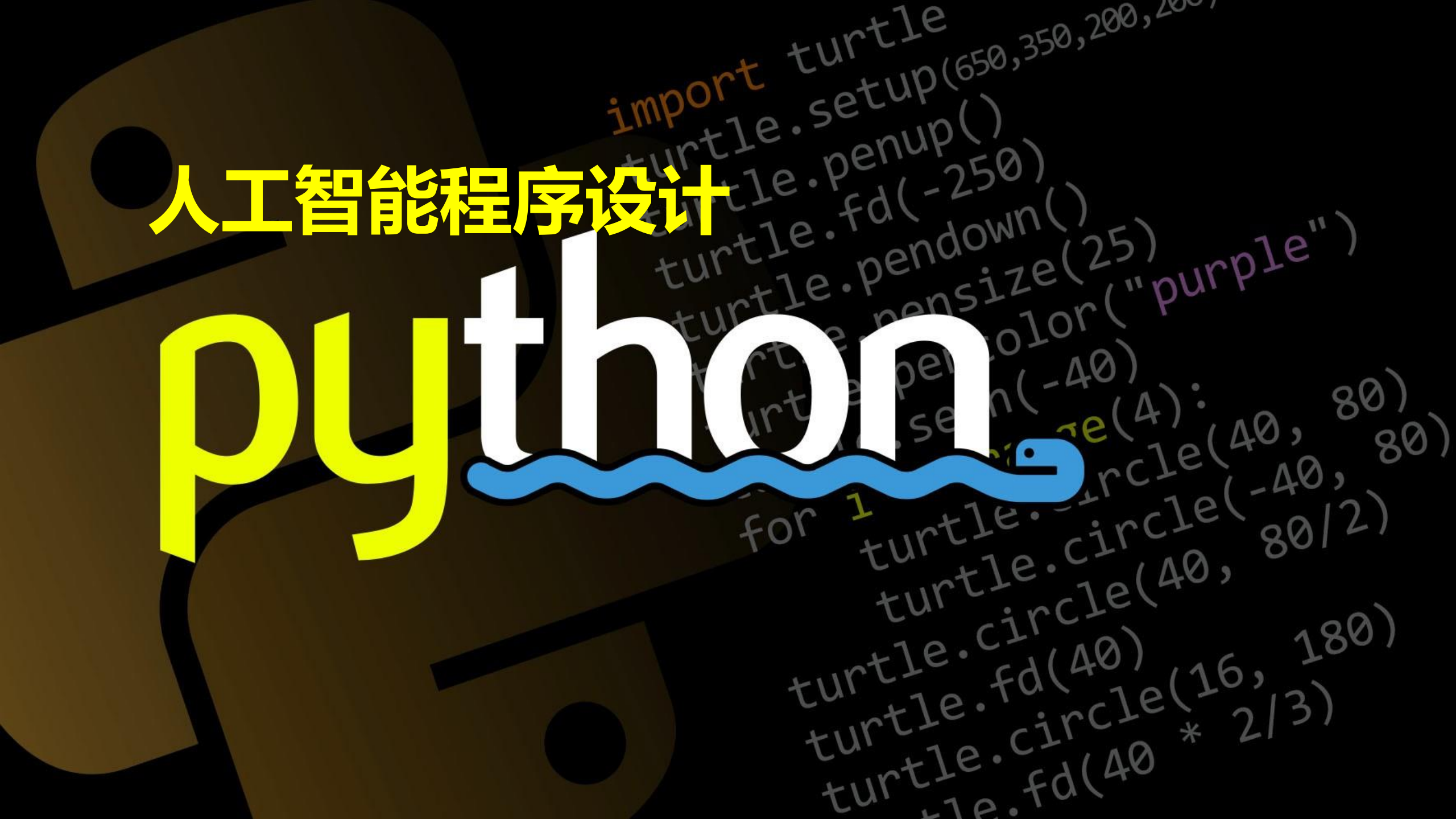
人工智能程序设计


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 1 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

16.2 大模型API应用

北京石油化工学院 人工智能研究院

刘 强

学习内容

- OpenAI API使用方法
- 文本生成与对话实现
- 提示工程核心技巧

16.2.1 OpenAI API使用

学习内容：

- API密钥配置
- 基本API调用方法
- 参数设置说明

API密钥配置

大模型服务通过API接口提供，需要先获取API密钥进行身份认证。

配置步骤：

1. 在平台注册账号（火山引擎、OpenAI等）
2. 在控制台申请API密钥
3. 通过环境变量安全配置到应用中

火山引擎的大模型API兼容OpenAI标准。

安装必要的库

安装用于API调用的Python库：

```
pip install openai requests
```

配置API参数（火山引擎示例）：

```
import openai
import os

# 配置API参数
openai.api_key = os.getenv("VOLC_API_KEY")
openai.api_base = "https://ark.cn-beijing.volces.com/api/v3"
openai.api_type = "open_ai"
```

