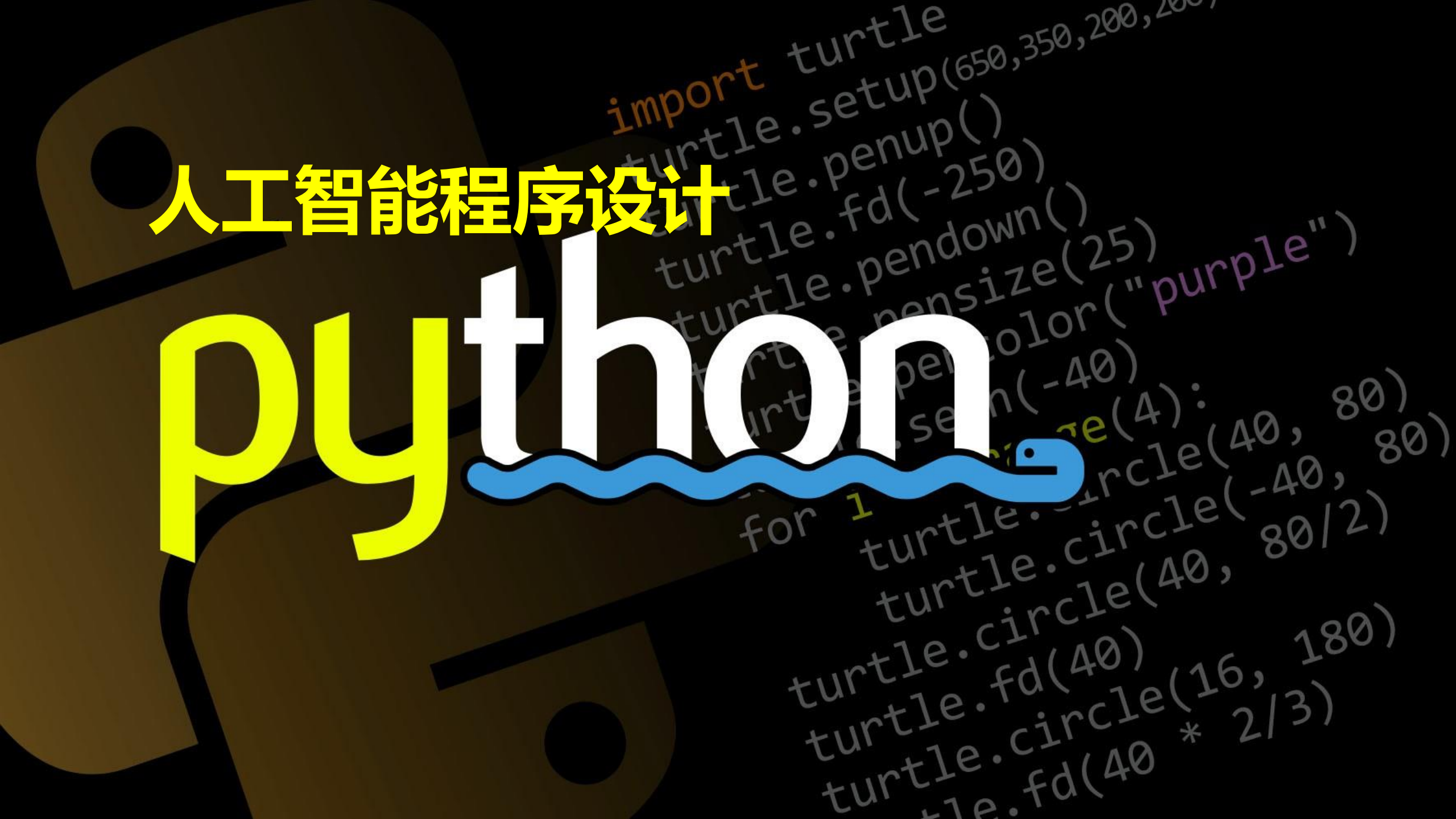
人工智能程序设计


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

10.1 数值计算基础：NUMPY

北京石油化工学院 人工智能研究院

刘 强

NumPy 简介

NumPy（Numerical Python）是Python科学计算的基础库：

- 提供高性能的多维数组对象和处理工具
- 几乎所有Python数据科学库的基础
- 包括 Pandas、Matplotlib、Scikit-learn 等

10.1.1 NumPy简介与环境搭建

- **高性能数组对象**：`ndarray`（N维数组）比Python列表快10-100倍
- **广播机制**：支持不同形状数组之间的运算
- **丰富的数学函数**：线性代数、随机数生成、傅里叶变换等

环境搭建

安装 **NumPy** 库：

```
## 使用pip安装
pip install numpy

## 验证安装
python -c "import numpy as np; print(np.__version__)"
```

导入 **NumPy** 的标准方式：

```
import numpy as np
```

示例 10.1.1：NumPy数组基础操作

体验 **NumPy** 的基本功能：创建数组、进行运算和使用数学函数：

```
import numpy as np

## 创建数组
arr = np.array([1, 2, 3, 4, 5])
print("数组: ", arr)
print("数组类型: ", type(arr))

## 数组运算
result = arr * 2
print("数组乘以2: ", result)

## 数学函数
sqrt_result = np.sqrt(arr)
print("数组平方根: ", sqrt_result)
```

10.1.2 数组创建与基本操作

最直接的数组创建方法是从Python列表转换，可以创建一维或多维数组：

```
## 一维数组
arr1d = np.array([1, 2, 3, 4])
print("一维数组: ", arr1d)

## 二维数组
arr2d = np.array([[1, 2, 3], [4, 5, 6]])
print("二维数组: ")
print(arr2d)
```

