# Python面向对象 教案

周次：3

课次：1

章节内容：类与对象

课时：2

## 单元教学目标

- 知识目标：能够正确使用类的初始化方法（构造函数）和属性访问器/setter方法   
- 技能目标：能够编写一个简单的类，并实例化对象，调用其方法   
- 素养目标：能够遵循封装原则，合理设计类的属性和方法

## 教学重点

• 类的初始化方法\_\_init\_\_：用于设置对象的属性值，是面向对象编程的核心概念   
• 对象的属性与方法：包括属性的赋值（\_\_setattr\_\_）和获取（\_\_getattr\_\_），以及方法的定义与调用   
• 继承与多态：通过继承构建类的层次结构，通过方法重写实现多态性，是面向对象编程的进阶特性

## 教学难点

• 类与对象的概念易混淆：学生可能将“类”理解为“对象的模板”，而忽略其本质是抽象数据类型，难以区分“类”与“实例”的区别，导致在创建类和实例时出现逻辑错误。   
  
• 继承中的方法覆盖难点：学生可能在继承时误以为父类方法可以直接复用，忽略需通过`\_\_init\_\_`或`\_\_setattr\_\_`等方法实现方法重写，导致程序行为与预期不符。

## 教学活动

新课导入   
时间：5分钟   
方法：故事导入 + 案例分析   
内容：以“超市管理系统”为情境，讲述小明设计商品类（属性：名称、价格，方法：显示信息）解决实际问题的过程，激发学生兴趣。通过提问“如何用代码描述商品？”引导学生思考，为后续学习铺垫。  
  
讲授新课   
时间：20分钟   
方法：讲授法 + 案例分析 + 头脑风暴 + 分组讨论   
内容：   
1. \*\*概念讲解\*\*（5分钟）   
 - 讲授类（模板）与对象（实例）定义，强调“类是模板，对象是实例”。   
 - 举例：`class Animal: ...`，引导学生写出类的结构。   
  
2. \*\*案例分析\*\*（7分钟）   
 - 分组分析“动物类”案例：   
 - 属性：名称、种类、年龄   
 - 方法：`def speak(self): print("动物叫")`   
 - 学生讨论如何设计类的属性和方法，教师点评重点。   
  
3. \*\*互动练习\*\*（6分钟）   
 - 学生尝试编写“水果类”代码，教师巡视指导，重点强调属性命名规范（如`\_\_color`）和方法定义。   
  
4. \*\*总结归纳\*\*（3分钟）   
 - 教师总结：类的结构包括属性和方法，对象是类的实例。   
 - 学生用“角色扮演”复述内容，教师提问“如何用类描述学生？”强化理解。  
  
巩固练习   
时间：10分钟   
方法：分层练习 + 分组协作   
内容：   
1. \*\*基础练习\*\*（3分钟）   
 - 学生编写“汽车类”代码，要求包含属性`color`和`speed`，方法`start()`。   
  
2. \*\*中等练习\*\*（4分钟）   
 - 学生尝试添加`\_\_init\_\_`方法，实现初始化属性，教师示范并纠正错误。   
  
3. \*\*高级练习\*\*（3分钟）   
 - 学生设计“继承关系”（如`Dog`继承`Animal`），编写`def bark(self)`: `print("汪!")`，教师点评重点。   
  
归纳总结   
时间：5分钟   
方法：教师总结 + 学生复述   
内容：   
1. 教师总结：类的定义、对象创建、方法使用、继承概念。   
2. 学生用“头脑风暴”复述课堂内容，教师提问“如何用类描述老师？”引导学生巩固知识。

## 教学资源

• 教学设备和工具   
 - 投影仪/白板（用于展示代码示例和教学内容）   
 - 多媒体设备（支持播放教学视频或动画）   
 - 电脑/平板（用于运行Python代码、演示交互式编程）   
 - 开发工具：Python解释器（如IDLE、PyCharm、VS Code）   
 - 交互式白板（支持实时代码编写与展示）   
  
