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 №11
дисциплины «Программирование на Python»

Вариант 7

Выполнил:
Зармухамбетов Булат Эльдарович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Руководитель практики:
Воронкин Р. А.

_____
(подпись)

Отчет защищен с оценкой_____ Дата защиты_____

Ставрополь, 2023

Тема: Работа с функциями в языке Python

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.

Листинг:

```
#!/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(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(staff, 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()
```

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scripts\python.exe C:\User
>>> add
Фамилия и инициалы? Куцев Дмитрий Евгеньевич
Должность? стажёр
Год поступления? 2010
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Куцев Дмитрий Евгеньевич         | стажёр               |     2010 |
+------+----------------------------------+----------------------+----------+
>>> add
Фамилия и инициалы? Грачёв Денис Александрович
Должность? Доцент
Год поступления? 1988
>>> list
+------+----------------------------------+----------------------+----------+
|  №   |             Ф.И.О.               |      Должность       |   Год    |
+------+----------------------------------+----------------------+----------+
|    1 | Грачёв Денис Александрович       | Доцент               |     1988 |
|    2 | Куцев Дмитрий Евгеньевич         | стажёр               |     2010 |
+------+----------------------------------+----------------------+----------+
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
```

Рисунок 1. Тест №1.1

```
help - отобразить справку;
exit - завершить работу с программой.
>>> select 1
+------+---------------------------------+----------------------+----------+
|  №   |             Ф.И.О.              |      Должность       |   Год    |
+------+---------------------------------+----------------------+----------+
|    1 | Грачёв Денис Александрович      | Доцент               |     1988 |
|    2 | Куцев Дмитрий Евгеньевич        | стажёр               |     2010 |
+------+---------------------------------+----------------------+----------+
>>> select 10
+------+---------------------------------+----------------------+----------+
|  №   |             Ф.И.О.              |      Должность       |   Год    |
+------+---------------------------------+----------------------+----------+
|    1 | Грачёв Денис Александрович      | Доцент               |     1988 |
|    2 | Куцев Дмитрий Евгеньевич        | стажёр               |     2010 |
+------+---------------------------------+----------------------+----------+
>>> select 20
+------+---------------------------------+----------------------+----------+
|  №   |             Ф.И.О.              |      Должность       |   Год    |
+------+---------------------------------+----------------------+----------+
|    1 | Грачёв Денис Александрович      | Доцент               |     1988 |
+------+---------------------------------+----------------------+----------+
>>> exit

Process finished with exit code 0
```

Рисунок 2. Тест №1.2

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".

Листинг:

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

def test():
    """
    Запрашиваем число и вызываем соответствующую функцию.

    Вызыв positive(), если число положительное.
```

```
    Вызыов negative(), если отрицательное.
    Возвращение None, если == 0.
    """
    number = int(input("Введите целое число: "))
    if number > 0:
        positive()
    elif number < 0:
        negative()
    else:
        return


def positive():
    """
    Вывести на экран 'Положительное'.
    """
    print("Положительное")


def negative():
    """
    Вывести на экран 'Отрицательное'.
    """
    print("Отрицательное")


if_name == '_main_':
    test()
```

```
C:\Users\User\PycharmProjects\Python_laba_11\
Введите целое число: 45
Положительное

Process finished with exit code 0
```

Рисунок 3. Тест 1

```
C:\Users\User\PycharmProjects\Python_laba_11\
Введите целое число: -78
Отрицательное

Process finished with exit code 0
```

Рисунок 4. Тест 2

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().

Листинг:

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

import math


def circle(radius):
    return math.pi * radius ** 2


def cylinder():
    radius = float(input("Введите радиус основания цилиндра: "))
    height = float(input("Введите высоту цилиндра: "))

    choice = input("Хотите ли вы получить только площадь боковой поверхности
цилиндра? (да/нет): ")

    if choice.lower() == "да":
        side_area = 2 * math.pi * radius * height
        print("Площадь боковой поверхности цилиндра:", side_area)
    else:
        side_area = 2 * math.pi * radius * height
        full_area = side_area + 2 * circle(radius)
        print("Полная площадь цилиндра:", full_area)


cylinder()
```

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scripts\python.exe C:\Users\Us
Введите радиус основания цилиндра: 5
Введите высоту цилиндра: 8
Хотите ли вы получить только площадь боковой поверхности цилиндра? (да/нет): да
Площадь боковой поверхности цилиндра: 251.32741228718345

Process finished with exit code 0
```

Рисунок 5. Тест 1

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scripts\python.exe C:\Users\Use
Введите радиус основания цилиндра: 45.6
Введите высоту цилиндра: 123
Хотите ли вы получить только площадь боковой поверхности цилиндра? (да/нет): нет
Полная площадь цилиндра: 48306.133951245814

