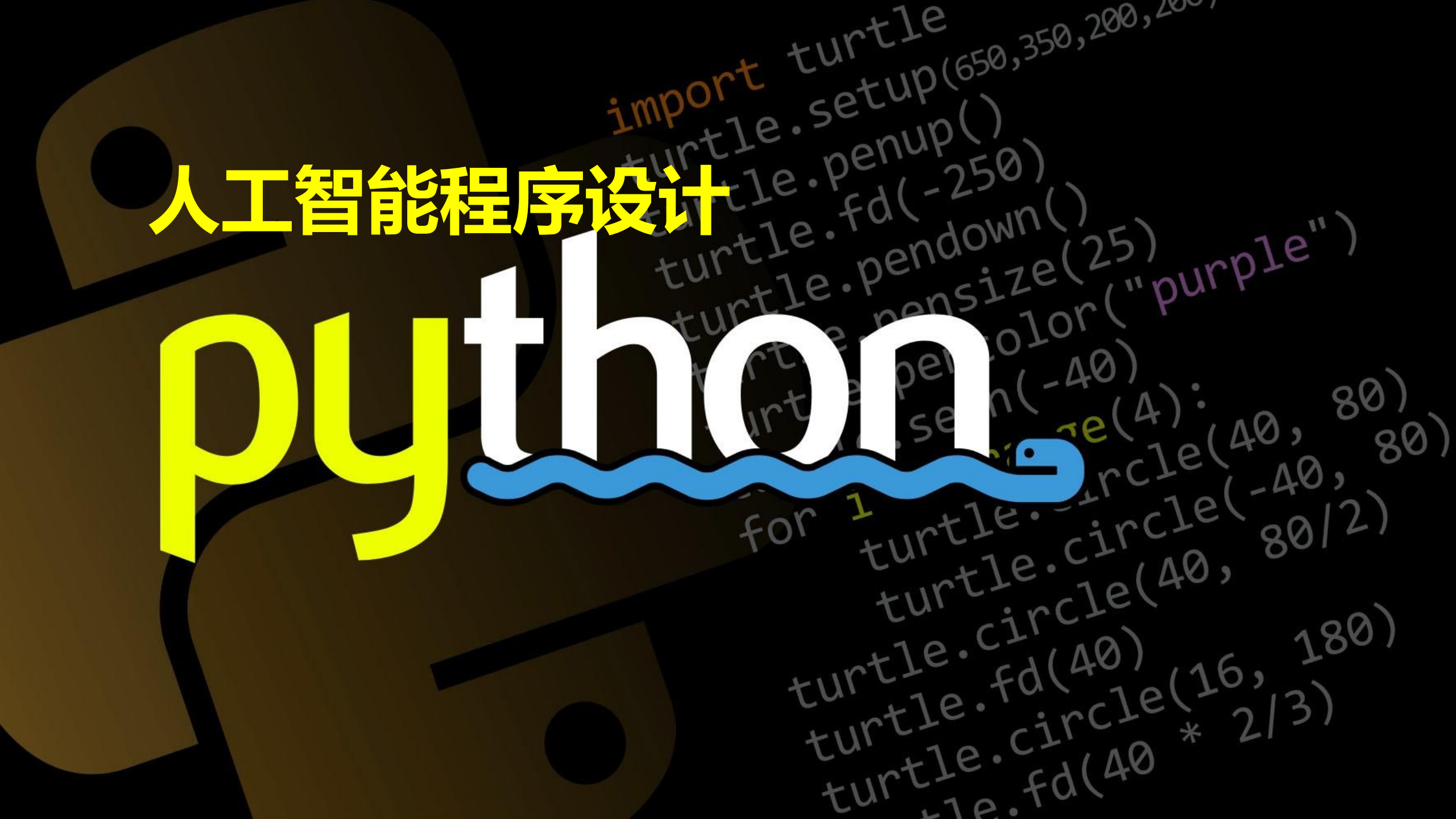
# 人工智能程序设计


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

# 14.2 现代NLP工具与框架

北京石油化工学院 人工智能研究院

刘 强

# NLP工具库概览

| 工具 | 特点 | 适用场景 |
| --- | --- | --- |
| NLTK | 经典、教学友好 | 学习和研究 |
| spaCy | 高性能、现代化 | 生产环境 |
| jieba | 中文分词首选 | 中文处理 |
| HanLP | 多语言、功能全面 | 中文NLP |
| transformers | 预训练模型 | 深度学习NLP |

