МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФГАОУ ВО
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
Институт цифрового развития

Кафедра инфокоммуникаций

дисциплина Языки программирования

Отчет по лабораторной работе №2.12

Декораторы
функций в языке Python

Выполнил: студент группы ИТС-б-о-21-1
Пушкин Максим Алексеевич

_____

(подпись)

Проверил: кандидат технических наук, доцент кафедры
инфокоммуникаций,
Роман Александрович Воронкин

_____

(подпись)

Ставрополь, 2022

1. **Цель работы:** приобретение навыков по работе с декораторами функций при написании программ с помощью языка программирования Python версии 3.x.

2. **Теоретическое сведения**

Декораторы — один из самых полезных инструментов в Python, однако новичкам они могут показаться непонятными.

То же касается и объектно-ориентированного программирования, где вы определяете классы и создаёте на их основе объекты. Декораторы не принадлежат ни одной из этих парадигм и исходят из области функционального программирования. Однако не будем забегать вперёд, разберёмся со всем по порядку.

Декоратор — это функция, которая позволяет обернуть другую функцию для расширения её функциональности без непосредственного изменения её кода. Вот почему декораторы можно рассматривать как практику метапрограммирования, когда программы могут работать с другими программами как со своими данными. Чтобы понять, как это работает, сначала разберёмся в работе функций в Python. Как работают функции Все мы знаем, что такое функции, не так ли? Не будьте столь уверены в этом.

У функций Python есть определённые аспекты, с которыми мы нечасто имеем дело, и, как следствие, они забываются. Давайте проясним, что такое функции и как они представлены в Python. Функции как процедуры С этим аспектом функций мы знакомы лучше всего. Процедура — это именованная последовательность вычислительных шагов. Любую процедуру можно вызвать в любом месте программы, в том числе внутри другой процедуры или даже самой себя. По этой части больше нечего

сказать, поэтому переходим к следующему аспекту функций в Python. Функции как объекты первого класса

В Python всё является объектом, а не только объекты, которые вы создаёте из классов. В этом смысле он (Python) полностью соответствует идеям объектно-ориентированного программирования.

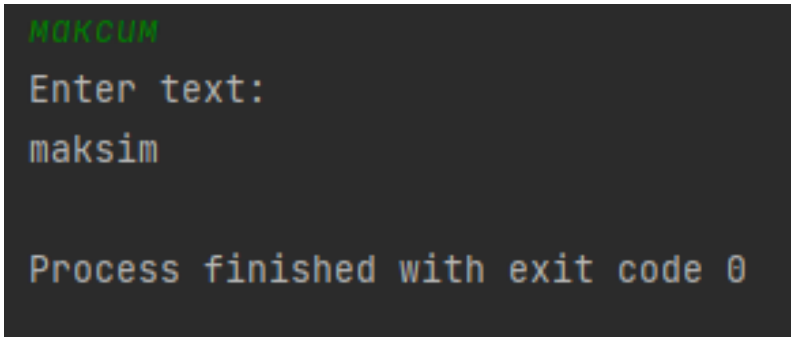
Метод и порядок выполнения

4. Объявите функцию с именем to_lat , которая принимает строку на кириллице и преобразовывает ее в латиницу, используя следующий словарь для замены русских букв на соответствующее латинское написание,

Код:

```
main.py

# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    def decorator_function(func):
        def to_lat(st):
            t = {'ё': 'yo', 'а': 'a', 'б': 'b', 'в': 'v', 'г': 'g', 'д': 'd', 'е': 'e',
                 'ж': 'zh',
                 'з': 'z', 'и': 'i', 'й': 'y', 'к': 'k', 'л': 'l', 'м': 'm', 'н': 'n', 'о':
                     'o', 'п': 'p',
                 'р': 'r', 'с': 's', 'т': 't', 'у': 'u', 'ф': 'f', 'х': 'h', 'ц': 'c', 'ч':
                     'ch', 'ш': 'sh',
                 'щ': 'shch', 'ъ': '', 'ы': 'y', 'ь': '', 'э': 'e', 'ю': 'yu', 'я': 'ya'}

            alphabet = ['Ъ', 'ь', 'Ъ', 'ъ', 'А', 'а', 'Б', 'б', 'В', 'в', 'Г', 'г', 'Д', 'д', 'Е', 'е', 'Ё', 'ё',
                        'Ж', 'ж', 'З', 'з', 'И', 'и', 'Й', 'й', 'К', 'к', 'Л', 'л', 'М', 'м', 'Н', 'н', 'О', 'о',
                        'П', 'п', 'Р', 'р', 'С', 'с', 'Т', 'т', 'У', 'у', 'Ф', 'ф', 'Х', 'х', 'Ц', 'ц', 'Ч', 'ч',
                        'Ш', 'ш', 'Щ', 'щ', 'Ы', 'ы', 'Э', 'э', 'Ю', 'ю', 'Я', 'я']
            print("Enter text: ")
            result = str()

            len_st = len(st)
            for i in range(0, len_st):
                if st[i] in alphabet:
                    simb = t[st[i]]
                else:
                    simb = st[i]
                result = result + simb

            print(result)
        return to_lat
    st = str(input())
    @decorator_function
    def func(st):
        print(st)
    func(st)
```

Результат:



Git:

https://github.com/manulmax21/lab_rab2.git

Вопросы для защиты работы

1. Что такое декоратор?

   один из самых полезных инструментов в Python, однако новичкам они могут показаться непонятными. Возможно, вы уже встречались с ними, например, при работе с Flask, но не хотели особо вникать в суть их работы

2. Почему функции являются объектами первого класса?

Объектами первого класса в контексте конкретного языка программирования называются элементы, с которыми можно делать всё то же, что и с любым другим объектом: передавать как параметр, возвращать из функции и присваивать переменной.

3. Каково назначение функций высших порядков?

Функции высших порядков — это такие функции, которые могут принимать в качестве аргументов и возвращать другие функции.

4. Как работают декораторы?

Декоратор — это функция, которая позволяет обернуть другую функцию для расширения её функциональности без непосредственного изменения её кода.

5. Какова структура декоратора функций?

def decorator_function(func):

def wrapper():

```
print('Функция-обёртка!')
print('Оборачиваемая функция: {}'.format(func))
print('Выполняем обёрнутую функцию...')
func()
print('Выходим из обёртки')
return wrapper
```

Здесь decorator_function() является функцией-декоратором. Как вы могли заметить, она является функцией высшего порядка, так как принимает функцию в качестве аргумента, а также возвращает функцию. Внутри decorator_function() мы определили другую функцию, обёртку, так сказать, которая обёртывает функцию-аргумент и затем изменяет её поведение. Декоратор возвращает эту обёртку.

Вывод: в ходе работы были приобретены приборетины навыки по работе с декораторами функций при написании программ с помощью языка программирования Python версии 3.x.