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

4.3 WHILE循环语句

北京石油化工学院 人工智能研究院

刘 强

4.3 while循环语句

循环语句让程序能够自动重复执行代码块，大大提高了程序的效率和代码的简洁性

Python中有两种主要的循环结构：for循环和while循环

for循环已在第3章3.3节中结合列表遍历进行了详细学习，它特别适合遍历序列或已知循环次数的情况

while循环更加灵活，可以处理循环次数未知的情况

4.3 while循环语句

while循环是Python中的基本循环结构之一，它在指定条件为真时重复执行代码块。while循环特别适用于不知道确切循环次数，但知道循环终止条件的情况。

while循环有三个核心要素：

1. **循环条件**：决定循环是否继续执行的布尔表达式
2. **循环体**：需要重复执行的代码块
3. **循环控制**：使用break和continue语句灵活控制循环流程

4.3.1 基本语法

```
while condition:
        # 循环体
        statement1
        statement2
        ...
```

当condition为True时，循环体会被重复执行；

当condition为False时，循环结束。

4.3.2 简单while循环示例

基础计数循环

```
## 简单的计数循环
count = 1
while count <= 5:
    print(f"这是第{count}次循环")
    count += 1  # 重要：更新循环变量，避免无限循环

print("循环结束")
```

4.3.2 简单while循环示例

用户输入验证

```
## 输入验证示例
password = ""
while password != "python123":
    password = input("请输入密码：")
    if password != "python123":
        print("密码错误，请重新输入")

print("密码正确，欢迎登录！")
```

4.3.3 循环控制语句

break语句：break语句用于立即退出循环：

```
## 使用break提前退出循环
count = 1
while True:  # 无限循环
    print(f"计数：{count}")
    if count >= 3:
        print("达到退出条件，结束循环")
        break
    count += 1

print("循环已结束")
```

4.3.3 循环控制语句

continue语句：continue语句用于跳过当前循环的剩余部分，直接进入下一次循环：

```
## 使用continue跳过特定条件
number = 0
while number < 10:
    number += 1
    if number % 2 == 0:  # 跳过偶数
        continue
    print(f"奇数：{number}")
print("循环结束")
```

4.3.4 嵌套while循环

```
## 嵌套循环：打印乘法表
print("九九乘法表：")
i = 1
while i <= 9:
    j = 1
    while j <= i:
        result = i * j
        print(f"{j}×{i}={result:2d}", end="  ")
        j += 1
    print()  # 换行
    i += 1
```

4.3.3 循环控制语句

break vs continue 对比：

- break：终止整个循环，跳出循环体，执行循环后的代码
- continue：跳过本次循环，直接开始下一次循环判断

形象理解：

- break 就像"彻底退出游戏"，不再玩了
- continue 就像"跳过这一轮"，继续下一轮游戏

4.3.5 Ask AI：探索for循环与while循环

当你想要深入了解不同循环类型的特点和应用场景时，可以向AI助手提出以下问题：

- "for循环和while循环有什么区别？什么时候使用哪种循环？"
- "如何将for循环改写为while循环？有什么注意事项？"
- "在什么情况下while循环比for循环更适合？"
- "如何选择最合适的循环类型来解决特定问题？"
- "循环的性能差异和优化技巧有哪些？"

实践练习

练习 4.3.1：数字累加器

使用while循环编写程序，不断接受用户输入的数字，直到用户输入0为止，然后输出所有数字的总和。

练习 4.3.2：猜数字游戏

编写一个猜数字游戏，程序随机生成1-100之间的数字，用户不断猜测，直到猜对为止，并显示猜测次数。

练习 4.3.3：质数判断

使用while循环编写程序判断一个数是否为质数，并找出指定范围内的所有质数。