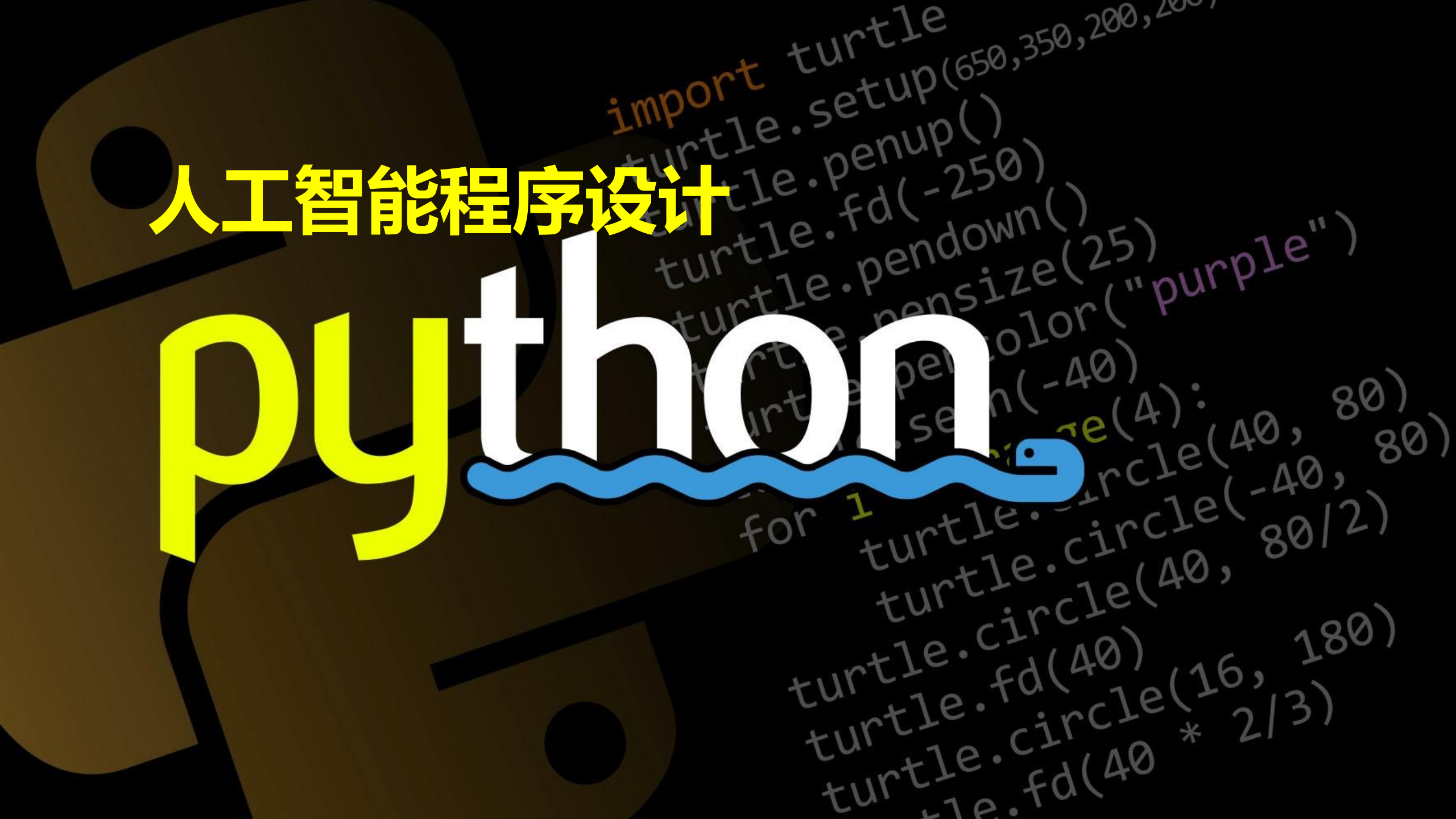
人工智能程序设计


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

9.2 PYGAME核心功能

北京石油化工学院 人工智能研究院

刘 强

本节概述

在掌握了 Pygame 的基本框架后，我们需要学习游戏开发中最重要的两个核心功能：

- **图形绘制**：创建各种视觉元素
- **事件处理**：响应用户的各种操作

这些技能是制作任何游戏的基础。

9.2.1 图形绘制功能

Pygame 提供了丰富的图形绘制功能：

- **基本几何图形**：矩形、圆形等游戏中常用的图形
- **颜色系统**：支持 RGB 颜色模式，可创建丰富的色彩效果
- **坐标系统**：使用像素坐标系，左上角为 (0,0)，向右向下为正方向

示例 9.2.1：图形绘制

创建一个程序，分为4个步骤：

1. **步骤1**：程序初始化
2. **步骤2**：定义颜色
3. **步骤3**：绘制图形函数
4. **步骤4**：游戏主循环

步骤1：程序初始化

初始化 Pygame 并创建游戏窗口。

```
import pygame
import sys

## 初始化Pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("图形绘制演示")
clock = pygame.time.Clock()
```

步骤2：定义颜色

定义游戏中使用的 **RGB** 颜色常量。

RGB 颜色：每个分量取值 0-255

```
## 定义常用颜色（RGB值）
WHITE = (255, 255, 255)
RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 255)
BLACK = (0, 0, 0)
```

