МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Лабораторная работа 2.8

Работа с функциями в языке Python

Выполнил студент группы ИВТ-б-о-20-1

Симанский М.Ю « »_____20__г.

Подпись студента_____

Работа защищена « »_____20__г.

Проверил Воронкин Р.А. _____

(подпись)

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

Пример 1. Для примера 1 лабораторной работы 2.6, оформить каждую команду в виде вызова отдельной функции

Код

```
#!/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
    return{
        'name': name,
        'post': post,
        'year': year,
    }
def display_workers(staff):
    """
    Отобразить список работников.
    """
    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 30,
        '-' * 20,
        '-' * 8
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)
        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(workers, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('name', ''),
                    worker.get('post', ''),
                    worker.get('year', 0)
```

```
                )
            )
        print(line)
    else:
        print("Список работников пуст.")

def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """
    # Получить текущую дату.
    today = date.today()
    # Сформировать список работников.
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
            result.append(employee)
    # Возвратить список выбранных работников.
    return result

def main():
    """
    Главная функция программы.
    """
    # Список работников.
    workers = []
    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()
        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break
        elif command == 'add':
            # Запросить данные о работнике.
            worker = get_worker()
            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Отобразить всех работников.
            display_workers(workers)
        elif command.startswith('select '):
            # Разбить команду на части для выделения стажа.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемый стаж.
            period = int(parts[1])
            # Выбрать работников с заданным стажем.
            selected = select_workers(workers, period)
            # Отобразить выбранных работников.
            display_workers(selected)
        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)
```

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

Результат выполнения программы

```
Фамилия и инициалы? Симанский М.ю
Должность? Глвный
Год поступления? 1999
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Симанский М.ю                    | Глвный               |     1999 |
+------+----------------------------------+----------------------+----------+
>>>
```

Рисунок 1 – Результат

Задание 1

Решить следующую задачу: 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".

Код

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def positive():
    print("Число положительное")
def negative():
    print("Число отрицательное")
def test():
    num = int(input('Введите число '))
    if(num > 0):
        positive()
    else:
        negative()
```

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

Результат выполнения программы

```
C:\Users\rac-e\anaconda3\python.exe C:/Users/rac-e/PycharmProjects/pythonProject/venv/task1.py
Введите число -4
Число отрицательное

Process finished with exit code 0
```

Рисунок 2 – Программа выполнена

Задание 2

Решите следующую задачу: 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().

Код

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import math
def circle(r):
    return math.pi * r ** 2
def cylinder():
    r = float(input("Радиус "))
    h = float(input("Высота "))
    s = 2 * math.pi * r * h
    ans = int(input("Хотите получить площадь боковой поверхности(1) или
полную площадь(2) - "))
    if (ans == 1):
        print(s)
    elif(ans == 2):
        print(s + circle(r) * 2)
if __name__ == '__main__':
    cylinder()
```

Результат

```
C:\Users\rac-e\anaconda3\python.exe "C:/Users/rac-e/YandexDisk/Лабы 3 семестр/Технологии программирования/Лаб 10/zd2.py"
Радиус 5
Высота 4
Хотите получить площадь боковой поверхности(1) или полную площадь(2) - 2
282.6

Process finished with exit code 0
```

Рисунок 3 – Программа выполнена

Задание 3

Решите следующую задачу: 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.

Код

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def test():
    answer = 1
    while 1:
        num = int(input("Введите цифру "))
        if (num == 0) : break
        answer *= num
    return(answer)
print(test())
if __name__ == '__main__':
    test()
```

Результат выполнения работы

```
C:\Users\rac-e\anac
Введите цифру 1
Введите цифру 2
Введите цифру 4
Введите цифру 5
Введите цифру 7
Введите цифру 0
280
Введите цифру
```

Рисунок 4 – Результат выполнения программы

Задание 4

Решите следующую задачу: 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.

Код

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def get_input():
    str = input("Введите строку")
    return str
def test_input(str):
    try:
        int(str)
        return True
    except ValueError:
```

```
        return False
def str_to_int(str):
    a = int(str)
    return a
def print_int(str):
    print(str)
if __name__ == '__main__':
    str = get_input()
    ans = test_input(str)
    if(ans == True):
        print_int(str_to_int(str))
    else:
        print("Не число")
```

