

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**  
**Федеральное государственное автономное  
образовательное учреждение высшего образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Кафедра инфокоммуникаций**

**Институт цифрового развития**

**ОТЧЁТ**

**по лабораторной работе**

Дисциплина: «Объектно – ориентированное программирование»

Выполнил студент группы

ИВТ-б-о-21-1

Яковлева Е.А. « » \_\_\_\_\_ 20\_\_ г.

Подпись студента \_\_\_\_\_

Работа защищена « » \_\_\_\_\_ 20\_\_ г.

Проверил доцент

Кафедры инфокоммуникаций, старший  
преподаватель

Воронкин Р.А.

\_\_\_\_\_  
(подпись)

## Аннотация типов

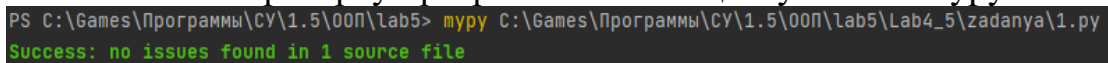
**Цель работы:** приобретение навыков по работе с аннотациями типов при написании программ с помощью языка программирования Python версии 3.x. Рассмотрен вопрос контроля типов переменных и функций с использованием комментариев и аннотаций. Приведено описание PEP'ов, регламентирующих работу с аннотациями, и представлены примеры работы с инструментом муру для анализа Python кода.

**Ход работы:**

**Индивидуальное задание.**

Выполнить индивидуальное задание 2 лабораторной работы 2.19, добавив аннотации типов.

Выполнить проверку программы с помощью утилиты муру.



```
PS C:\Games\Программы\СУ\1.5\00П\lab5> mypy C:\Games\Программы\СУ\1.5\00П\lab5\Lab4_5\zadanya\1.py
Success: no issues found in 1 source file
```

Рисунок 1. Установка утилиты и проверка программы

### 1. Для чего нужны аннотации типов в языке Python?

Аннотации типов в языке Python представляют собой способ указать ожидаемый тип данных для аргументов функций, возвращаемых значений функций и переменных. Вот несколько причин, по которым аннотации типов могут быть полезны:

1. Документация: Аннотации типов могут служить документацией для кода, помогая другим разработчикам понять ожидаемые типы данных в функциях и методах.

2. Поддержка инструментов статического анализа: Аннотации типов могут использоваться инструментами статического анализа кода, такими как Муру, Руге или Pyright, чтобы проверять соответствие типов данных во время компиляции или анализа кода.

3. Улучшение читаемости: Аннотации типов могут помочь улучшить читаемость кода, особенно в случае сложных или больших проектов, где явное указание типов данных может помочь понять назначение переменных и результатов функций.

4. Интеграция с IDE: Некоторые интегрированные среды разработки (IDE), такие как PyCharm, могут использовать аннотации типов для

предоставления подсказок о типах данных и автоматической проверки соответствия типов.

## **2. Как осуществляется контроль типов в языке Python?**

В языке Python контроль типов данных может осуществляться несколькими способами:

1. Аннотации типов: Как уже упоминалось, в Python можно использовать аннотации типов для указания ожидаемых типов данных для аргументов функций, возвращаемых значений функций и переменных. Это позволяет документировать ожидаемые типы данных и использовать инструменты статического анализа кода для проверки соответствия типов.

2. Использование инструментов статического анализа: Существуют сторонние инструменты, такие как Муру, Pyre и Pyright, которые могут использоваться для статической проверки соответствия типов данных в Python-коде. Эти инструменты могут обнаруживать потенциальные ошибки типов данных и предоставлять рекомендации по улучшению кода.

3. Вручную проверять типы данных: В Python можно вручную выполнять проверку типов данных с помощью условных операторов и функций, таких как `isinstance()`. Например, можно написать условие для проверки типа данных перед выполнением определенной операции.

4. Использование аннотаций типов в комбинации с декораторами: В Python можно использовать декораторы, такие как `@overload` из модуля `functools`, для реализации перегрузки функций с разными типами аргументов.

## **3. Какие существуют предложения по усовершенствованию Python для работы с аннотациями типов?**

Предложения по усовершенствованию работы с аннотациями типов в Python включают расширение поддержки аннотаций типов, улучшение интеграции с инструментами статического анализа, улучшение документации и рекомендаций, а также разработку стандартной библиотеки типов. Эти изменения могут сделать работу с аннотациями типов более мощной и удобной для разработчиков.

## **4. Как осуществляется аннотирование параметров и возвращаемых значений функций?**

В Python аннотирование параметров и возвращаемых значений функций осуществляется с использованием двоеточия и указания типа данных после имени параметра или перед знаком ">" для возвращаемого значения. Например:

```
def greet(name: str) -> str:  
    return "Hello, " + name
```

В этом примере name: str указывает, что параметр name должен быть строкой, а -> str указывает, что функция возвращает строку.

### **5. Как выполнить доступ к аннотациям функций?**

В Python можно получить доступ к аннотациям функций с помощью специального атрибута `__annotations__`. Этот атрибут содержит словарь, в котором ключами являются имена параметров или "return" (для возвращаемого значения), а значениями - указанные типы данных.

Пример:

```
def greet(name: str) -> str:  
    return "Hello, " + name  
print(greet.__annotations__)
```

Этот код выведет на экран словарь с аннотациями функции greet:

```
{'name': <class 'str'>, 'return': <class 'str'>}
```

Таким образом, вы можете получить доступ к аннотациям функции и использовать их в своем коде, например, для проверки типов данных или для документирования функций.

### **6. Как осуществляется аннотирование переменных в языке Python?**

В Python переменные можно аннотировать с использованием синтаксиса аннотаций типов. Это позволяет указать ожидаемый тип данных для переменной, хотя интерпретатор Python не выполняет никакой проверки типов во время выполнения.

### **7. Для чего нужна отложенная аннотация в языке Python?**

Отложенная аннотация в Python (Delayed Evaluation Annotation) позволяет создавать аннотации типов, используя строковые литералы вместо ссылок на фактические классы. Это может быть полезно в случаях, когда требуется аннотировать типы данных, которые еще не определены или

недоступны в момент написания аннотации.

Отложенные аннотации могут быть полезны при работе с циклическими зависимостями между классами или модулями, при использовании динамически загружаемых модулей или при аннотации типов в коде, который будет выполняться на разных версиях Python.