人工智能程序设计


python
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
turtle.circle(40, 80)
turtle.circle(-40, 80)
turtle.fd(40)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)



人工智能程序设计

第6章 类与面向对象程序设计

北京石油化工学院 人工智能研究院

刘 强

面向对象程序设计

面向对象程序设计/面向对象编程是现代软件开发的核心思想之一。它通过类和对象的概念，将数据和操作数据的方法组织在一起，实现了代码的模块化、重用性和可维护性。

面向过程编程与面向对象编程的差异

面向过程编程将程序看作一系列函数和数据的组合，函数处理数据，数据与操作分离。随着程序规模增大，函数之间的依赖关系变得复杂，代码重用困难，修改一处可能影响多处，维护成本高。

面向对象编程将数据和操作数据的方法封装在一起，形成**对象**。

通过**类**来定义**对象**的模板，通过继承实现代码重用，通过多态实现灵活的程序设计。这种方式更符合人类对现实世界的认知，使程序结构更清晰、更易维护。

6.1 类的定义与实例化

类（Class）是面向对象编程的核心概念，通过类，我们可以将数据和操作数据的方法组织在一起，实现代码的模块化和重用。

类是对象的蓝图或模板，定义了对象应该具有的属性和方法。

对象（Object）是类的实例，是具体的数据实体。

6.1 类的定义与实例化

类与对象的核心概念：

1. 类：定义对象的结构和行为的模板，是创建对象的蓝图
2. 对象：类的具体实例，拥有类定义的属性和方法
3. 属性：对象的数据特征，用于存储对象的状态信息
4. 方法：对象的行为或功能，定义对象可以执行的操作

6.1.1 创建类和对象

定义类的基本语法

类的定义使用 class关键字开头，后跟类名和冒号。类的内部结构包括：

- __init__方法：构造方法，在创建对象时自动调用，用于初始化对象的属性
- 实例方法：定义对象的行为，第一个参数必须是 self，代表对象本身
- 文档字符串：用三引号包围，描述类的功能和用途

示例 6.1.1：学生信息管理类

通过 类名(参数)的方式可以创建对象实例，每个实例都拥有独立的属性值。

```
class Student:
    """学生类的简单示例"""
    def __init__(self, name, age):
        """初始化方法"""
        self.name = name
        self.age = age
    def introduce(self):
        """自我介绍方法"""
        return f"我是{self.name}，今年{self.age}岁"
## 创建对象（实例化）
student1 = Student("张三", 18)
student2 = Student("李四", 19)
## 使用对象的方法
print(student1.introduce())  # 输出：我是张三，今年18岁
print(student2.introduce())  # 输出：我是李四，今年19岁
```

示例 6.1.1：学生信息管理类

语法要点

- class关键字用于定义类
- __init__方法是构造方法，用于初始化对象
- self参数代表对象本身，必须作为方法的第一个参数
- 通过 类名(参数)的方式创建对象

6.1.2 类和方法的命名规范

类的命名规范

使用大驼峰命名法（PascalCase），每个单词首字母大写，不使用下划线。

```
class Student:          # 正确
class BankAccount:      # 正确
class student:          # 错误：应该首字母大写
class bank_account:     # 错误：应该使用大驼峰
```

6.1.2 类和方法的命名规范

方法的命名规范

使用小写字母和下划线，方法名应该是动词或动词短语。

```
def add_numbers(self, a, b):        # 好：动词+名词
def calculate_average(self):        # 好：动词+名词
def is_positive(self, number):      # 好：is开头的判断方法
```

6.1.2 类和方法的命名规范

文档字符串

类和重要方法应该包含文档字符串，说明功能和用法。

```
class Rectangle:
    """矩形类，用于计算面积和周长。"""

    def __init__(self, width, height):
        """初始化矩形，设置宽度和高度。"""
        self.width = width
        self.height = height

    def calculate_area(self):
        """计算并返回矩形面积。"""
        return self.width * self.height
```

实践练习

练习 6.1.1：宠物类设计

设计一个Pet类，包含宠物名称、种类、年龄属性，以及显示宠物信息的方法。

练习 6.1.2：手机类

创建一个Phone类，包含品牌、型号、电量属性和充电、打电话方法。

练习 6.1.3：课程类

设计一个Course类，包含课程名称、学分、教师属性，以及显示课程信息和选课方法。