

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5
дисциплины
«Объектно-ориентированное программирование»
Вариант 15

Выполнил:
Степанов Леонид Викторович
3 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)

Проверил:
Богданов Сергей Сергеевич,
Ассистент департамента цифровых,
робототехнических систем и
электроники

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

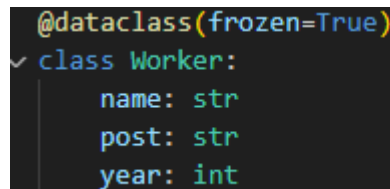
Ставрополь, 2024 г.

Тема: Аннотация типов

Цель: 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, регламентирующих работу с аннотациями, и представлены примеры работы с инструментом mypy для анализа Python кода.

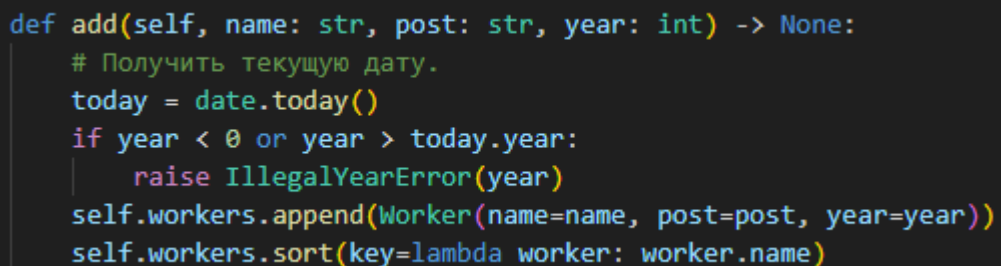
Порядок выполнения работы:

Пример 1: Для примера 1 лабораторной работы 14 добавьте аннотации типов.



```
@dataclass(frozen=True)
class Worker:
    name: str
    post: str
    year: int
```

Рисунок 1 – Аннотации типов класса



```
def add(self, name: str, post: str, year: int) -> None:
    # Получить текущую дату.
    today = date.today()
    if year < 0 or year > today.year:
        raise IllegalYearError(year)
    self.workers.append(Worker(name=name, post=post, year=year))
    self.workers.sort(key=lambda worker: worker.name)
```

Рисунок 2 Аннотации типов в функции

На рисунках 1-2 отображено использование аннотации типов, на рисунке 1 для класса задаются определенные типы данных, а на рисунке 2 входным переменным аннотируют типы и возвращает эта функция None

Индивидуальные задания

Выполнить индивидуальное задание 2 лабораторной работы 2.19, добавив аннотации типов. Выполнить проверку программы с помощью утилиты mypy.

Задание 2 задания лабораторной работы 2.19:

Разработайте аналог утилиты tree в Linux. Используйте возможности модуля argparse для управления отображением дерева каталогов файловой системы. Добавьте дополнительные уникальные возможности в данный программный продукт.

```
class FileLister:
    def __init__(self, path: Path, show_time: bool, show_size: bool, only_dir: bool, max_level: int):
        self.path = path
        self.show_time = show_time
        self.show_size = show_size
        self.only_dir = only_dir
        self.max_level = max_level
```

Рисунок 3 Аннотация типов в конструкторе

```
class FileLister:
    def __init__(self, path: Path, show_time: bool, show_size: bool,
    def list_files(self, level: int = 0) -> None: ...
    def print_file_info(self, item: Path, indent: str) -> None: ...
    def print_dir_info(self, item: Path, indent: str) -> None: ...
    def get_size(self, item: Path) -> Optional[int]: ...
    def get_time(self, item: Path) -> Optional[datetime]: ...
```

Рисунок 4 Аннотация типов аргументов и возвращаемых значений
методов

```
(base) D:\Python-labs\lab4.5>python -m mypy .\prog\ind.py
Success: no issues found in 1 source file

(base) D:\Python-labs\lab4.5>
```

Рисунок 5 Результат выполнения mypy

Ссылка на github: <https://github.com/FiaLDI/lab4.5>

Контрольные вопросы

1. Для чего нужны аннотации типов в языке Python?

Аннотации типов в Python служат для указания ожидаемых типов данных для параметров функций и возвращаемых значений. Это помогает улучшить читаемость кода и облегчает его поддержку, позволяя

разработчикам быстрее понимать, какие типы данных используются в функции.

2. Как осуществляется контроль типов в языке Python?

Контроль типов в Python осуществляется неявно, так как язык является динамически типизированным. Это означает, что типы переменных проверяются во время выполнения, а не на этапе компиляции.

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

Существуют различные предложения по усовершенствованию Python, включая улучшение поддержки отложенных аннотаций и расширение возможностей модуля `typing`

4. Как осуществляется аннотирование параметров и возвращаемых значений функций?

Аннотирование параметров и возвращаемых значений функций осуществляется с помощью синтаксиса, который включает указание типа после имени параметра и после стрелки `->` для возвращаемого значения. Например: `def add(a: int, b: int) -> int: return a + b`

5. Как выполнить доступ к аннотациям функций?

Доступ к аннотациям функций можно получить через атрибут `__annotations__`. Этот атрибут возвращает словарь, где ключами являются имена параметров, а значениями — аннотированные типы.

6. Как осуществляется аннотирование переменных в языке Python?

Аннотирование переменных в Python можно осуществить с помощью синтаксиса, аналогичного аннотированию параметров функций. Например: `x: int = 10, name: str = "Alice"`

7. Для чего нужна отложенная аннотация в языке Python?

Отложенная аннотация в Python позволяет ссылаться на типы, которые еще не определены, что особенно полезно в случаях, когда типы ссылаются друг на друга. Это достигается с помощью строки, содержащей имя типа, вместо непосредственного указания типа.

Вывод: в ходе выполнения работы были приобретены навыки по работе с аннотациями типов при написании программ с помощью языка программирования Python версии 3.x., был рассмотрен вопрос контроля типов переменных и функций с использованием комментариев и аннотаций, приведено описание PEP, регламентирующих работу с аннотациями, и представлены примеры работы с инструментом mypy для анализа Python кода.