梁祖宁　21307140019

Task1：

```
Audio bgm("../audio/mountain.mp3");
bgm.play_loop();
Audio gameover("../audio/collide.mp3");
Audio applause("../audio/applause.mp3");
Audio flap("../audio/flap.mp3");
```

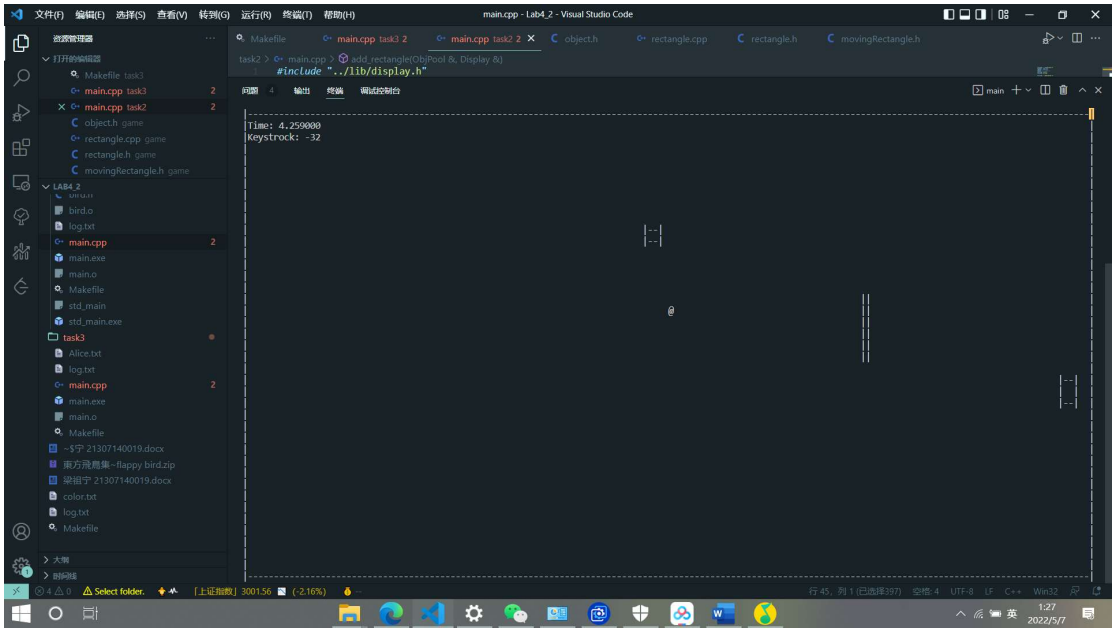
调用 Audio api 添加 bgm 与音效的变量名。

游戏结束时，停止 bgm，播放游戏结束的对应音乐

```
bgm.stop();
gameover.play_once();
applause.play_once();
```

Task2：

运行截图



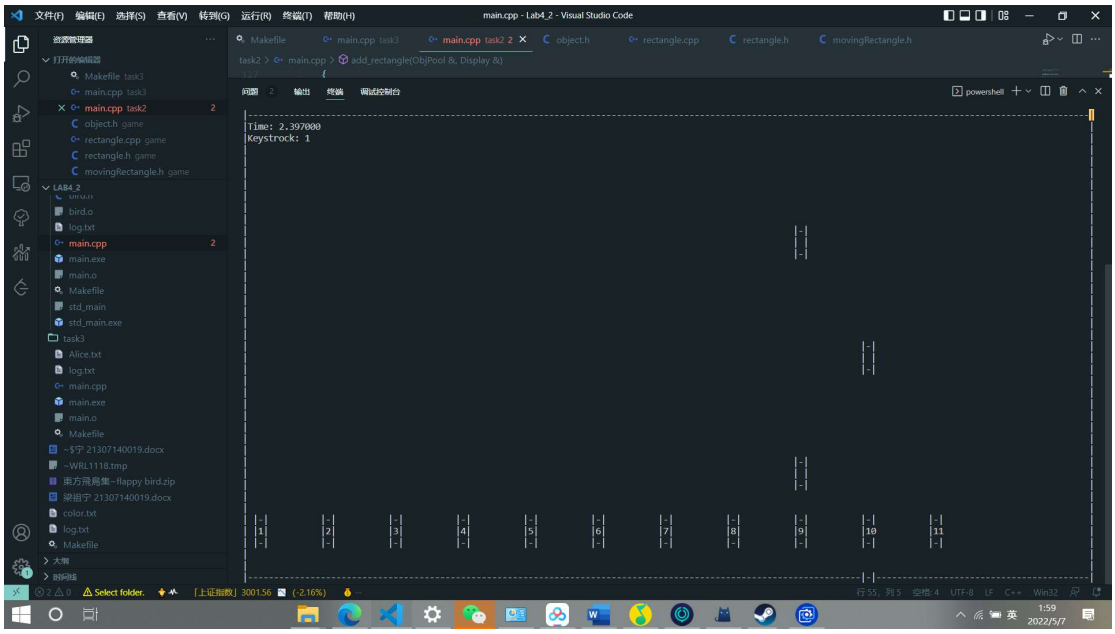
实现逻辑

1. 播放 bgm
2. 设置变量 prec 记录上一次输入，若此次输入为空格且上一次不是，则播放 flap。
3. 设置变量 times 用于记录循环次数。每次循环+1，每 40 次，调用一次 add_rectangle.
4. Add_rectangle 逻辑：生成一个横坐标在最右侧，纵坐标随机的 movingrectangle，x 轴速度为-1.并添加到 obj_pool 中
5. 所有的 objpool 中的元素，tick 一次，进行本次时间单位的位置变化。

6. Obj_pool 中,将 border 与所有的 moving rectangle 检测一下是否碰撞。
7. Obj_pool 中，将 bird 与 border 和所有的 moving rectangle 检测一下是否碰撞
8. 如果 bird 有碰撞，进入 show_game_over 函数
9. 遍历 obj_pool，如果 moving rectangle 有碰撞，在 obj_pool 中，删除对应元素
10. 在 objpool 中，将每个元素调用 draw 函数。

Task3 简单音游

截图



实现过程

基本上和 task2 原理相同。在音频，movingrectangle 和判定
首先在 audio 库中加入 c 大调音阶的音频库。
然后将 task2 的 add_rectangle 改为 11 等分 x 轴，每个节奏块宽度是 3，高度是 3。
设置一个读入谱子的函数，读入一个按照简谱写的 txt 文件，至 vector<int> music。
每 15 个 tick，读入一个 music 中的数字，按数字生成一个对应位置的 movingrectangle

在和 movingrectangle 相应的位置，屏幕底部，生成 ObjPool keys，作为判定线。

每次读入输入，按照 1-10 的顺序，根据输入播放对应的音符。