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»

Выполнил:
Кожуховский Виктор Андреевич
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:
Воронкин Роман Александрович

_____
(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‘ов, регламентирующих работу с аннотациями, и представлены примеры работы с инструментом mypy для анализа Python кода

Порядок выполнения работы:

1. Изучил теоретический материал работы.

2. Создал общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python.

3. Выполнил клонирование созданного репозитория.

4. Дополнил файл .gitignore необходимыми правилами для работы с IDE.

5. Организовал свой репозиторий в соответствие с моделью ветвления git-flow.

6. Создал проект в папке репозитория.

7. Проработал примеры лабораторной работы.

8. Выполнил индивидуальное задание для варианта 14.

Задание 1

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

```python
import os
import argparse
import stat


def is_hidden(filepath: str) -> bool:
    if os.path.basename(filepath).startswith('.'):
        return True
    try:
        return bool(os.stat(filepath).st_file_attributes &
                    stat.FILE_ATTRIBUTE_HIDDEN)
    except AttributeError:
        return False


def list_files(startpath: str, depth: int = -1, level: int = 0,
               show_hidden: bool = False) -> None:
    if depth == 0:
        return
    if level == 0:
        print(startpath)

    for element in os.listdir(startpath):
        path = os.path.join(startpath, element)
        indent = ' ' * 4 * level
        if (element.startswith('.') or is_hidden(path)) and not show_hidden:
            continue
        if os.path.isdir(path):
            print(f'{indent}└── {element}/')
            list_files(path, depth-1, level+1, show_hidden)
        else:
            if element.startswith('.') or is_hidden(path):
                # Скрытые файлы будут отображаться на сером фоне
                print(f'{indent}└── \033[100m{element}\033[0m')
            else:
                print(f'{indent}└── {element}')


def main() -> None:
    parser = argparse.ArgumentParser(description='Display directory tree')
    parser.add_argument('path', nargs='?', default='.', help='Directory path')
    parser.add_argument('-d', '--depth', type=int,
                        default=-1, help='Depth of the tree')
    parser.add_argument('--hidden', action='store_true',
                        help='Show hidden files')
    args = parser.parse_args()

    list_files(args.path, args.depth, show_hidden=args.hidden)


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

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  GITLENS  COMMENTS

PS C:\Users\viktor\Desktop\ncfu\OOP\OOP-5> & "C:/Program Files/Python311/python.exe" c:/Use
C:\Users\viktor\Desktop\ncfu\OOP\OOP-5
└── doc/
    └── Лр 5 Кожуховский.docx
└── example_1.py
└── LICENSE
└── main.py
└── pyproject.toml
└── README.md
PS C:\Users\viktor\Desktop\ncfu\OOP\OOP-5> python -m mypy main.py
Success: no issues found in 1 source file
```

Рисунок 1. Код решения индивидуального задания и его выполнение

9. Зафиксировал сделанные изменения в репозитории.

10. Выполнил слияние ветки для разработки с веткой master/main.

11. Отправил сделанные изменения на сервер GitHub.

Ссылка: https://github.com/Viktorkozh/OOP-5

Контрольные вопросы:

1. Для чего нужны аннотации типов в языке Python?

Чтобы повысить информативность исходного кода и иметь возможность с помощью сторонних инструментов производить его анализ. Они позволяют контролировать типы переменных и функций.

2. Как осуществляется контроль типоа в язяке Python?

Контроль типов в языке Python осуществляется с использованием комментариев, составленных определенным образом, а также с помощью специального инструмента mypy, который выполняет соответствующую проверку типов.

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

PEP 3107 (Function Annotations), PEP 484 (Type Hints), PEP 526 (Syntax for Variable Annotations) и PEP 563 (Postponed Evaluation of Annotations).

4. Как осуществляется аннотирование параметров и возвращаемых значений функций?

Путем указания типов аргументов через двоеточие после их имени и типа возвращаемого значения с использованием символов "->" после имени функции.

5. Как выполнить доступ к аннотациям функций?

Через атрибут annotations, в котором аннотации представлены в виде словаря, где ключами являются атрибуты, а значениями – аннотации.

6. Как осуществляется аннотирование переменных в языке Python?

Через комментарии в формате # type: type_name, через синтаксис var: annotation и через синтаксис var: annotation = value.

7. Для чего нужна отложенная аннотация в языке Python?

Для решения проблем, связанных с тем, что определение типов переменных происходит во время импорта модуля. Это позволяет определять переменные до получения информации об их типах и ускоряет выполнение программы, так как при загрузке модулей не тратится время на проверку типов.

Вывод: приобретены навыки 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 mypy для анализа Python кода.