

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 №2\_8**  
**дисциплины**  
**«Основы программной инженерии»**

Выполнил:

Коныжев Максим Викторович  
2 курс, группа ПИЖ-б-о-21-1,  
09.03.04 «Программная инженерия»,  
направленность (профиль) «Разработка  
и сопровождение программного  
обеспечения», очная форма обучения

---

(подпись)

Проверил:

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь 2022

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

1. Был создан репозиторий в Github в который были добавлены правила gitignore для работы IDE PyCharm, была выбрана лицензия MIT, сам репозиторий был клонирован на локальный сервер и был организован в соответствии с моделью ветвления git-flow.

Owner \* Repository name \*

MaxDrill / lab2\_8 ✓

Great repository names are short and memorable. Need inspiration? How about [laughing-garbanzo](#)?

Description (optional)

☒ Public  
Anyone on the internet can see this repository. You choose who can commit.

☐ Private  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ Add a README file  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License

This will set `main` as the default branch. Change the default name in your [settings](#).

*i* You are creating a public repository in your personal account.

Create repository

Рисунок 1 – Создание репозитория

```

C:\Users\UESR\gitproj>git clone https://github.com/MaxDrill/lab2_8.git
Cloning into 'lab2_8'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
C:\Users\UESR\gitproj>

```

Рисунок 2 – Клонирование репозитория

```

C:\Users\UESR\Desktop\2.1\proj\venv\Scripts\python.exe C:/Users/UESR/Desktop/2.1
>>> add
Фамилия и инициалы? Max K
Должность? director
Год поступления? 2000
>>> list
+-----+-----+-----+-----+
| № |           Ф.И.О.           |     Должность     |   Год   |
+-----+-----+-----+-----+
|  1 | Max K                       |     director      |   2000   |
+-----+-----+-----+-----+
>>> select 5
      1: Max K
>>> exit

Process finished with exit code 0

```

Рисунок 6 – Результат работы программы

### Задача 1

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".

Понятно, что вызов `test()` должен следовать после определения функций. Однако имеет ли значение порядок определения самих функций? То есть должны ли определения `positive()` и `negative()` предшествовать `test()` или могут следовать после него? Проверьте вашу гипотезу, поменяв объявления функций местами. Попробуйте объяснить результат.

Код программы:

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

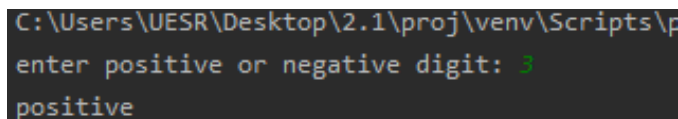
import sys

def test():
    """
    inputing a digit and checking the meaning """
    dig = int(input('enter positive or negative digit: '))
    if dig >= 0:
        positive()
    else:
        negative()

def positive():
    """
    output to the screen """
    print('positive')

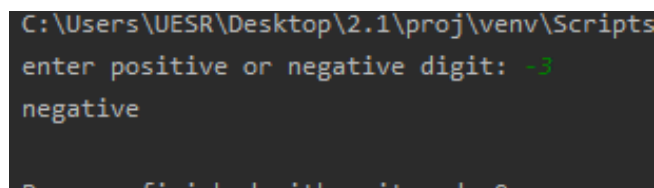
def negative():
    """
    output to the screen """
    print('negative')

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



C:\Users\UESR\Desktop\2.1\proj\venv\Scripts\p  
enter positive or negative digit: 3  
positive

Рисунок 7 – Результат работы программы при положительном числе



C:\Users\UESR\Desktop\2.1\proj\venv\Scripts\p  
enter positive or negative digit: -3  
negative  
Program finished with exit code 0

Рисунок 8 – Результат работы программы при отрицательном числе

## Задача 2

В основной ветке программы вызывается функция `cylinder()`, которая вычисляет площадь цилиндра. В теле `cylinder()` определена функция `circle()`, вычисляющая площадь круга по формуле  $S = \pi r^2$ . В теле `cylinder()` у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле  $S_{\text{бок}} = 2\pi r h$ , или полную площадь цилиндра.