# 14.2.1 NLTK与spaCy基础应用

NLTK和spaCy是两个最重要的英文NLP工具库：

- **NLTK**：Python中最经典的NLP库，教学和研究首选
- **spaCy**：现代化NLP库，生产环境首选

# NLTK工具库

**NLTK**是Python中最经典的NLP库，提供了：

- 丰富的语言资源（语料库、词典）
- 完整的算法实现
- 优秀的文档和教程

**安装方式：**

```
pip install nltk
```

# NLTK文本分词

使用NLTK进行文本分词和词性标注：

**输出示例：**

- 分词：['Natural', 'language', 'processing', 'is', 'amazing', '!']
- 词性：[('Natural', 'JJ'), ('language', 'NN'), ...]

```
## 文本分词和词性标注
tokens = nltk.word_tokenize("Natural language processing is amazing!")
pos_tags = nltk.pos_tag(tokens)
print(pos_tags)

## 句子分割
sentences = nltk.sent_tokenize("Hello world. This is NLTK.")
print(sentences)
```

# spaCy工具库

**spaCy**是现代化的NLP库，特点：

- 高性能，适合处理大规模文本
- 预训练模型支持多种语言
- API设计简洁易用

**安装方式：**

```
pip install spacy
python -m spacy download en_core_web_sm
```

# spaCy文本处理

使用spaCy进行文档处理和实体识别：

```
## 加载语言模型并处理文档
nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying U.K. startup for $1 billion")

## 实体识别和词性分析
for ent in doc.ents:
    print("{}: {}".format(ent.text, ent.label_))
for token in doc:
    print("{}: {}".format(token.text, token.pos_))
```

# spaCy实体识别结果

**识别结果示例：**

- Apple: ORG（组织机构）
- U.K.: GPE（地理政治实体）
- $1 billion: MONEY（金额）

spaCy自动识别文本中的命名实体，包括人名、地名、组织、时间、金额等。

# NLTK vs spaCy对比

| 特性 | NLTK | spaCy |
| --- | --- | --- |
| 设计目标 | 教学研究 | 生产应用 |
| 处理速度 | 较慢 | 快速 |
| 易用性 | 灵活但复杂 | 简洁高效 |
| 预训练模型 | 有限 | 丰富 |
| 适用场景 | 学习、实验 | 实际项目 |

# 14.2.2 中文NLP工具：jieba与HanLP

中文NLP面临特殊挑战：

- 没有天然的词语分隔符
- 分词是所有中文NLP任务的基础
- 需要专门的中文处理工具

常用工具：**jieba**和**HanLP**

# jieba分词库

**jieba**是最流行的中文分词库，特点：

- 支持多种分词模式
- 词性标注功能
- 支持自定义词典
- 开源免费

**安装方式：**

```
pip install jieba
```

# jieba分词模式

使用jieba进行中文文本的分词：

**输出结果：**

- 精确模式：我/ 来到/ 北京/ 清华大学
- 全模式：我/ 来到/ 北京/ 清华/ 清华大学/ 华大/ 大学

```
## 精确分词和全模式分词
seg_precise = jieba.cut("我来到北京清华大学", cut_all=False)
seg_full = jieba.cut("我来到北京清华大学", cut_all=True)
print("精确模式:", "/ ".join(seg_precise))
print("全模式:", "/ ".join(seg_full))
```

# jieba词性标注

使用jieba进行词性标注：

**输出结果：**

- 我: r（代词）
- 爱: v（动词）
- 自然语言: n（名词）
- 处理: v（动词）

```
## 词性标注
words = pseg.cut("我爱自然语言处理")
for word, flag in words:
    print("{}: {}".format(word, flag))
```