Process finished with exit code 0
```

Рисунок 5. Тест 2

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.

Листинг:

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

def multiplyyy_numbers():
    product = 1
    while True:
        num = float(input("Введите число (введите 0 для завершения): "))
        if num == 0:
            break
        product *= num
    return product

result = multiplyyy_numbers()
print("Произведение введенных чисел:", result)
```

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scrip
Введите число (введите 0 для завершения): 3
Введите число (введите 0 для завершения): 4
Введите число (введите 0 для завершения): 5
Введите число (введите 0 для завершения): 6
Введите число (введите 0 для завершения): 7
Введите число (введите 0 для завершения): 0
Произведение введенных чисел: 2520.0

Process finished with exit code 0
```

Рисунок 6. Тест 1

```
C:\Users\User\PycharmProjects\Python_laba_11\ven
Введите число (введите 0 для завершения): 5.6
Введите число (введите 0 для завершения): 345
Введите число (введите 0 для завершения): 6654
Введите число (введите 0 для завершения): 443
Введите число (введите 0 для завершения): 0
Произведение введенных чисел: 5694998903.999999

Process finished with exit code 0
```

Рисунок 7. Тест 2

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. 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.

Листинг:

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

def get_input():
    """
    Функция не имеет параметров, запрашивает ввод с клавиатуры.

    Возвращает в основную программу полученную строку
    """
    return input("Введите значение: ")

def test_input(value):
    """
    Имеет один параметр. В теле она проверяет, можно ли переданное
    ей значение преобразовать к целому числу.

    Если можно, возвращает логическое True.
    Если нельзя - False.
    """
    try:
        int(value)
        return True
    except ValueError:
        return False

def str_to_int(value):
    """
    Имеет один параметр. В теле преобразовывает переданное
    значение к целочисленному типу.
```

```
    Возвращает полученное число.
    """
    return int(value)

def print_int(value):
    """
    Имеет один параметр. Она выводит переданное значение на экран.

    Ничего не возвращает.
    """
    print(value)

# Основная ветка программы
input_value = get_input()
if test_input(input_value):
    int_value = str_to_int(input_value)
    print_int(int_value)
else:
    print("Введенное значение нельзя преобразовать к целому числу.")
```

```
C:\Users\User\PycharmProjects\Python_laba_11\ve
Введите значение: 35
35

Process finished with exit code 0
```

Рисунок 8. Тест 1

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scripts\
Введите значение: df
Введенное значение нельзя преобразовать к целому числу.

Process finished with exit code 0
```

Рисунок 9. Тест 2

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Script
Введите значение: 5.6
Введенное значение нельзя преобразовать к целому числу.

Process finished with exit code 0
```

Рисунок 10. Тест 3

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

Листинг:

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


import sys


def add_train(trains):
    """
    Функция для добавления информации о поезде в список trains.

    Parameters:
    trains (list): Список поездов.

    """
    destination = input('Название пункта назначения? ')
    number = input('Номер поезда? ')
    departure_time = input('Время отправления? ')

    train = {
        'destination': destination,
        'number': number,
        'departure_time': departure_time
    }

    trains.append(train)

    trains.sort(key=lambda item: item.get('departure_time', ''))


def list_trains(trains):
    """
    Функция для вывода списка всех поездов.

    Parameters:
    trains (list): Список поездов.

    """
    line = '+-{}-+{}-+{}-+'.format(
        '-' * 20,
        '-' * 15,
        '-' * 20
    )
    print(line)
    print(
        '| {:^20} | {:^15} | {:^20} |'.format(
            "Пункт назначения",
            "Номер поезда",
            "Время отправления"
        )
    )
    print(line)

    for idx, train in enumerate(trains, 1):
        print(
            '| {:<20} | {:^15} | {:^20} |'.format(
                train.get('destination', ''),
                train.get('number', ''),
                train.get('departure_time', '')
            )
        )
    print(line)
```

```python
def select_trains(trains, command):
    """
    Функция для вывода информации о поездах в заданном пункте назначения.

    Parameters:
    trains (list): Список поездов.
    command (str): Команда в формате "select <пункт_назначения>".

    """
    parts = command.split(' ', maxsplit=1)
    destination = parts[1]

    selected_trains = [train for train in trains if train['destination'] ==
destination]

    if selected_trains:
        line = '+-{}-+{}-+{}-+'.format(
            '-' * 20,
            '-' * 15,
            '-' * 20
        )
        print(line)
        print(
            '| {:^20} | {:^15} | {:^20} |'.format(
                "Пункт назначения",
                "Номер поезда",
                "Время отправления"
            )
        )
        print(line)
        for train in selected_trains:
            print(
                '| {:<20} | {:^15} | {:^20} |'.format(
                    train.get('destination', ''),
                    train.get('number', ''),
                    train.get('departure_time', '')
                )
            )
        print(line)
    else:
        print(f'Поездов в пункт "{destination}" не найдено')


def show_help():
    """
    Функция для вывода списка доступных команд.

