Overloading

February 21, 2024

1 Method overloading:

When inheriting from other classes we can overwrite inherited methods, or overload them (modifying their behaviour).

[12]:
```python
class Parent:
    def give_feedback(self):
        return "Well done!"

# Overwriting
class ChildOne(Parent):
    def give_feedback(self):
        return "What do I care?"

# Overloading
class ChildTwo(Parent):
    def give_feedback(self):
        msg = super().give_feedback()
        return f"{msg} I suppose."

child1 = ChildOne()
print(child1.give_feedback())

child2 = ChildTwo()
print(child2.give_feedback())
```

```
What do I care?
Well done! I suppose.
```

[13]:
```python
# Overloading with arguments:

class A:
    def __init__(self, name):
        self.name = name

    def say_hi(self, mood):
        if mood == "happy":
            return f"Hello, my name is {self.name}. What's your name?"
        else:
            return "Leave me alone!"

class B(A):
    def __init__(self, name, age):
        super().__init__(name)
        self.age = age
    def say_hi(self, mood):
        if mood == "happy":
            return super().say_hi(mood)
        elif mood == "sad":
            return f"With my {self.age}, I am too old for this."
        else:
            return super().say_hi(mood)

b = B("Simon", 33)
print(b.name, b.age)
print(b.say_hi("happy"))
print(b.say_hi("grumpy"))
print(b.say_hi("sad"))
```

```
Simon 33
Hello, my name is Simon. What's your name?
Leave me alone!
With my 33, I am too old for this.
```

[ ]: