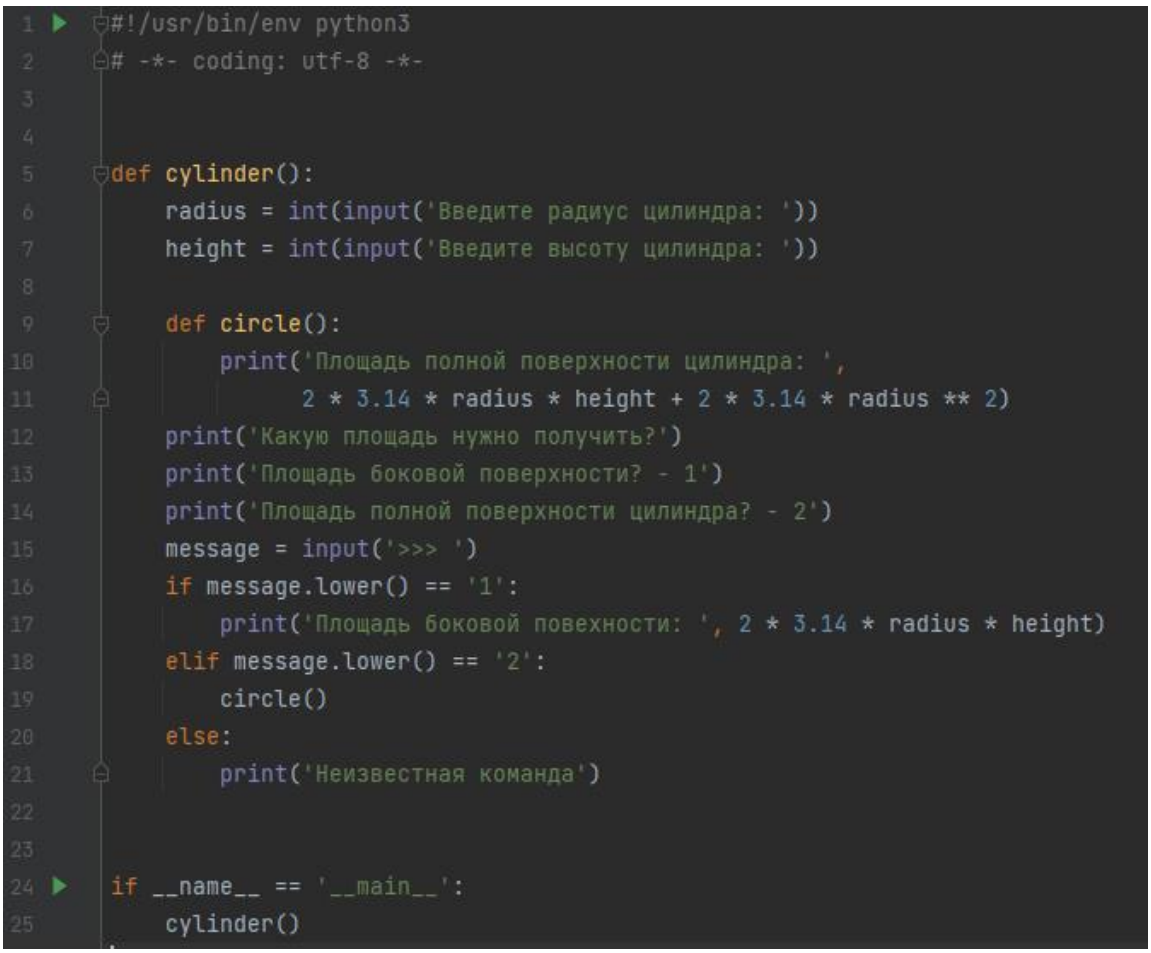
```
Введите целое число:15
Число 15 положительное

Process finished with exit code 0
```

Рисунок 3 – Результат выполнения программы

4. Приступил к выполнению второго задания.

Условие: в основной ветке программы вызывается функция cylinder(), которая вычисляет площадь цилиндра. В теле cylinder() определена функция circle(), вычисляющая площадь круга по формуле . В теле cylinder() у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле , или полную площадь цилиндра. В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции circle().



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


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)
    print('Какую площадь нужно получить?')
    print('Площадь боковой поверхности? - 1')
    print('Площадь полной поверхности цилиндра? - 2')
    message = input('>>> ')
    if message.lower() == '1':
        print('Площадь боковой повехности: ', 2 * 3.14 * radius * height)
    elif message.lower() == '2':
        circle()
    else:
        print('Неизвестная команда')


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

Рисунок 4 – Код для решения задачи

```
Введите радиус цилиндра: 2
Введите высоту цилиндра: 3
Какую площадь нужно получить?
Площадь боковой поверхности? - 1
Площадь полной поверхности цилиндра? - 2
>>> 1
Площадь боковой повехности:  37.68

Process finished with exit code 0
```

Рисунок 5 – Результат выполнения кода второго задания

5. Написал программу к третьей задаче.

Условие: напишите функцию, которая считывает с клавиатуры числа и перемножает их до тех пор, пока не будет введен 0. Функция должна возвращать полученное произведение. Вызовите функцию и выведите на экран результат ее работы.

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


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


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

Рисунок 6 – Код третьего задания

```
Введите число: 2
Произведение равно: 2
Введите число: 6
Произведение равно: 12
Введите число: 4
Произведение равно: 48
Введите число: 3
Произведение равно: 144
Введите число: 0
Произведение равно 0

Process finished with exit code 0
```

Рисунок 7 – Результат выполнения кода

6. Выполнил четвертое задание.

Условие: напишите программу, в которой определены следующие четыре функции:

1. Функция get_input() не имеет параметров, запрашивает ввод с клавиатуры и возвращает в основную программу полученную строку.