В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции circle()

Код программы:

```
#!/usr/bin/env python3
# __coding: utf-8 __

import math

def cylinder():
    """
    calculation of the full or side square cylinder
    """
    while True:
        option = int(input("<<<What kind of square do you want to calculate?
>>>\n"
                           "1 - Side surface\n"
                           "2 - Full surface\n"
                           ">>> "))
        if (option != 1) and (option != 2):
            print('unknown command')
            break
        r = int(input("Enter the radius: "))
        h = int(input("Enter the height of the cylinder: "))
        if option == 1:
            s = 2 * math.pi * r * h
            print("S(side.) = ", '{:.3f}'.format(s))
            break
        else:
            s = (2 * math.pi * r * h) + (circle(r) * 2)
            print("S(full.) = ", '{:.3f}'.format(s))
            break

def circle(r):
    """
    Calculation of the square of the circle by a given radius
    """
    return math.pi * (r ** 2)

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

```
<<<What kind of square do you want to calculate?>>>
1 - Side surface
2 - Full surface
>>> 1
Enter the radius: 5
Enter the height of the cylinder: 3
S(side.) = 94.248

Process finished with exit code 0
```

Рисунок 9 – Результат работы программы

### Задача 3:

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 программы:

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

def multiply():
    """
    multiplying digits while it is not 0
    """
    print("<<<Enter digits that will be multiplied>>>")
    res = 1
    while True:
        a = int(input(">>> "))
        if a == 0:
            break
        else:
            res *= a
    return res

if __name__ == '__main__':
    print('result is', multiply())
```

```
<<<Enter digits that will be multiplied>>>
>>> 3
>>> 5
>>> 6
>>> 7
>>> 0
result is 630
Process finished with exit code 0
```

Рисунок 10 – Результат работы 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. 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.

#### Код программы

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

import sys

def get_input():
    """
    requests keyboard input and returns the given string
    """
    get_str = input("enter digit: ")
    return get_str

def test_input(a):
    """
    checks if the given string can be transformed into a number. If yes, it
    returns yes.
    """
    if type(a) == int or type(a) == float:
        return True
    elif a.isnumeric():
        return True
    else:
        return False

def str_to_int(b):
    """
    converts transmitted meaning into int type
    """
    c = int(b)
    return c

def print_int(c):
    """
    Displays the transmitted value on the screen
    """
    print(c)

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

```

bol = test_input(s)
if bol:
    nmb = str_to_int(s)
    print_int(nmb)
else:
    print(f"the entered value is not a number!", file=sys.stderr)

```

```

C:\Users\UESR\Desktop\2.1\proj\venv\Scripts\python
enter digit: 7
7

Process finished with exit code 0

```

Рисунок 11 – Результат работы программы

### Индивидуальное задание 7

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

Код программы:

```

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

import sys

def get_flight():
    dest = input("What destination do you need? ")
    numb = int(input("What number of the train? "))
    date = float(input("What time do you need? "))

    # creation of dictionary
    return {
        'destination': dest,
        'number_train': numb,
        'flight_date': date,
    }

def display_flights(train):
    if train:
        line = '+-{}--{}--{}--{}--'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 18
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^18} |'.format(
                "№",
                "Destination",
                "Number of the train",
                "Flight Date"
            )
        )
        print(line)
        for i, item in enumerate(train, 1):
            print(