Результат

```
C:\Users\rac-e\anaconda3\python.exe "C:/l
Введите строку65
65

Process finished with exit code 0
```

Рисунок 5 – Результат выполнения программы

Индивидуальное задание

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

Код

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Использовать словарь, содержащий следующие ключи: название
товара; название магазина, в котором продается товар; стоимость товара в
руб. Написать программу, выполняющую следующие действия: ввод с
клавиатуры данных в список, состоящий из словарей заданной структуры;
записи должны быть размещены в алфавитном порядке по названиям
магазинов; вывод на экран информации о товарах, продающихся в магазине,
название которого введено с клавиатуры; если такого магазина нет, выдать на
дисплей соответствующее сообщение
"""
import sys

def get_shop():
    """
```

```
    Данная функция добавляет пары
    ключ-значение в словарь
    для каждого товара
    """
    name = input("Название магазина ")
    product = input("Товар ")
    price = int(input("Цена "))
    return{
        'name': name,
        'product': product,
        'price': price,
    }
def display_shops(shops):
    """
    Отображает данные о товаре в виде таблицы и
    Сортирует данные, по названию маганзина
    """
    line = '+-{}-+-{}-+-{}-+-{}-+'.format(
        '-' * 4,
        '-' * 30,
        '-' * 8,
        '-' * 20
    )
    print(line)
    print(
        '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
            "No",
            "Название.",
            "Товар",
            "Цена"
        )
    )
    print(line)
    for idx, shop in enumerate(shops, 1):
        print(
            '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(

                idx,
                shop.get('name', ''),
                shop.get('product', ''),
                shop.get('price', 0)

            )
        )
        print(line)
def select_shops(shops):
    """
    По заданому магазину находит товары, находящиеся в нем,
    если магазина нет - показывает соответсвующее сообщение
    """
    sname = input("Название магазина ")
    cout = 0
    for shop in shops:
        if (shop.get('name') == sname):
            cout = 1
            print(
                ' | {:<5} | {:<5} '.format(
                    shop.get('product', ''),
                    shop.get('price', 0),
                )
            )
        elif cout == 0:
            print("Такого магазина нет")
```

```python
def main():
    """
    главная функция программы
    """
    shops = []
    while True:
        command = input(">>> ").lower()
        if command == 'exit':
            break
        elif command == 'add':
            shop = get_shop()
            shops.append(shop)
            if len(shops) > 1:
                shops.sort(key=lambda item: item.get('product', ''))
        elif command == 'list':
            display_shops(shops)
        elif command.startswith('select '):
            selected = select_shops(shops)
            display_shops(selected)
        elif command == 'help':
            print("Список команд:\n")
            print("add - добавить магазин;")
            print("select - показать товары из заданного магазина;")
            print("list - вывести список магазинов;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)

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

Результат выполнения

```
>>> add
Название магазина magnit
Товар frkadelki
Цена 456
>>> pyaterochka
>>> Неизвестная команда pyaterochka

Неизвестная команда
>>> add
Название магазина lenta
Товар uran
Цена 344
>>> list
+------+--------------------------------+----------+----------------------+
|  No  |           Название.            |  Товар   |        Цена          |
+------+--------------------------------+----------+----------------------+
|    1 | magnit                         | frkadelki|                  456 |
+------+--------------------------------+----------+----------------------+
|    2 | lenta                          | uran     |                  344 |
+------+--------------------------------+----------+----------------------+
>>>
```

Рисунок 6 – Результат работы программы

Ответы на контрольные вопросы

1. Функции можно сравнить с небольшими программками, которые сами по себе, т. е. автономно, не исполняются, а встраиваются в обычную программу. Нередко их так и называют – подпрограммы. Других ключевых отличий функций от программ нет. Функции также при необходимости могут получать и возвращать данные. Только обычно они их получают не с ввода (клавиатуры, файла и др.), а из вызывающей программы. Сюда же они возвращают результат своей работы.

2. Def – определение функции. Return – возвращение функцией значения.

3. Соответственно, 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.

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

5. 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.

6. 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.

7. 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. Документирование кода в python - достаточно важный аспект, ведь от нее порой зависит читаемость и быстрота понимания вашего кода, как другими людьми, так и вами через полгода. 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. Одиночные 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.