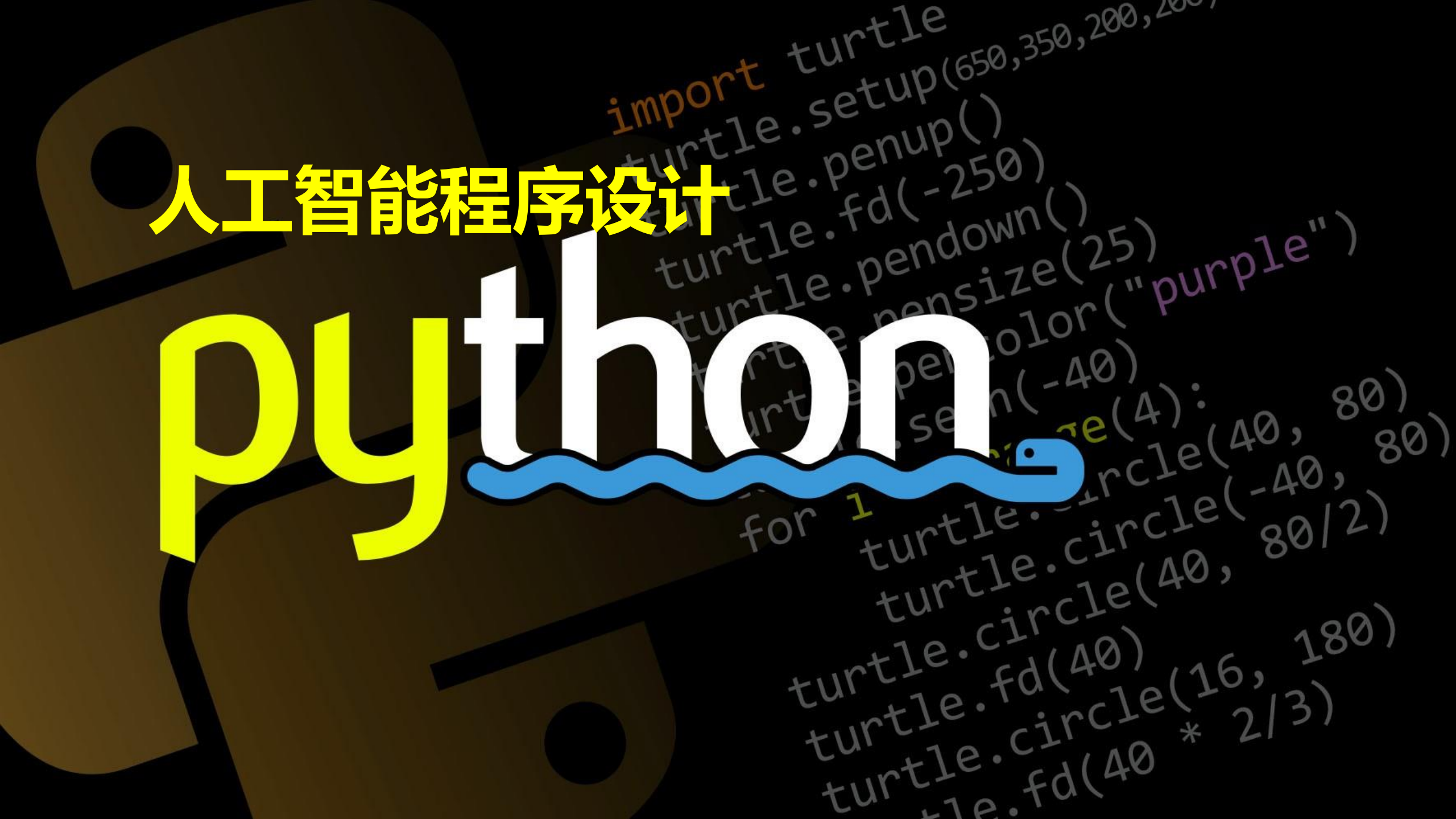
人工智能程序设计


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

15.5 语音应用实践

北京石油化工学院 人工智能研究院

刘 强

学习内容

- 智能语音助手开发
- 语音会议转录系统
- 语音技术前沿探索

语音应用实践概述

通过构建完整的语音应用项目，将语音识别、语音合成、语音交互等技术有机结合，开发实用的语音处理系统。

本节通过智能语音助手和会议转录系统两个典型项目，展示语音技术的实际应用。

15.5.1 智能语音助手开发

学习内容：

- 系统架构设计
- 语音输入输出
- 意图识别与任务执行

系统架构设计

模块化架构智能语音助手包括以下核心模块：

模块	功能
语音输入	采集用户语音
语音识别	语音转文本
自然语言理解	意图识别
任务执行	执行用户指令
语音合成	文本转语音
语音输出	播放语音反馈

语音助手工作流程

工作流程：

用户语音输入 → 唤醒词检测 → 语音识别 → 意图识别 → 任务执行 → 语音合成 → 语音输出

示例 15.5.1：智能语音助手系统

实现语音输入、语音识别、语音合成的基础框架。

```
import speech_recognition as sr
import pyttsx3

# 初始化语音组件
recognizer = sr.Recognizer()
microphone = sr.Microphone()
tts_engine = pyttsx3.init()

# 配置语音参数
tts_engine.setProperty('rate', 150)
tts_engine.setProperty('volume', 0.8)
```