基本API调用

使用OpenAI兼容格式进行基本的文本生成调用：

```
response = openai.ChatCompletion.create(
    model="your-model-name",  # 替换为实际的模型ID
    messages=[
        {"role": "user", "content": "请介绍Python编程语言的特点"}
    ],
    temperature=0.7,
    max_tokens=1000
)

result = response.choices[0].message.content
print(result)
```

常用参数说明

参数	说明	推荐值
model	模型名称/ID	根据平台选择
messages	对话消息列表	包含role和content
temperature	随机性控制	0.7（创意）/ 0.2（准确）
max_tokens	最大输出长度	根据需求设置

16.2.2 文本生成与对话

学习内容：

- 智能内容生成器
- 文章生成与文本摘要
- 多轮对话实现

示例 16.2.1：智能内容生成器

封装大模型API调用，实现文章生成和文本摘要功能。

```
class ContentGenerator:
    def __init__(self, model: str = "your-model-name"):
        self.model = model

    def generate_article(self, topic: str, style: str = "正式") -> str:
        """生成文章"""
        prompt = """
        请根据以下要求写一篇文章：
        主题：{}
        风格：{}
        要求：结构清晰，内容准确，语言流畅
        """.format(topic, style)
        return simple_chat_completion(prompt)
```

文本摘要生成

为长文本生成简洁摘要：

```
def generate_summary(self, text: str, max_length: int = 200) -> str:
    """生成文本摘要"""
    prompt = "请为以下文本生成{}字以内的摘要：\n\n{}".format(
        max_length, text)
    return simple_chat_completion(prompt)

# 使用示例
generator = ContentGenerator()
article = generator.generate_article("人工智能的发展趋势", "科普")
summary = generator.generate_summary(long_text, max_length=100)
```

16.2.3 提示工程技巧

学习内容：

- 提示设计核心原则
- 思维链提示
- 角色扮演与少样本学习

提示设计核心原则

有效的提示设计需要遵循以下原则：

原则	说明
清晰性	明确任务目标和要求
具体性	提供相关背景信息
结构化	列出具体要求和格式
示例性	通过示例展示期望输出

结构化提示示例

结构化提示能够帮助模型更准确地理解任务需求。

```
任务：将以下英文翻译成中文
背景：这是一篇技术文档的摘要
要求：
1. 保持技术术语的准确性
2. 语言要简洁明了
3. 保持原文的逻辑结构

请翻译：Machine learning is a subset of artificial intelligence.
```

思维链提示

思维链提示模型逐步分析问题，提高复杂问题的解决质量。

特别适用于数学计算和逻辑推理任务。

```
问题：如何优化一个Python程序的性能？

请按照以下步骤思考：
1. 分析问题的关键信息
2. 确定解决思路
3. 逐步推理
4. 得出最终答案

请详细展示思考过程：
```

角色扮演提示

让模型扮演特定角色，以相应的专业水准回答问题。

```
role_prompt = """
你是一位资深Python开发工程师，具有以下专业知识：
性能优化、代码架构、最佳实践。

请以Python专家的身份回答：如何提高Python代码的执行效率？

要求：
1. 体现专业水准
2. 使用相关技术术语
3. 提供实用的建议
"""

answer = simple_chat_completion(role_prompt)
```

少样本学习

通过提供示例来指导模型行为，帮助模型理解任务要求。

```
请按照以下格式分析代码功能：

示例1：
代码：print("Hello World")
功能：输出字符串到控制台
用途：程序调试、信息显示

示例2：
代码：x = [1, 2, 3]
功能：创建包含三个整数的列表
用途：数据存储、序列操作

现在请分析：def add(a, b): return a + b
```

16.2.4 Ask AI：高级应用自学

想要学习更多API高级功能，可以向AI助手询问：

- "如何实现流式响应处理？"
- "怎样优化批量API调用？"
- "如何构建多轮对话系统？"

通过与AI对话，探索更多高级应用技巧。

实践练习

练习 16.2.1：API调用封装

将本章学到的API调用方法封装成一个Python类，实现统一的接口管理和错误处理。

实践练习

练习 16.2.2：提示模板库

基于16.2.3的提示工程技巧，创建一个提示模板库，包含不同类型任务的标准化提示模板。

实践练习

练习 16.2.3：批量内容处理

使用本章的内容生成工具，实现一个批量处理系统，能够同时处理多个文档的摘要生成或翻译任务。