# HanLP工具库

**HanLP**是面向生产环境的多语言NLP工具包：

- 支持多种中文处理任务
- 提供预训练模型
- 准确率高，速度快

**安装方式：**

```
pip install hanlp
```

# HanLP多任务处理

使用HanLP进行综合的中文文本分析：

```
## 初始化HanLP并进行多任务处理
HanLP = hanlp.load(hanlp.pretrained.mtl.CLOSE_TOK_POS_NER_SRL_DEP_SDP_CON_ELECTRA_SMALL_ZH)
result = HanLP(['2021年HanLP发布了多语言版本。'])

## 输出分析结果
print("分词:", result['tok'])
print("词性:", result['pos'])
print("命名实体:", result['ner'])
```

# HanLP分析结果

**输出示例：**

- 分词：['2021年', 'HanLP', '发布', '了', '多语言', '版本', '。']
- 词性：['t', 'nz', 'v', 'u', 'n', 'n', 'w']
- 命名实体：[('2021年', 'DATE'), ('HanLP', 'PRODUCT')]

HanLP提供一站式的中文NLP处理能力。

# jieba vs HanLP对比

| 特性 | jieba | HanLP |
| --- | --- | --- |
| 分词速度 | 快 | 中等 |
| 功能范围 | 分词为主 | 全面 |
| 准确率 | 良好 | 优秀 |
| 学习成本 | 低 | 中等 |
| 适用场景 | 简单任务 | 复杂任务 |

# 14.2.3 Ask AI：Transformer模型应用

**Transformer架构**彻底改变了NLP领域：

- 2017年Google提出
- 自注意力机制
- 并行计算能力强
- 长距离依赖建模

**代表模型：**BERT、GPT、T5等

# transformers库

Hugging Face的transformers库提供：

- 数千个预训练模型
- 统一的API接口
- 多种NLP任务支持

**安装方式：**

```
pip install transformers
```

# 英文情感分析

使用预训练模型进行情感分析：

```
## 英文情感分析
classifier = pipeline("sentiment-analysis")
result = classifier("I love this movie!")
print(result)
```

**输出结果：**

```
[{'label': 'POSITIVE', 'score': 0.9998}]
```

# 中文情感分析

使用中文预训练模型进行情感分析：

```
## 中文情感分析
classifier_zh = pipeline("sentiment-analysis",
                         model="uer/roberta-base-finetuned-chinanews-chinese")
result_zh = classifier_zh("这部电影真的很棒！")
print(result_zh)
```

**输出结果：**

```
[{'label': 'LABEL_1', 'score': 0.9876}]
```

# pipeline支持的任务

transformers库支持多种NLP任务：

| 任务 | pipeline名称 |
| --- | --- |
| 情感分析 | sentiment-analysis |
| 文本分类 | text-classification |
| 命名实体识别 | ner |
| 问答 | question-answering |
| 文本生成 | text-generation |
| 翻译 | translation |

# Ask AI学习建议

向AI助手询问以下进阶内容：

1. **"如何选择合适的预训练模型进行微调？"**
2. **"Transformer模型的注意力机制是如何工作的？"**

通过Ask AI深入学习Transformer架构的原理和应用。

# 实践练习

**练习 14.2.1：工具库对比**

比较NLTK、spaCy、jieba、HanLP的处理速度和功能差异。

测试要点：

- 分词速度对比
- 准确率评估
- 功能范围比较

# 实践练习

**练习 14.2.2：中文处理实践**

使用jieba和HanLP处理中文文本，对比分词和词性标注结果。

测试文本建议：

- 新闻标题
- 商品评论
- 技术文档

# 实践练习

**练习 14.2.3：Transformer模型探索**

使用transformers库尝试不同预训练模型的文本分类效果。

探索内容：

- 尝试不同的中文模型
- 比较模型推理速度
- 分析分类准确率