

# 湖南大学实训报告

课程名称： 计算与人工智能概论 实验类型： 团队实训

实验项目名称： 猫狗世纪大战

学生姓名： 刘子楠 班级： 数学 2402 学号：  
202410040229

同组姓名： 冯钰淞 班级： 数学 2402 学号：  
202410040205

同组姓名： 热吾扎·达吾列提汗 班级： 数学 2402 学号：  
202410040232

## 一、实训目的

1. 巩固计算与人工智能概念课程所学基础知识
2. 拓展训练学生计算和 AI 思维能力
3. 加强团队分工与合作，培养学生团队协作能力

## 二、实训内容和团队分工

1 任务描述：使用 Python 创建一个飞机大战的游戏背景，编写一个能创建友方战机并发射子弹的程序，编写一个能创建敌方战机并发射子弹的程序并完成整个游戏编写。

2 团队分工：

代码：冯钰淞 刘子楠

报告：热吾扎·达吾列提汗

## 三、实训环境

开发硬件：Lenovo XiaoXin 14IRL18

开发软件：Windows11

库：python 里的 pygame 库

## 四、实训方法和步骤

步骤 1 从 python 里下载 pygame 库

步骤 2 导入 pygame 库

```
import random
import pygame
```

步骤 3 设置屏幕大小和刷新帧率

```
# 屏幕大小的常量
SCREEN_RECT = pygame.Rect(0, 0, 480, 700)
# 刷新的帧率
Hello from the pygame community. https://www.pygame.org/contribute.html
FRAME_PER_SEC = 60
```

步骤 4 创建游戏精灵类

```
class GameSprite(pygame.sprite.Sprite):
    """猫狗世纪大战游戏精灵"""
    def __init__(self, image_name, speed=1):
        # 调用父类的初始化方法
        super().__init__()
        # 定义对象的属性
        self.image = pygame.image.load(image_name)
        self.rect = self.image.get_rect()
        self.speed = speed
    # 更新数据显示
    def update(self):
        # 在屏幕的垂直方向上移动
        self.rect.y += self.speed
```

步骤 5 创建游戏背景精灵

```
class Background(GameSprite):
    """游戏背景精灵"""
    def __init__(self, is_alt=False):
        # 1. 调用父类方法实现精灵的创建(image/rect/speed)
        super().__init__("./images/background.png")
        # 2. 判断是否是交替图像, 如果是, 需要设置初始位置
        if is_alt:
            self.rect.y = -self.rect.height
    def update(self):
        # 1. 调用父类的方法实现
        super().update()
        # 2. 判断是否移出屏幕, 如果移出屏幕, 将图像设置到屏幕的上方
        if self.rect.y >= SCREEN_RECT.height:
            self.rect.y = -self.rect.height
```

步骤 6 主程序编写

```

class DogsandCatsGame(object):
    """猫狗世纪大战主游戏"""
    def __init__(self):
        print("游戏初始化")
        # 1. 创建游戏的窗口
        self.screen = pygame.display.set_mode(SCREEN_RECT.size)
        # 2. 创建游戏的时钟
        self.clock = pygame.time.Clock()
        # 3. 调用私有方法，精灵和精灵组的创建
        self.__create_sprites()
    def __create_sprites(self):
        # 创建背景精灵和精灵组
        bg1 = Background()
        bg2 = Background(True)
        self.back_group = pygame.sprite.Group(bg1, bg2)
    def start_game(self):
        print("游戏开始...")
        while True:
            # 1. 设置刷新帧率
            self.clock.tick(FRAME_PER_SEC)
            # 2. 事件监听
            self.__event_handler()
            # 3. 更新/绘制精灵组
            self.__update_sprites()
            # 4. 更新显示
            pygame.display.update()
    def __event_handler(self):
        for event in pygame.event.get():
            # 判断是否退出游戏
            if event.type == pygame.QUIT:
                DogsandCatsGame.__game_over()
    def __update_sprites(self):
        self.back_group.update()
        self.back_group.draw(self.screen)
    @staticmethod
    def __game_over():
        print("游戏结束")
        pygame.quit()
        exit()

```

步骤 7 运行主程序

```

if __name__ == '__main__':
    # 创建游戏对象
    game = DogsandCatsGame()
    # 启动游戏
    game.start_game()

```

步骤 8 创建猫咪，小狗和毛球精灵

```

class Hero(GameSprite):
    """猫咪精灵"""
    def __init__(self):
        # 1. 调用父类方法, 设置image&speed
        super().__init__("./images/me1.png", 0)
        # 2. 设置猫咪的初始位置
        self.rect.centerx = SCREEN_RECT.centerx
        self.rect.bottom = SCREEN_RECT.bottom - 120
        # 3. 创建毛球的精灵组
        self.bullets = pygame.sprite.Group()
    def update(self):
        # 猫咪在水平方向移动
        self.rect.x += self.speed
        # 控制猫咪不能离开屏幕
        if self.rect.x < 0:
            self.rect.x = 0
        elif self.rect.right > SCREEN_RECT.right:
            self.rect.right = SCREEN_RECT.right
    def fire(self):
        print("发射毛球...")
        for i in (0, 1, 2):
            # 1. 创建毛球精灵
            bullet = Bullet()
            # 2. 设置精灵的位置
            bullet.rect.bottom = self.rect.y - i * 20
            bullet.rect.centerx = self.rect.centerx
            # 3. 将精灵添加到精灵组
            self.bullets.add(bullet)

class Bullet(GameSprite):
    """毛球精灵"""
    def __init__(self):
        # 调用父类方法, 设置毛球图片, 设置初始速度
        super().__init__("./images/bullet1.png", -2)
    def update(self):
        # 调用父类方法, 让毛球沿垂直方向飞行
        super().update()
        # 判断毛球是否飞出屏幕
        if self.rect.bottom < 0:
            self.kill()
    def __del__(self):
        print("毛球被销毁...")

