火眼金睛(一)

1.探索新知

1.1 基本游戏框架

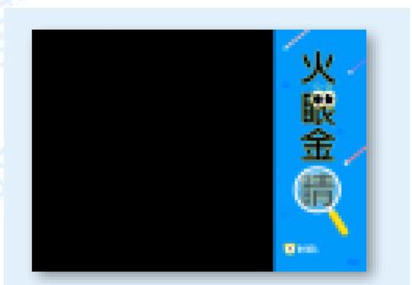
我们编写程序框架模板，设定一个新窗口

```
# 导入pgzero库
import pgzrun

# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 725  # 窗口的高度


# 启动游戏，写在所有程序的最后
pgzrun.go()
```

- **pgzrun** 就是 pygame zero run
- 窗口大小设定为 **1000 * 725**



任务一新增代码：

```
# （新增）导入Pgzero库
import pgzrun

# （新增）设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 725  # 窗口的高度

# （新增）启动游戏
pgzrun.go()
```

1.2 绘制背景

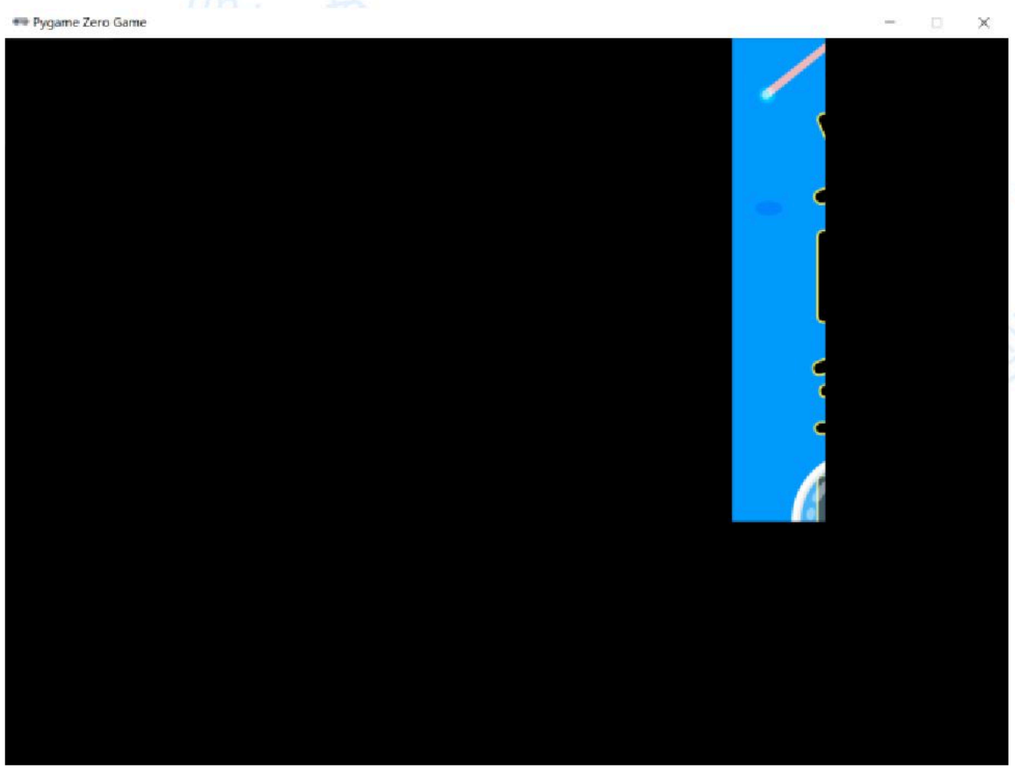
火眼金睛.png

项目类型: PNG 文件
分辨率: 1000 x 725
大小: 60.4 KB

前面我们已经学过如何绘制背景，试着自己完成吧！

```
def draw():
    _____________________________  # 绘制背景
```

运行后发现，窗口出现在屏幕的右下区域，拖到中间时背景没有显示完全。这是为什么呢？



draw()函数每秒刷新60次，需满足下面至少一个条件：

- 使用了 update() 函数
- 触发时钟事件

clock.schedule_interval(函数名，间隔)

- 触发其它输入事件（按键、鼠标等）

想要让背景完整呈现在我们眼前，可以添加一个update()函数

```
def draw():
    screen.blit('火眼金睛.png', (0,0))

def update():
    pass
```

