流星雨

田春迅，姜鑫

摘要：本创意作品通过模块化设计，建立实时系统并计时计次，以随机生成、寻路算法、样式设计实现了随机下落的流星雨，玩家能与其交互，进行躲避和道具使用等操作，最终结算分数。并在此框架之上拓展了三种道具及相应的动画、音乐，最终实现了一款内容完备、过程流畅、容易上手的下落式躲避游戏。

1. 选题及创意介绍

本组以设计一款趣味小游戏为选题，从东方绀珠传中汲取灵感，最终在micro bit上实现了一款纵向躲避小游戏。玩家可以操纵一架飞机，躲避屏幕中不停滑过的流星，获取更高的分数。为了平衡游戏难度以及提高玩法趣味性，我们额外设计了三款道具，可以在游戏开始前进行选择，供玩家在游戏过程中灵活使用，化解困境。

1. 设计方案
2. 整体目标：

玩家在选择特定道具后，通过左右按键操纵飞机以及使用道具躲避上方生成的流星雨，失败后结算最终分数。

1. 功能模块设计：

玩家移动机制：通过左右按键控制移动。

流星雨生成：随机生成，随难度增加提高生成密度，同时保证至少有一条出路，能随着时间位置不断下移。

计分机制：游玩时间越长，游戏难度增加，流星雨下落越快，最终得分越高。

道具机制：开局时可以选一个道具，如：被逼到边缘时可以从一边边缘窜到另一边边缘的能力；可以使用一定次数的时缓能力；一键清除所有流星的能力。

1. 实现方案及代码分析
2. 实现方案：
   1. 基础框架：
      1. while True循环，以一个循环（约30ms）为最小原子时间（tick）运行
      2. 5\*5二维列表img用不同亮度代表流星雨和飞机的位置并实时更新，用于判定和显示
   2. 基础玩法：
      1. 在while True循环中不断计次代表时间和得分，达到一定次数后流星下进1位，更新img；每个tick内检测玩家有无按动按键，更新img。img一旦更新立即检测plane与star是否重合。如果死亡，则终止循环，结束游戏。
      2. 用list储存流星位置和长度。流星步进时修改位置，概率生成新流星，调用randint随机生成不同位置、长度，通过dfs寻路算法检测是否有解后更新img
   3. 道具实现：
      1. 用列表储存三种道具，用index代表当前选择。在循环中检测左右键并修改index，检测到中心键后确认，更新道具可使用情况，中止循环。
      2. Jump/一边边缘窜到另一边：移动机制细化
      3. Bomb/一键清除所有流星：在while True中检测中心键有无按动和道具可使用情况，如符合则清空流星list
      4. Timedown/时缓能力：在计次步进中增加timedowning状态，该状态下流星步进需要的计次数增加，timedowning随时间流逝减少直至为0恢复为普通状态。在while True中检测中心键有无按动和道具可使用情况，如符合，timedowning增加一定量tick。
3. 代码分析：
   1. 库引入和初始化
   2. 道具动画与选择：
      1. animations储存道具动画，delays储存播放间隔，switch函数负责更新道具index并播放相应的动画，注意将wait设置为False防止阻塞。
      2. 进入道具选择后根据按键情况switch（1）或（-1），如果中心键被按动，根据index更新相应道具状态。EGG为彩蛋系统，当检测到摇晃时出现问号，中心键被按动后随机给予all，任一道具或none。道具选择结束后倒数，播放音乐。
   3. 死亡函数die()：播放失败动画和音乐，显示分数
   4. 寻路函数searchWay(img, plane)：
      1. 进行一次回收防止内存溢出
      2. 用dfs储存待搜索的用tuple表示的位置，将img拷贝img2，current表示当前搜索节点，自下往上左右搜索，并将搜索过的节点在img2中标记。如果搜索到顶端就返回True如果不能就返回False。
   5. 星图生成summon\_stars(stars)：

将list中的流星在img上生成，在处于timedown状态时会显示低亮度，并在临近结束时闪烁。

* 1. 步进函数step(difficulty)：

星星参数：位置坐标（loc，y），长度lenth，以tuple形式储存在stars中。

* + 1. 将所有星星的y坐标减一，如果离开屏幕，则删除。
    2. 根据difficulty进行一次随机判定是否生成新星星，如判定生成，进行5次尝试，随机生成loc和lenth，生成后检查是否与已有的星星重合，如重合，则再次尝试；如成功，则将summon更新为true，并停止召唤。
    3. 清空img并调用summon\_stars(stars)，并判定玩家是否失败，如失败，将dead更新为True。
    4. 如果b步成功召唤，c步未死亡，则调用searchWay(img, plane)，如果寻路失败，从stars弹出b步召唤的流星并重新清空img，调用summon\_stars(stars)，在img中放置plane，显示图像，步进完成。
  1. 进入游戏：
     1. 计次加一，得分加一
     2. 如果调用bomb且bomb可使用，清空stars列表，清空屏幕，播放动画音效
     3. 如果调用timedown且timedown可使用，timedowning增加，并立刻调用summon\_stars(stars)，显示timedown状态的特殊亮度
     4. 计次部分，如果处在timedown状态，实时更新屏幕用于显示闪烁效果，降低步进频率；如果处在正常状态则无需实时更新屏幕，计次调用step(difficulty)即可
     5. 移动机制，按钮被按动时，如果不在角落或者jump可使用，则进行相应移动，并判定玩家是否失败。
     6. 如果失败，调用die()，游戏结束

1. 后续工作展望

优化流星生成算法，避免由于概率原因而造成难度突跃。同时还可以进行扩展，设计不同下落速度的流星，以及横向一次生成多个流星，丰富游玩体验。

优化积分系统，增加更多得分项目，比如惊险地躲过可以获得更高分数。

连接硬件，比如像素更高的显示屏以实现更丰富的玩法、更精美的效果与动画。

1. 小组分工合作

田春迅：确立框架，完成代码主体，优化性能使程序稳定运行；海报设计。

姜鑫：测试反馈，动图效果设计，游戏平衡性调整；撰写实习报告；拍摄视频