2. Функция test_input() имеет один параметр. В теле она проверяет, можно ли переданное ей значение преобразовать к целому числу. Если можно, возвращает логическое True.

Если нельзя – False.

3. Функция str_to_int() имеет один параметр. В теле преобразовывает переданное значение к целочисленному типу. Возвращает полученное число.

4. Функция print_int() имеет один параметр. Она выводит переданное значение на экран и ничего не возвращает.

В основной ветке программы вызовите первую функцию. То, что она вернула, передайте во вторую функцию. Если вторая функция вернула True, то те же данные (из первой функции) передайте в третью функцию, а возвращенное третьей функцией значение – в четвертую.

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

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 8 – Код четвертого задания

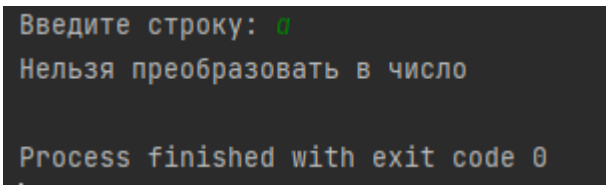


Рисунок 9 – Результат выполнения кода не с числами

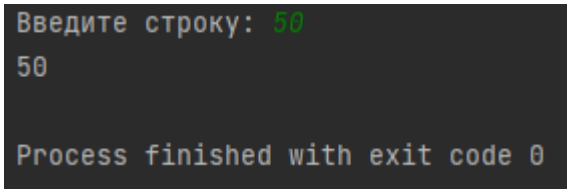


Рисунок 10 – Результат выполнения кода с числом

7. Приступил к выполнению индивидуального задания.

Использовал задание с работы 2.6 (Словари).

Условие: использовать словарь, содержащий следующие ключи: фамилия и инициалы; номер группы; успеваемость (список из пяти элементов). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по возрастанию номера группы; вывод на дисплей фамилий и номеров групп для всех студентов, включенных в массив, если средний балл студента больше 4.0; если таких студентов нет, вывести соответствующее сообщение.

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

import sys


def show_commands():
    print("Список команд:\n")
    print("add - добавить студента;")
    print("list - вывести список студентов;")
    print("select <средний балл> - запросить студентов с баллом выше 4.0;")
    print("exit - завершить работу с программой.")


def add_student():
    # Запросить данные о студенте.
    name = input("Фамилия и инициалы? ")
    group = input("Номер группы? ")
    grade = str(input('Успеваемость: '))
    # Создать словарь.
    student = {
        'name': name,
        'group': group,
        'grade': grade,
    }
    # Добавить словарь в список.
    students.append(student)
```

Рисунок 11 – Код индивидуального задания, часть 1

```
    # Отсортировать список в случае необходимости.
    if len(students) > 1:
        students.sort(key=lambda item: item.get('group')[::-1])


def show_list():
    # Заголовок таблицы.
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 30,
        '-' * 20,
        '-' * 15
    )
    print(line)
    print(
        '| {:^4} | {:^30} | {:^20} | {:^15} |'.format(
            "№",
            "Ф.И.О.",
            "Группа",
            "Успеваемость"
        )
    )
    print(line)
    # Вывести данные о всех студентах.
    for idx, student in enumerate(students, 1):
        print(
            '| {:>4} | {:<30} | {:<20} | {:>15} |'.format(
                idx,
                student.get('name', ''),
                student.get('group', ''),
                student.get('grade', 0)
```

Рисунок 12 – Код индивидуального задания, часть 2

```
                student.get('grade', 0)
            )
        )
    print(line)


def show_selected():
    # Инициализировать счетчик.
    count = 0
    # Проверить сведения студентов из списка.
    for student in students:
        grade = list(map(int, student.get('grade', '').split()))
        if sum(grade) / max(len(grade), 1) >= 4.0:
            print(
                '{:>4} {}'.format('*', student.get('name', '')),
                '{:>1} {}'.format('группа №', student.get('group', ''))
            )
            count += 1
    if count == 0:
        print("Студенты с баллом 4.0 и выше не найдены.")


def main():
    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break
```

Рисунок 13 – Код индивидуального задания, часть 3

```
89
90          elif command == 'add':
91              add_student()
92
93          elif command == 'list':
94              show_list()
95
96          elif command.startswith('select'):
97              show_selected()
98
99          else:
100             print(f"Неизвестная команда {command}", file=sys.stderr)
101
102
103 if __name__ == '__main__':
104     # Список студентов.
105     students = []
106     show_commands()
107
108     main()
```

Рисунок 14 – Код индивидуального задания, часть 4

8. Далее проверил работоспособность кода.

```
add - добавить студента;
list - вывести список студентов;
select <средний балл> - запросить студентов с баллом выше 4.0;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Бобров Н.В.
Номер группы? 5
Успеваемость: 5 4 5 5 4
>>> add
Фамилия и инициалы? Иванов И.И.
Номер группы? 1
Успеваемость: 2 3 4 2 3
>>> list
+------+--------------------------------+----------------------+----------------+
|  №   |            Ф.И.О.              |        Группа        |  Успеваемость  |
+------+--------------------------------+----------------------+----------------+
|    1 | Иванов И.И.                    | 1                    |       2 3 4 2 3 |
|    2 | Бобров Н.В.                    | 5                    |       5 4 5 5 4 |
+------+--------------------------------+----------------------+----------------+
>>> select
   * Бобров Н.В. группа № 5
>>> exit

Process finished with exit code 0
```

Рисунок 15 – Результат работы кода

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?

1) Любая функция может обратиться к глобальной переменной. 2) 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.

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 ходе выполнения лабораторной работы были приобретены навыки по работе с функциями при написании программ с помощью языка программирования Python версии 3.