《Python程序设计基础》程序设计作品说明书

题目： 外星人入侵

学院： 21计科04

姓名： 程克智

学号： B20210101330

指导教师： 周景

起止日期：2023.11.10-2023.12.10

Github地址：https://github.com/CodCat/pythongame.git

摘要

本项目使用外星人入侵的模板制作了一款哆啦a梦使用铜锣烧向大雄复仇的游戏

关键词：

第1章 需求分析

研究既有代码，确实实现新功能前是否要进行重构

在屏幕上左上角添加一个大雄（外星人），并指定合适的边距

根据第一个大雄的边距和屏幕上尺寸计算屏幕上可容纳多少个大雄，我们将编写一个循环来创建一些大雄，这些大雄填满了屏幕的上半部分。

让大雄群向两边和下方移动，直到大雄被全部击落，有大雄撞到哆啦a梦，或有大雄抵达屏幕

低端。如果所有大雄都被击落，我们将再创建一排大雄。如果有大雄撞到了飞船或抵达屏幕

低端，我们将销毁哆啦a梦并创建一排大雄。

限制玩家可用的哆啦a梦数量左上角的哆啦a梦用完后，游戏结束

第2章 分析与设计

1. 系统架构设计

本游戏将基于Python的pygame库进行开发，整个系统架构主要包括以下几个模块：

游戏界面模块：负责渲染游戏画面，包括背景、哆啦A梦、大雄等元素。 游戏逻辑模块：处理游戏的核心逻辑，包括大雄的移动、哆啦A梦的发射子弹、碰撞检测等。 玩家输入模块：捕获玩家的输入，包括点击、拖动等操作。 游戏状态管理模块：维护游戏的状态，包括当前关卡、得分、剩余生命等。 音效和音效模块：播放背景音乐和音效，增加游戏的氛围。 2. 系统流程设计

游戏的主要流程如下：

初始化游戏，创建游戏窗口和初始的游戏状态。 根据游戏状态，创建初始的大雄和哆啦A梦，并设置其初始位置和速度。 进入游戏循环，不断更新游戏状态，并根据玩家输入和游戏逻辑更新画面。 在每一帧中，检查是否有大雄被击落或到达屏幕底部，如果有，则创建新的一排大雄，并更新游戏得分。 检查玩家是否用完了所有的哆啦A梦，如果是，则结束游戏。 根据玩家的表现和游戏进度，播放相应的音效和动画效果。 3. 系统模块详细设计

游戏界面模块：使用pygame的函数绘制游戏画面，包括背景、哆啦A梦、大雄等元素。使用pygame的函数来处理动画效果和帧更新。 游戏逻辑模块：根据玩家输入和大雄的位置速度等信息，计算大雄的移动路径，检测碰撞，计算得分等。使用Python的列表和字典等数据结构来存储和管理游戏状态。 玩家输入模块：使用pygame的函数来捕获玩家的输入，包括点击、拖动等操作，并将这些操作转化为游戏逻辑模块可以理解的指令。 游戏状态管理模块：维护游戏的状态，包括当前关卡、得分、剩余生命等。使用Python的数据结构来存储和管理这些信息。 音效和音效模块：使用pygame的函数来播放背景音乐和音效。可以使用Python的字典数据结构来存储各种音效和对应的播放函数。 4. 数据库设计

由于本游戏是小型游戏，并没有复杂的数据库需求。可以将玩家的最高得分和当前得分保存在一个文本文件中，以供下次游戏时读取。可以使用Python的字典数据结构来存储这些信息。

第3章 软件测试

本章的内容主要包括以类和函数作为单元进行单元测试，编写的对系统的主要功能的测试用例，以及测试用例执行的测试报告。

单元测试用例

```
import pygame
from settings import Settings

def test_Settings():
    # 创建Settings对象
    settings = Settings()

    # 测试外星人速度设置
    assert settings.alien_speed_factor == 1

    # 测试飞船速度设置
    assert settings.ship_speed_factor == 1.5

    # 测试子弹速度设置
    assert settings.bullet_speed_factor == 3

    # 测试屏幕设置
    assert settings.screen_width == 1200
    assert settings.screen_height == 800

    # 测试随游戏进行而变化的设置
    settings.increase_speed()
    assert settings.ship_speed_factor == 1.5 * 1.1
    assert settings.bullet_speed_factor == 3 * 1.1
    assert settings.alien_speed_factor == 1 * 1.1

    # 测试外星人点数设置
```

```
    assert settings.alien_points == 50
```

这个测试代码通过断言来验证Settings类的各个设置是否正确,后续程序没有报错证明通过测试

结论

通过系统架构设计，我们确定了使用Python的pygame库进行游戏开发，并将游戏划分为不同的模块，每个模块负责特定的功能，这有助于代码的组织和维护。

我们设计了游戏的系统流程，明确了游戏的主要循环和各个模块之间的交互方式。这有助于确保游戏的逻辑性和流畅性。

通过对系统模块的详细设计，我们确定了每个模块的具体实现方法和使用的数据结构，这有助于代码的编写和调试。

考虑到本游戏的规模较小，我们决定使用文本文件来保存玩家的得分信息，而不是引入复杂的数据库系统。这简化了数据存储和读取的实现。

综上所述，通过合理的系统架构设计和模块划分，我们成功地实现了小游戏外星人入侵的分析与设计。然而，需要注意的是，在实际开发过程中可能会遇到一些技术挑战和实现细节上的调整。因此，建议在开发过程中保持灵活性，并根据实际情况进行迭代和改进。

参考文献