```

```

class DogsandCatsGame(object):
    """猫狗世纪大战主游戏"""
    def __init__(self):
        print("游戏初始化")
        # 1. 创建游戏的窗口
        self.screen = pygame.display.set_mode(SCREEN_RECT.size)
        # 2. 创建游戏的时钟
        self.clock = pygame.time.Clock()
        # 3. 调用私有方法，精灵和精灵组的创建
        self.__create_sprites()
        # 4. 设置定时器事件 - 创建小狗 1s
        pygame.time.set_timer(CREATE_ENEMY_EVENT, 1000)
        pygame.time.set_timer(HERO_FIRE_EVENT, 500)
    def __create_sprites(self):
        # 创建背景精灵和精灵组
        bg1 = Background()
        bg2 = Background(True)
        self.back_group = pygame.sprite.Group(bg1, bg2)
        # 创建小狗的精灵组
        self.enemy_group = pygame.sprite.Group()
        # 创建猫咪的精灵和精灵组
        self.hero = Hero()
        self.hero_group = pygame.sprite.Group(self.hero)
    def start_game(self):
        print("游戏开始...")
        while True:
            # 1. 设置刷新帧率
            self.clock.tick(FRAME_PER_SEC)
            # 2. 事件监听
            self.__event_handler()
            # 3. 碰撞检测
            self.__check_collide()
            # 4. 更新/绘制精灵组
            self.__update_sprites()
            # 5. 更新显示
            pygame.display.update()
    def __event_handler(self):
        for event in pygame.event.get():
            # 判断是否退出游戏
            if event.type == pygame.QUIT:
                PlaneGame.__game_over()
            elif event.type == CREATE_ENEMY_EVENT:
                # print("小狗出场...")
                # 创建小狗精灵
                enemy = Enemy()
                # 将小狗精灵添加到小狗精灵组
                self.enemy_group.add(enemy)

```

```

        elif event.type == HERO_FIRE_EVENT:
            self.hero.fire()
        # elif event.type == pygame.KEYDOWN and event.key == pygame.K_RIGHT:
        #     print("向右移动...")
        # 使用键盘提供的方法获取键盘按键 - 按键元组
        keys_pressed = pygame.key.get_pressed()
        # 判断元组中对应的按键索引值 1
        if keys_pressed[pygame.K_RIGHT]:
            self.hero.speed = 2
        elif keys_pressed[pygame.K_LEFT]:
            self.hero.speed = -2
        else:
            self.hero.speed = 0
    def __check_collide(self):
        # 1. 老鼠摧毁小狗
        pygame.sprite.groupcollide(self.hero.bullets, self.enemy_group, True, True)
        # 2. 小狗摧毁猫咪
        enemies = pygame.sprite.spritecollide(self.hero, self.enemy_group, True)
        # 判断列表时候有内容
        if len(enemies) > 0:
            # 让猫咪牺牲
            self.hero.kill()
            # 结束游戏
            PlaneGame.__game_over()
    def __update_sprites(self):
        self.back_group.update()
        self.back_group.draw(self.screen)
        self.enemy_group.update()
        self.enemy_group.draw(self.screen)
        self.hero_group.update()
        self.hero_group.draw(self.screen)
        self.hero.bullets.update()
        self.hero.bullets.draw(self.screen)
    @staticmethod
    def __game_over():
        print("游戏结束")
        pygame.quit()
        exit()

```

步骤 9 运行主程序

```

if __name__ == '__main__':
    # 创建游戏对象
    game = DogsandCatsGame()
    # 启动游戏
    game.start_game()

```

## 五、实训结果和分析

代码主要由以下几大部分构成：

def \_\_create\_sprites(self): 创建所需要的精灵和精灵组；

def \_\_event\_handler(self): 监听所发生的事件，如本例中监听是否处罚界面关闭事件；

def \_\_update\_sprites(self): 更新精灵函数；

def \_\_game\_over(): 执行退出游戏界面函数。

## 六、讨论与心得

1 从技术层面来看，我们更加熟练地掌握了 Python 语言的使用，尤其是在图形处理、游戏逻辑实现等方面有了很大的提升。

2 我们学会了如何使用第三方库来扩展 Python 的功能，为以后开发更复杂的游戏奠定了基础。

3 从团队协作角度来说，我们提高了沟通和协作能力。在与不同背景和专业技能的团队成员合作过程中，我们学会了倾听他人的意见，尊重彼此的想法，共同解决问题。这种团队协作能力不仅在游戏开发中非常重要，在今后的工作和学习中也将受益匪浅。