1.2 绘制背景

任务二新增代码：

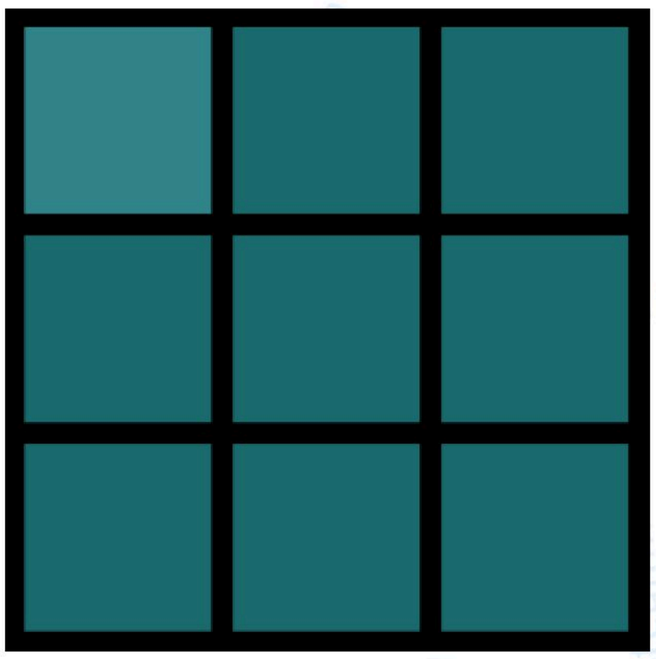
```
# 导入pgzero库
import pgzrun

# 设置窗口
WIDTH = 1000  # 窗口的宽度
HEIGHT = 725  # 窗口的高度

# (新增)
def draw():
    screen.blit('火眼金睛.png', (0,0))
def update():
    pass

# 启动游戏
pgzrun.go()
```

每个方格，都是长宽为 side 边长的矩形



【绘制实心矩形】

screen.draw.filled_rect(rect, color)

- rect 矩形对象
- color 颜色

接下来我们需要做两件事情：

1. 创建方格对象 rect
2. 配置颜色 color

1.3 创建方格对象

【创建矩形】

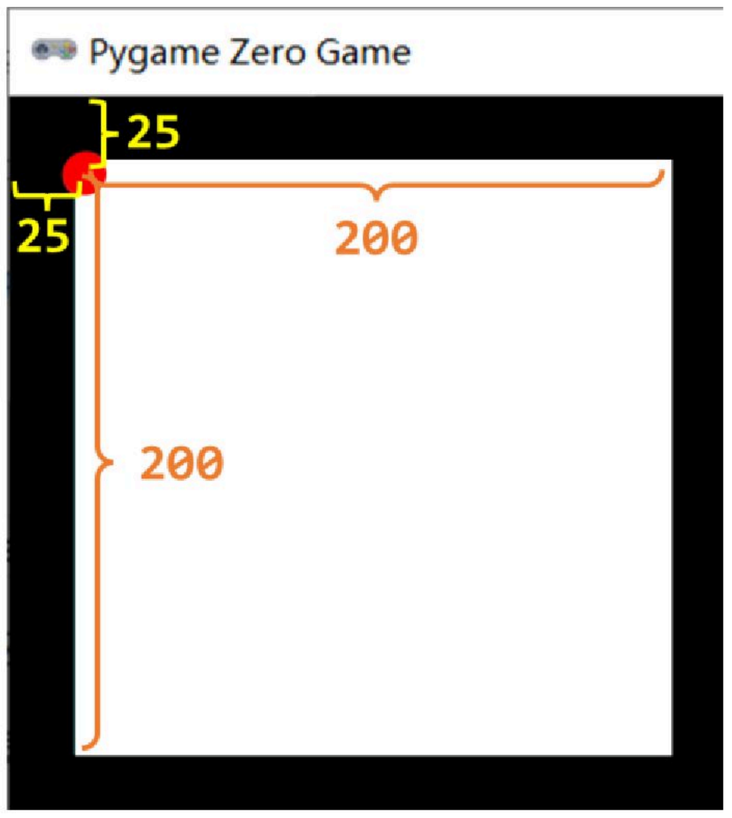
Rect(left, top, width, height)

- left、top 代表距离左边缘和上边缘的距离，即矩形左上角坐标。本课中即为 方格坐标
- width 、height 代表矩形的宽和高。本课中即为 方格边长