使用NumPy函数创建数组

NumPy 提供了多种便捷函数来创建特殊类型的数组：

```
## 创建零数组
zeros = np.zeros((3, 4))
print("零数组: ")
print(zeros)

## 创建全1数组
ones = np.ones((2, 3))
print("全1数组: ")
print(ones)

## 创建等差数列
range_arr = np.arange(0, 10, 2)
print("等差数列: ", range_arr)

## 创建等间隔数组
linspace_arr = np.linspace(0, 1, 5)
print("等间隔数组: ", linspace_arr)
```

数组的基本属性

每个 NumPy 数组都有重要的属性信息，了解这些属性有助于更好地操作数组：

属性详解：

- **shape**：各维度的大小，(2, 3) 表示2行3列
- **ndim**：维度数，2 表示二维数组
- **size**：元素总数
- **dtype**：数据类型

```
arr = np.array([[1, 2, 3], [4, 5, 6]])

print("数组形状: ", arr.shape)      # (2, 3)
print("数组维度: ", arr.ndim)       # 2
print("数组大小: ", arr.size)       # 6
print("数据类型: ", arr.dtype)      # int64
```

数组索引与切片

一维数组的索引方式与Python列表相同，支持正向和反向索引：

```
arr = np.array([10, 20, 30, 40, 50])

print("索引0的元素：", arr[0])      # 10
print("索引-1的元素：", arr[-1])    # 50
print("前三个元素：", arr[:3])      # [10 20 30]
```

二维数组索引

二维数组支持行列索引，可以方便地访问行、列或特定元素：

索引语法说明：

- `arr2d[0]`：获取第一行
- `arr2d[:, 1]`：冒号表示所有行，1表示第二列
- `arr2d[1, 2]`：第2行第3列的元素

```
arr2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print("第一行：", arr2d[0])         # [1 2 3]
print("第二列：", arr2d[:, 1])      # [2 5 8]
print("特定元素：", arr2d[1, 2])    # 6
```

条件索引

条件索引允许根据条件筛选数组元素，在数据分析中非常有用：

条件表达式返回布尔数组，用于筛选满足条件的元素。

```
arr = np.array([1, 2, 3, 4, 5])
condition = arr > 3
print("条件: ", condition)           # [False False False True True]
print("满足条件的元素: ", arr[condition])  # [4 5]
```

10.1.3 数组运算与数学函数

NumPy 支持元素级（element-wise）的数学运算，这是相比普通Python列表的重要优势：

```
arr1 = np.array([1, 2, 3, 4])
arr2 = np.array([5, 6, 7, 8])

## 基本运算
print("加法: ", arr1 + arr2)       # [ 6  8 10 12]
print("减法: ", arr1 - arr2)       # [-4 -4 -4 -4]
print("乘法: ", arr1 * arr2)       # [ 5 12 21 32]
print("除法: ", arr2 / arr1)       # [5. 3. 2.33 2.]

## 标量运算
print("数组乘以标量: ", arr1 * 2)  # [2 4 6 8]
```

常用数学函数

NumPy 提供了丰富的数学函数，包括平方根、指数、对数、三角函数等：

```
arr = np.array([1, 4, 9, 16])

## 平方根
print("平方根: ", np.sqrt(arr))    # [1. 2. 3. 4.]

## 指数函数
print("指数: ", np.exp([1, 2, 3])) # [2.718 7.389 20.086]

## 对数函数
print("自然对数: ", np.log([1, 2, 3]))  # [0. 0.693 1.099]

## 三角函数
angles = np.array([0, np.pi/2, np.pi])
print("正弦值: ", np.sin(angles))  # [0. 1. 0.]
```

统计函数

NumPy 提供了强大的统计分析功能，包括均值、中位数、标准差等：

这些统计函数在数据分析中经常使用。

```
data = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])

print("平均值: ", np.mean(data))    # 5.5
print("中位数: ", np.median(data))  # 5.5
print("标准差: ", np.std(data))     # 2.87
print("最大值: ", np.max(data))     # 10
print("最小值: ", np.min(data))     # 1
print("总和: ", np.sum(data))       # 55
```

数组形状操作

NumPy 允许灵活地改变数组的形状，包括重塑、展平和转置：

```
arr = np.array([1, 2, 3, 4, 5, 6])

## 改变形状
reshaped = arr.reshape(2, 3)
print("重塑后：")
print(reshaped)

## 展平数组
flattened = reshaped.flatten()
print("展平后：", flattened)

## 转置
transposed = reshaped.T
print("转置后：")
print(transposed)
```

实践练习

练习 10.1.1：数组创建与基本操作

1. 创建一个3×4的零数组和一个2×5的全1数组
2. 创建一个包含0到20的偶数的数组
3. 创建一个3×3的单位矩阵（对角线为1，其他为0）

实践练习

练习 10.1.2：传感器数据处理

给定温度传感器数据：[22.5, 23.1, 21.8, 24.2, 22.8, 21.2, 23.4, 24.6, 22.1, 23.8]

1. 计算平均温度、最高温度、最低温度
2. 计算温度数据的标准差和方差
3. 找出高于平均温度的数据点

实践练习

练习 10.1.3：矩阵运算应用

1. 创建两个3×3矩阵，计算加法、减法和元素级乘法
2. 使用NumPy函数计算矩阵的行和与列和
3. 创建1到12的数组，重塑为3×4矩阵，然后转置

本节小结

- **NumPy** 是 Python 科学计算的基础库
- **ndarray** 数组比 Python 列表高效 10-100 倍
- 核心操作：创建、索引、切片、运算
- 常用函数：数学函数、统计函数、形状操作