

Python-разработчик

# Наследование



# Сегодня на уроке



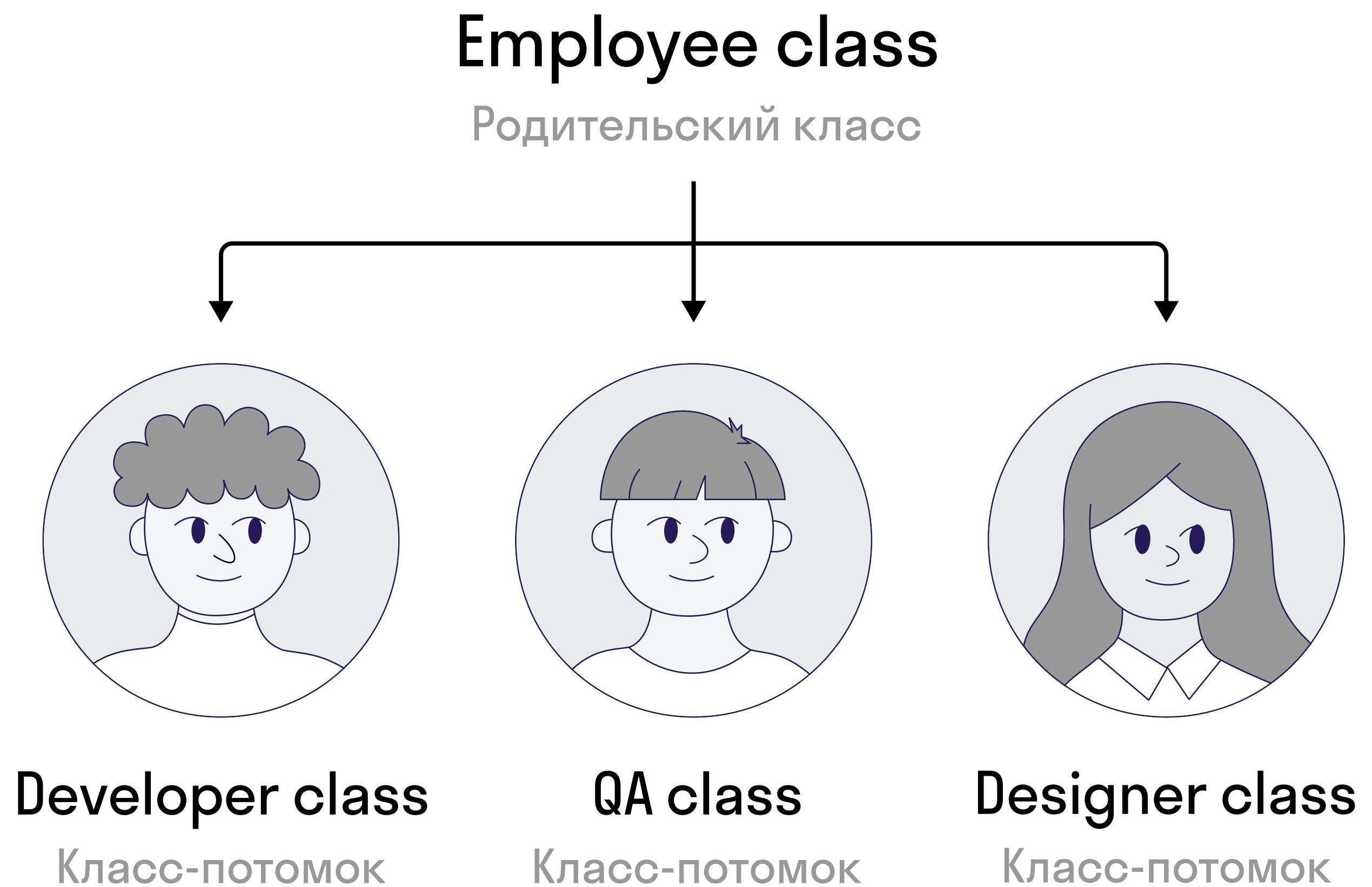
# План урока

1. Рассмотрим, что такое наследование в Python.
2. Познакомимся с функцией `super()`  
для обращения к родительским классам.
3. Узнаем, как работают функции `issubclass()`  
и `isinstance()`.