```



```

        '| {:>4} | {:<30} | {:<20} | {:>18} |'.format(
            i,
            item.get('destination', ''),
            item.get('number_train', ''),
            item.get('flight_date', 0)
        )
    )
    print(line)
else:
    print('list is empty')

def select_flights(train):
    t = input("enter your destination: ")
    count = 0
    for i in train:
        if i.get('destination') == t:
            count += 1
            print(
                '{:>4}: {} {} {}'.format(
                    count,
                    i.get('destination', ''),
                    i.get('flight_date', ''),
                    i.get("number_train")
                )
            )
    if count == 0:
        print("We couldn't find this destination")

def main():
    # list of the flights
    train = []

    # organization of an endless loop
    while True:
        info = input(">>> ").lower()
        if info == 'exit':
            break

        elif info == 'add':
            flight = get_flight()
            train.append(flight)
            if len(train) > 1:
                train.sort(key=lambda item: item.get('flight_date', ''))

        elif info == 'list':
            display_flights(train)

        elif info.startswith('select'):
            select_flights(train)

        elif info == 'help':
            print("command list:\n")
            print("add - add information about a flight;")
            print("list - display the flight schedule;")
            print("select <type> - select destination;")
            print("help - show reference;")
            print("exit - leave a program.")
        else:
            print(f"unknown command {info}", file=sys.stderr)

if __name__ == '__main__':
    main()

```

```

C:\Users\UESR\Desktop\2.1\proj\venv\Scripts\python.exe C:/Users/UESR/Desktop/2.1/proj/lab11_id
>>> add
What destination do you need? london
What number of the train? 10
What time do you need? 12.45
>>> add
What destination do you need? oslo
What number of the train? 77
What time do you need? 21.15
>>> list
+-----+-----+-----+-----+
| № | Destination | Number of the train | Flight Date |
+-----+-----+-----+-----+
| 1 | london | 10 | 12.45 |
| 2 | oslo | 77 | 21.15 |
+-----+-----+-----+-----+
>>> select
enter your destination: oslo
1: oslo 21.15 77
>>> exit

```

Рисунок 12 – Результат работы программы

```

C:\Users\UESR\gitproj\lab2_8>git commit -m "Add proj"
[main 4a5b4c4] Add proj
6 files changed, 375 insertions(+)
create mode 100644 proj/lab11_ex.py
create mode 100644 proj/lab11_idz.py
create mode 100644 proj/lab11_num1.py
create mode 100644 proj/lab11_num2.py
create mode 100644 proj/lab11_num3.py
create mode 100644 proj/lab11_num4.py
C:\Users\UESR\gitproj\lab2_8>

```

Рисунок 14 – Коммит

Вывод: в результате лабораторной работы были приобретены навыки по работе с функциями при написании программ с помощью языка программирования Python версии 3.x.

## Контрольные вопросы и ответы на них:

### 1. Каково назначение функций в языке программирования Python?

Главной задачей функций в Python, как и в других языках программирования, является сокращение объёма кода и его структуризация. В функции, как правило, выносятся те части кода, которые выполняются в программе многократно.

### 2. Каково назначение операторов `def` и `return`?

Оператор `def` необходим для определения функции. После него идёт название самой функции, передаваемые в функцию параметры и само тело функции. Оператор `return` служит для возвращения результата выполнения функции в основную программу, где эта функция была вызвана.

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

Локальные переменные существуют только внутри функции. В другой части программы как-либо вызывать или изменить их невозможно.

Глобальные напротив – существуют во всей программе.

### 4. Как вернуть несколько значений из функции Python?

После оператора `return` необходимо записать все возвращаемые переменные через запятую, а при вызове функции нужно задать необходимое количество переменных. Куда будут возвращены параметры.

### 5. Какие существуют способы передачи значений в функцию? По ссылке и по значению.

### 6. Как задать значение аргументов функции по умолчанию?

Нужно в скобках передаваемых параметров присвоить им значение.

### 7. Каково назначение `lambda`-выражений в языке Python?

`Lambda`-выражения – это небольшие функции, которые вызываются в программе один раз.

### 8. Как осуществляется документирование кода согласно PEP257?

Если пояснение функции содержит одну строку, то достаточно двух кавычек с каждой стороны строки. Пример: `""" Пояснение """`. Если это

многострочное пояснение, то необходимо три кавычки с каждой стороны.

Пояснение находится в теле функции, сразу после её объявления