

**МИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ И МАССОВЫХ
КОММУНИКАЦИЙ**

**Ордена Трудового Красного Знамени
Федеральное государственное бюджетное общеобразовательное
учреждение высшего образования**

**“Московский технический университет связи и информатики”
(МТУСИ)**

Кафедра “Программная инженерия”

Лабораторная работа №5
по дисциплине “Введение в информационные технологии”

Выполнил: Студент группы
БПИ2503
Яричевский Даниил

Москва
2025

Цель работы

Получить практический опыт работы с ООП в Python.

Задачи

1. Базовый класс и методы
2. Работа с конструктором

Ход работы

1.1 Определяем класс Book и создаем три атрибута: title (название), author (автор), и year (год издания).

```
1 class Book:
2     def __init__(self, title, author, year):
3         self.title = title
4         self.author = author
5         self.year = year
```

Рис. (1.1)

1.2 Добавляем метод get_info(), который возвращает информацию о книге в формате: "Название книги: [title], Автор: [author], Год издания: [year]". Создаем переменную My_Book в которую через класс Book записываем наши параметры. После, с помощью метода get_info() выводим наши параметры.

```
1 class Book:
2     def __init__(self, title, author, year):
3         self.title = title
4         self.author = author
5         self.year = year
6
7     def get_info(self):
8         print(f"Название книги: [{self.title}], Автор: [{self.author}], Год издания: [{self.year}]")
9
10 My_Book = Book("My Book", "Me", 2025)
11
12 My_Book.get_info()
```

Рис. (1.2)

2.1 Определяем класс Circle для представления круга.

```
1 class Circle:
```

Рис. (2.1)

2.2 Используем конструктор __init__ для инициализации радиуса круга (radius).

```
1 class Circle:
2     def __init__(self, radius):
3         self.radius = radius
```

Рис. (2.2)

2.3 Добавляем метод get_radius(), который возвращает значение радиуса.

```
1 class Circle:
2     def __init__(self, radius):
3         self.radius = radius
4
5     def get_radius(self):
6         print(f'Радиус: {self.radius}')
```

Рис. (2.3)

2.4 Добавляем метод set_radius(new_radius), который позволяет изменить значение радиуса.

```
1 class Circle:
2     def __init__(self, radius):
3         self.radius = radius
4
5     def get_radius(self):
6         print(f'Радиус: {self.radius}')
7
8     def set_radius(self, new_radius):
9         self.radius = new_radius
```

Рис. (2.4)

2.5 Создаем объект класса Circle, изменяем его радиус и выводим новый радиус на экран.

```
1 class Circle:
2     def __init__(self, radius):
3         self.radius = radius
4
5     def get_radius(self):
6         print(f'Радиус: {self.radius}')
7
8     def set_radius(self, new_radius):
9         self.radius = new_radius
10
11 Circle_A = Circle(15)
12 Circle_A.set_radius(100)
13 Circle_A.get_radius()
```

Рис. (2.5)

Вывод

Я получил практический опыт работы с ООП в Python.