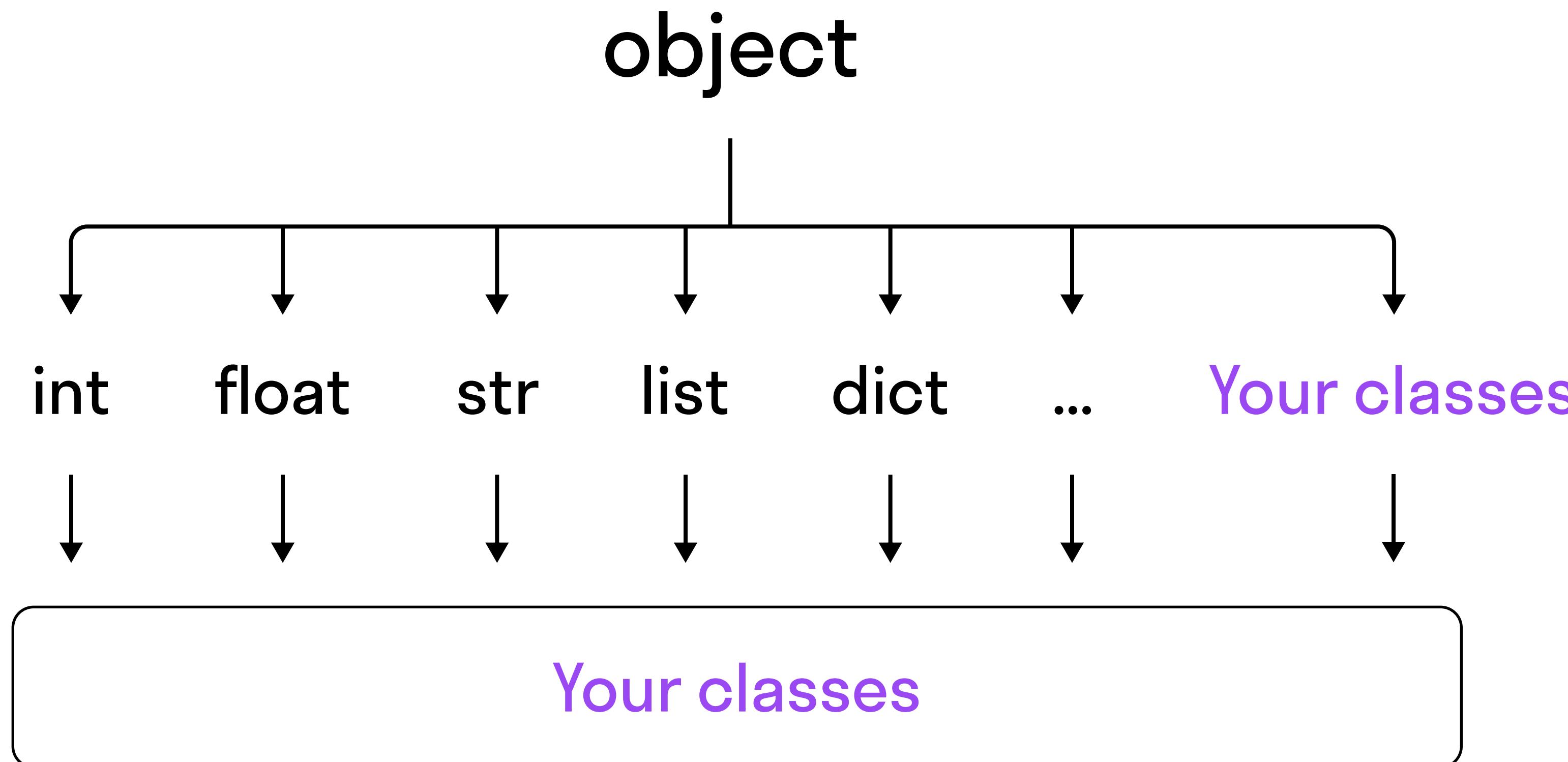
# Наследование

**Наследование** — механизм ООП, позволяющий описать новый класс на основе уже существующего (родительского), при этом свойства и функциональность родительского класса заимствуются новым классом.