    """
    print('Список команд:\n')
    print('add - добавить информацию о поезде;')
    print('list - вывести список всех поездов;')
    print('select <пункт_назначения> - запросить информацию о поездах в
заданном пункте назначения;')
    print('exit - завершить работу с программой.')


if__name__ == '__main__':
    trains = []

    while True:
        command = input('>>> ').lower()
```

```
        if command == 'exit':
            break
        elif command == 'add':
            add_train(trains)
        elif command == 'list':
            list_trains(trains)
        elif command.startswith('select '):
            select_trains(trains, command)
        elif command == 'help':
            show_help()
        else:
            print(f'Неизвестная команда "{command}"!', file=sys.stderr)
```

```
>>> add
Название пункта назначения? stavropol
Номер поезда? 111
Время отправления? 11:00
>>> stavropol
>>> Неизвестная команда "stavropol"!
add
Название пункта назначения? stavropol
Номер поезда? 222
Время отправления? 12:00
>>> list
+----------------------+----------------+----------------------+
|   Пункт назначения   |  Номер поезда  |  Время отправления   |
+----------------------+----------------+----------------------+
| stavropol            |       111      |        11:00         |
| stavropol            |       222      |        12:00         |
+----------------------+----------------+----------------------+
```

Рисунок 11. Тест 1

```
C:\Users\User\PycharmProjects\Python_laba_11\venv\Scripts\python.exe C:\Users\User\Pychar
>>> help
Список команд:

add - добавить информацию о поезде;
list - вывести список всех поездов;
select <пункт_назначения> - запросить информацию о поездах в заданном пункте назначения;
exit - завершить работу с программой.
>>> list
+----------------------+----------------+----------------------+
|   Пункт назначения   |  Номер поезда  |  Время отправления   |
+----------------------+----------------+----------------------+
+----------------------+----------------+----------------------+
>>> info
>>> Неизвестная команда "info"!
```

Рисунок 12. Тест 2

Вывод: в ходе выполнения данной лабораторной работы были приобретены навыки по взаимодействию с функциями при написании программ с помощью языка программирования Python версии 3.x.

Ответы на контрольные вопросы

1. Основной целью функций в языке программирования Python является разделение кода на повторно используемые блоки, которые могут выполнять определенные задачи. Функции позволяют более эффективное написание, отладку и поддержку кода.

2. Оператор def используется для определения функции в Python. Он позволяет задать имя функции и список параметров, которые функция принимает. Оператор return используется для возврата значения из функции. После выполнения оператора return, функция завершается и возвращает результат вызывающему коду.

3. Локальные переменные - это переменные, которые определены внутри функции и доступны только внутри этой функции. Глобальные переменные - переменные, которые определены за пределами функции и доступны в любом месте программы. При использовании локальных переменных, можно избежать конфликтов имён и сохранить состояние функции. Глобальные переменные позволяют обмениваться данными между функциями.

4. Для возврата нескольких значений из функции в Python, можно использовать кортежи, списки или словари. Например, функция может вернуть кортеж с несколькими значениями с помощью оператора return (значение1, значение2).

5. Существуют несколько способов передачи значений в функцию в Python:

- Позиционные аргументы: значения передаются в том же порядке, в котором параметры определены в функции.

- Именованные аргументы: значения передаются с указанием имени параметра в виде "имя=значение".

- Аргументы по умолчанию: параметрам функции могут быть назначены значения по умолчанию, которые будут использоваться, если при вызове функции эти параметры не указаны.

6. Значения аргументов функции по умолчанию можно задать при определении функции. Для этого используется следующий синтаксис: def имя_функции(параметр1=значение_по_умолчанию, параметр2=значение_по_умолчанию).

7. Lambda-выражения в Python представляют собой анонимные функции, которые могут содержать только одно выражение. Они часто используются в качестве аргументов функций высшего порядка или при работе с функциями map, filter, reduce.

8. Для документирования кода в соответствии со стандартом PEP257 в Python используется строка документации, которая является первой строкой после определения функции, класса или модуля. Строка документации должна быть заключена в тройные кавычки и содержать описание функции, класса или модуля. Она является частью документации и может быть извлечена с помощью инструментов автоматической генерации документации.

9. Однострочные формы строк документации обычно используются для описания функций, классов или модулей, которые имеют небольшое описание. Многострочные формы строк документации обычно используются для более подробного описания, которое может содержать несколько абзацев или форматирование в виде списков, таблиц и т.д.