• 推荐的参考资料或网站   
 - 书籍：   
 - 《Python编程：从入门到实践》（Eric Matthes）   
 - 《Python核心技术》（David Beazley）   
 - 在线资源：   
 - Python官方文档（https://docs.python.org/3/）   
 - LeetCode（练习编程题，结合面向对象概念）   
 - GitHub（搜索“Python OOP”相关开源项目）   
 - 教学平台：   
 - Coursera《Python for Everybody》课程   
 - edX《Python for Everybody》课程   
 - 练习案例：   
 - 《Python编程：从入门到实践》中的“类与对象”章节练习   
 - 网易云课堂《Python基础教程》（含面向对象部分）   
 - 知乎专栏《Python实战》（结合案例讲解类与对象）   
  
• 其他补充资源   
 - 代码示例：   
 - 《Python核心编程》中的经典类示例（如Student、Animal）   
 - GitHub开源项目（如“Python-OOP-Examples”）   
 - 互动练习：   
 - Python Tutor（在线调试工具，可视化类与对象结构）   
 - Codecademy（交互式编程练习，结合OOP概念）

## 教学反思

• 教学效果：学生对类的定义和对象的创建理解较好，但部分学生在继承和多态概念上存在困惑，需加强抽象概念的引导   
• 学生反馈：部分学生反映抽象类和继承的示例不够直观，难以理解；部分学生在实际编码时出现方法覆盖、初始化参数传递等常见错误   
• 改进建议：   
 • 增加生活化案例（如动物类、交通工具类）强化抽象概念的理解   
 • 引入代码模板（如使用类模板或工厂模式）降低实践门槛   
 • 设计分组讨论任务，让学生通过实际问题（如设计共享单车系统）巩固类与对象的知识   
 • 增设即时反馈环节，通过代码提交检测学生对初始化方法、属性访问的掌握情况   
 • 提供阶梯式练习题（从简单类到复杂继承结构）确保不同水平学生都能参与实践   
• 教学策略：采用“问题驱动+实例演示+小组合作”模式，结合代码调试工具（如PyCharm）提升实践体验   
• 课堂互动：增加实时问答环节，针对学生常见错误（如类方法与实例方法混淆）进行针对性讲解   
• 评估反馈：通过代码提交与课堂提问结合，量化学生对继承、多态等难点的掌握程度

## 教学评价

• 过程性评价   
• 1. 课堂参与度：教师观察记录学生是否积极回答问题、参与讨论，记录学生在小组活动中的表现（如发言频率、协作能力）。   
• 2. 作业完成情况：批改学生提交的代码作业，评估逻辑是否清晰、语法是否正确，记录错误类型及改正情况。   
• 3. 小组合作表现：通过小组任务完成情况（如任务分工是否合理、协作是否高效）评价学生团队意识和沟通能力。   
  
• 结果性评价   
• 1. 课堂测试：通过随堂小测验（如类定义、对象初始化、方法调用）评估学生对基础概念的掌握程度，满分10分。   
• 2. 项目完成度：根据学生提交的类与对象项目（如“动物饲养员”类）评分，重点考察代码结构、功能实现及文档规范性（满分10分）。   
• 3. 作业质量：批改学生提交的代码作业，综合逻辑性、语法准确性及创新性（满分10分）。   
  
• 评价标准与方式   
• 1. 课堂参与度：满分10分，权重30%。   
• 2. 作业完成情况：满分10分，权重30%。   
• 3. 项目完成度：满分10分，权重40%。   
• 4. 课堂测试/作业评分：满分10分，权重10%。   
  
• 评价工具与记录   
• 1. 课堂观察记录表（含学生发言、协作行为、代码规范等）   
• 2. 作业批改评分表（含逻辑性、语法、创新性）   
• 3. 项目评审表（含功能完整性、代码结构、文档说明）