определение



# Наследование



# Задача

- Создать класс для разработчиков, установить ставку индексации ЗП 10%.
- Создать двух разработчиков.

# Задача

Создание класса Developer.

Флоу решения:

- 1 Создать класс, наследуемый от базового
- 2 Переопределить атрибут класса
- 3 Создать два экземпляра класса

# Функция super()

Функция `super()` возвращает объект-посредник, который делегирует вызовы метода родительскому или родственному классу.

Позволяет **получать доступ из класса наследника к методам класса-родителя** в том случае, если наследник переопределил эти методы.

```
class Employee:
```

```
    def work(self):
```

```
        print('Do some work')
```

```
class Developer(Employee):
```

```
    def work(self):
```

```
        super().work()
```

```
        print('Write code')
```

```
class JavaDeveloper(Developer):
```

```
    def work(self):
```

```
        super().work()
```

```
        print('Write tests for code')
```

# Расширение и переопределение

Расширение базового класса —  
добавление новых атрибутов в дочерних классах.

определение

Переопределение базового класса —  
изменение поведения уже существующего  
функционала.

определение

# Расширение

```
class Employee:  
  
    def go_to_vacation(self):  
        print('Go to vacation')  
  
class Accountant(Employee):  
  
    def go_to_vacation(self):  
        """ Расширение """  
        print('Pass documents')  
        super().go_to_vacation()
```

# Переопределение

```
class Employee:  
  
    def go_to_vacation(self):  
        print('Go to vacation')  
  
class Developer(Employee):  
  
    def go_to_vacation(self):  
        """ Переопределение """  
        print('Go back to work, no vacation')
```

# Задача

Добавить разработчикам атрибут  
«язык программирования».

# Задача

Добавление атрибута.

Флоу решения:

- 1 Создать инициализатор
- 2 Вызвать super()
- 3 Дописать атрибут prog\_lang

# Функции `issubclass()` и `isinstance()`

Часто используются для проверки допустимости  
(совместимости) операций с объектами.

`issubclass()` — используется  
для проверки, наследуется ли  
какой-либо класс от другого.

`issubclass(A, Base)` —  
возвращает `True`, если класс  
`A` является подклассом  
класса `Base`.

`isinstance()` — используется  
для проверки принадлежности  
экземпляра к классу  
или его родителям.

`isinstance(a, Base)` — возвращает  
`True`, если объект `a` является  
экземпляром класса `Base`  
(также работает для родителей).

# Задача

Реализовать магический метод `add` и добавить проверку на сложение только с другими экземплярами класса `Employee` и дочерних классов.

# Задача

Добавление проверки `isinstance()`

Флоу решения:

- 1 Добавить метод `add`
- 2 Использовать `isinstance()`
- 3 Выбрасывать исключение

# Подведем итоги

.Level Up.



# Итоги урока

1. Наследование — механизм ООП, который позволяет описать новый класс на основе уже существующего.

# Итоги урока

1. Наследование — механизм ООП, который позволяет описать новый класс на основе уже существующего.
2. Функция `super()` позволяет вызывать методы из родительского класса.

# Итоги урока

1. Наследование — механизм ООП, который позволяет описать новый класс на основе уже существующего.
2. Функция `super()` позволяет вызывать методы из родительского класса.
3. Функция `issubclass()` проверяет, наследуется ли класс от указанного класса.

# Итоги урока

1. Наследование — механизм ООП, который позволяет описать новый класс на основе уже существующего.
2. Функция `super()` позволяет вызывать методы из родительского класса.
3. Функция `issubclass()` проверяет, наследуется ли класс от указанного класса.
4. Функция `isinstance()` проверяет, принадлежит ли экземпляр класса указанному классу и его родителям.

# Спасибо!