步骤3：绘制图形函数

创建绘图函数，绘制矩形和圆形。

```
def draw_shapes(screen):
    """绘制游戏常用图形"""
    # 清空屏幕
    screen.fill(WHITE)

    # 绘制矩形：pygame.draw.rect(surface, color, (x, y, width, height))
    pygame.draw.rect(screen, RED, (100, 100, 200, 150))

    # 绘制圆形：pygame.draw.circle(surface, color, (center_x, center_y), radius)
    pygame.draw.circle(screen, GREEN, (500, 200), 80)
```

步骤3：函数说明

- `screen.fill()`：用指定颜色填充整个屏幕表面
- `pygame.draw.rect()`：绘制矩形，参数为**(表面，颜色，(x，y，宽度，高度))**
- `pygame.draw.circle()`：绘制圆形，参数为**(表面，颜色，(中心x，中心y)，半径)**

步骤4：游戏主循环

运行游戏循环，处理事件和绘制图形。

```
## 游戏主循环
running = True
while running:
    # 处理事件
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    # 绘制图形
    draw_shapes(screen)

    # 更新显示
    pygame.display.flip()

    # 控制帧率
    clock.tick(60)

## 退出程序
pygame.quit()
sys.exit()
```

步骤4：函数说明

- `pygame.event.get()`：获取事件队列中的所有事件
- `pygame.display.flip()`：更新整个屏幕显示，显示所有绘制内容
- `clock.tick(60)`：控制游戏帧率为 60FPS

运行这个程序，你会看到窗口中同时显示红色矩形和绿色圆形。

9.2.2 事件处理系统

Pygame 的事件处理系统让我们能够响应用户的各种操作：

- **键盘事件**：检测方向键按下，控制游戏对象移动
- **鼠标事件**：检测鼠标位置，实现鼠标控制
- **窗口事件**：检测窗口关闭，安全退出程序

示例 9.2.2：事件处理

创建一个鼠标跟随的圆形程序，分为4个步骤：

1. **步骤1**：程序初始化和变量定义
2. **步骤2**：事件处理函数
3. **步骤3**：绘制函数
4. **步骤4**：主游戏循环

步骤1：程序初始化和变量定义

初始化 Pygame 并定义圆形的位置和颜色。

```
import pygame
import sys

## 初始化Pygame
pygame.init()
screen = pygame.display.set_mode((800, 600))
pygame.display.set_caption("事件处理演示")
clock = pygame.time.Clock()

## 定义颜色
WHITE = (255, 255, 255)
RED = (255, 0, 0)
LIGHT_BLUE = (173, 216, 230)

## 定义移动圆形的属性
circle_x = 400
circle_y = 300
speed = 5
```

步骤2：事件处理函数

处理窗口关闭事件和鼠标位置检测。

```
def handle_events():
    """处理用户输入事件"""
    global circle_x

    # 处理事件队列中的事件
    for event in pygame.event.get():
        # 窗口关闭事件
        if event.type == pygame.QUIT:
            return False

    # 鼠标位置检测
    mouse_x = pygame.mouse.get_pos()[0]
    circle_x = max(20, min(mouse_x, 780))

    return True
```

步骤2：函数说明

- `global circle_x`：声明 `circle_x` 为全局变量，允许函数修改它
- `pygame.event.get()`：获取事件队列中的所有事件，返回事件列表
- `pygame.QUIT`：窗口关闭事件类型常量
- `pygame.mouse.get_pos()`：获取鼠标当前位置，返回 `(x, y)` 坐标元组
- `max(20, min(mouse_x, 780))`：限制圆形位置在屏幕范围内

步骤3：绘制函数

绘制可移动的浅蓝色圆形。

- `screen.fill()`：用指定颜色填充整个屏幕表面
- `pygame.draw.circle()`：在屏幕上绘制圆形
- `pygame.display.flip()`：更新整个屏幕显示

```
def draw_game():
    """绘制游戏画面"""
    # 清空屏幕
    screen.fill(WHITE)

    # 绘制可移动的圆形
    pygame.draw.circle(screen, LIGHT_BLUE, (circle_x, circle_y), 20)

    # 更新显示
    pygame.display.flip()
```

步骤4：主游戏循环

运行游戏主循环，整合事件处理和绘制功能。

运行后，浅蓝色圆形会跟随鼠标水平移动。

```
## 游戏主循环
running = True
while running:
    # 处理事件
    running = handle_events()

    # 绘制画面
    draw_game()

    # 控制帧率
    clock.tick(60)

## 退出程序
pygame.quit()
sys.exit()
```

9.2.3 Ask AI：探索Pygame高级功能

掌握了 Pygame 的基本功能后，可以向 AI 助手询问更多高级游戏开发技术：

- "如何在 Pygame 中实现精灵（Sprite）系统？"
- "如何添加背景音乐和音效？"
- "如何实现游戏中的碰撞检测？"

实践练习

练习 9.2.1：图形绘制练习

1. 创建一个程序，绘制不同颜色和大小的几何图形
2. 实现一个简单的画板程序，鼠标拖拽时绘制轨迹
3. 制作一个彩虹效果，使用不同颜色绘制同心圆

实践练习

练习 9.2.2：事件处理练习

1. 制作一个简单的"接球游戏"，用键盘控制挡板左右移动
2. 实现鼠标跟随效果，一个图形始终跟随鼠标位置
3. 创建一个简单的绘图程序，支持不同颜色和画笔大小

本节小结

- 图形绘制：`pygame.draw.rect()`、`pygame.draw.circle()`
- 颜色系统：RGB 三元组 `(R, G, B)`
- 事件处理：`pygame.event.get()`、`pygame.QUIT`
- 鼠标位置：`pygame.mouse.get_pos()`