语音输入输出函数

封装语音输出和语音输入功能。

```
# 语音输出
def speak(text):
    tts_engine.say(text)
    tts_engine.runAndWait()

# 语音输入
def listen():
    with microphone as source:
        audio = recognizer.listen(source, timeout=5)
    return recognizer.recognize_google(audio, language='zh-CN')
```

意图识别与任务执行

根据用户语音内容识别意图并执行相应任务。

```
from datetime import datetime

def execute_task(text):
    if '时间' in text or '几点' in text:
        now = datetime.now()
        speak("现在是{}点{}分".format(now.hour, now.minute))
    elif '日期' in text or '几号' in text:
        now = datetime.now()
        speak("今天是{}年{}月{}日".format(now.year, now.month, now.day))
    else:
        speak("抱歉，我还不能理解这个请求")
```

扩展任务执行

添加更多任务类型。

```
import webbrowser

def execute_task(text):
    if '搜索' in text:
        query = text.split('搜索')[1].strip()
        webbrowser.open("https://www.baidu.com/s?wd={}".format(query))
        speak("已为您搜索{}".format(query))
    elif '打开' in text:
        if '浏览器' in text:
            webbrowser.open("https://www.baidu.com")
            speak("已打开浏览器")
```

完整语音助手实现

集成唤醒词检测和任务执行的完整系统。

```
# 语音助手主循环
is_active = False
speak("语音助手已启动")

while True:
    text = listen()
    if not is_active:
        if '助手' in text:
            is_active = True
            speak("我在")
    else:
        if '退出' in text:
            break
        elif '休眠' in text:
            is_active = False
        else:
            execute_task(text)
```

15.5.2 语音会议转录系统

学习内容：

- 系统需求分析
- 核心功能实现
- 会议记录导出

系统需求分析

功能需求：

- 实时语音转录
- 说话人识别
- 关键词提取
- 会议纪要生成
- 多格式导出

性能需求：

- 低延迟转录
- 高识别准确率
- 长时间稳定运行

会议转录核心功能

实现实时语音转录和会议记录。

```
from datetime import datetime

transcriptions = []

def record_and_transcribe():
    with microphone as source:
        audio = recognizer.listen(source, phrase_time_limit=5)

    text = recognizer.recognize_google(audio, language='zh-CN')
    timestamp = datetime.now().strftime("%H:%M:%S")

    transcriptions.append({'time': timestamp, 'text': text})
    print("[{}] {}".format(timestamp, text))
```

会议转录主程序

实现持续录音和异常处理。

```
def run_meeting():
    print("会议开始录制")

    while True:
        try:
            record_and_transcribe()
        except sr.UnknownValueError:
            pass  # 无法识别时继续
        except KeyboardInterrupt:
            break  # Ctrl+C结束会议
```

会议记录导出

将会议记录导出为文本文件。

```
def export_meeting():
    with open("meeting.txt", 'w', encoding='utf-8') as f:
        for entry in transcriptions:
            f.write("[{}] {}\n".format(entry['time'], entry['text']))
    print("会议记录已保存")

# 完整流程
run_meeting()
export_meeting()
```

15.5.3 Ask AI：语音技术前沿

学习内容：

- 前沿技术探索方向
- AI学习建议

前沿技术探索方向

技术方向	说明
多模态语音交互	结合语音、视觉、手势等多种交互方式
个性化语音合成	少量样本快速克隆个人语音特征
实时语音翻译	结合语音识别、机器翻译、语音合成

Ask AI学习建议

向AI助手询问以下进阶内容：

1. **"如何提高嘈杂环境下的语音识别准确率？"**
2. **"Whisper等端到端语音模型的原理和应用？"**
3. **"语音技术在不同行业的创新应用案例"**

实践练习

练习 15.5.1：语音助手扩展

为智能语音助手添加新功能，如邮件发送、日程管理、智能家居控制等。

实践练习

练习 15.5.2：会议系统优化

优化会议转录系统，添加说话人分离、实时字幕显示、自动摘要生成等功能。

实践练习

练习 15.5.3：语音应用集成

将语音技术集成到现有应用中，如为Web应用添加语音搜索功能。

实践练习

练习 15.5.4：性能评估系统

开发语音系统性能评估工具，测量识别准确率、响应延迟、用户满意度等指标。