小练习：

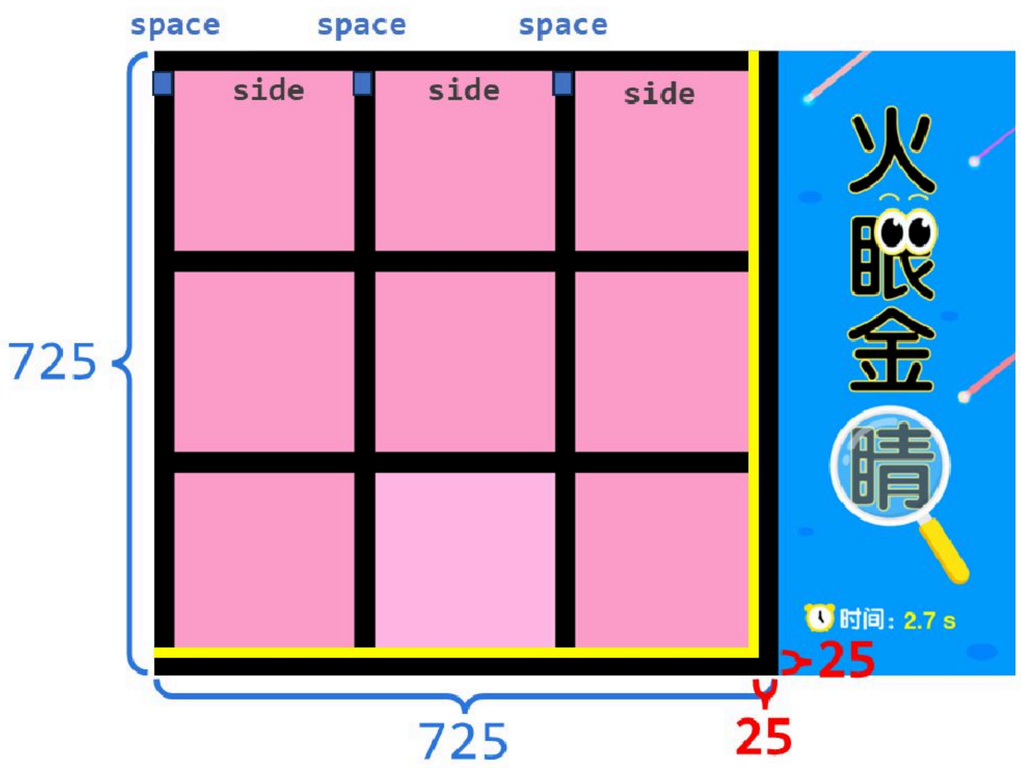
试着画出右侧的白色矩形

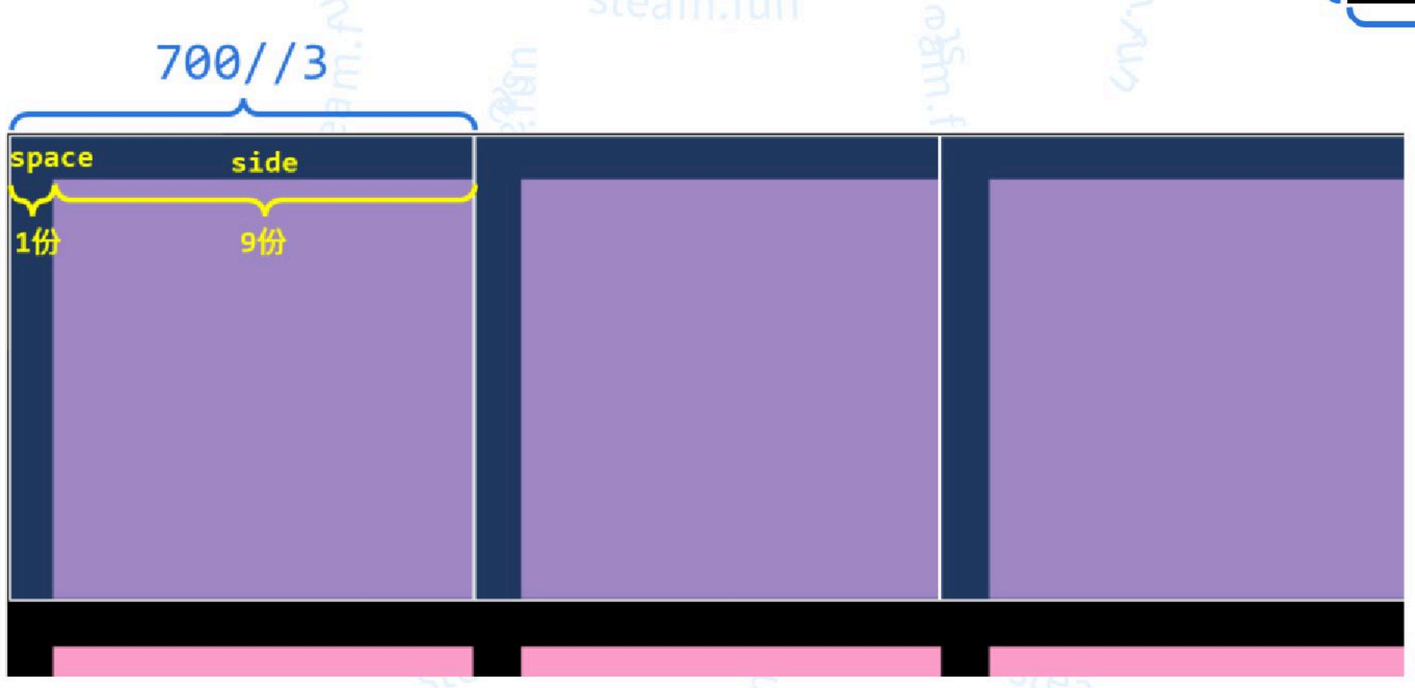
Rect(25, 25, 200, 200)




```
def draw():
    screen.blit('火眼金睛.png',(0,0))
```

游戏区域的面积恒定，为 725*725，
我们留25的固定间隔，用700来计算。



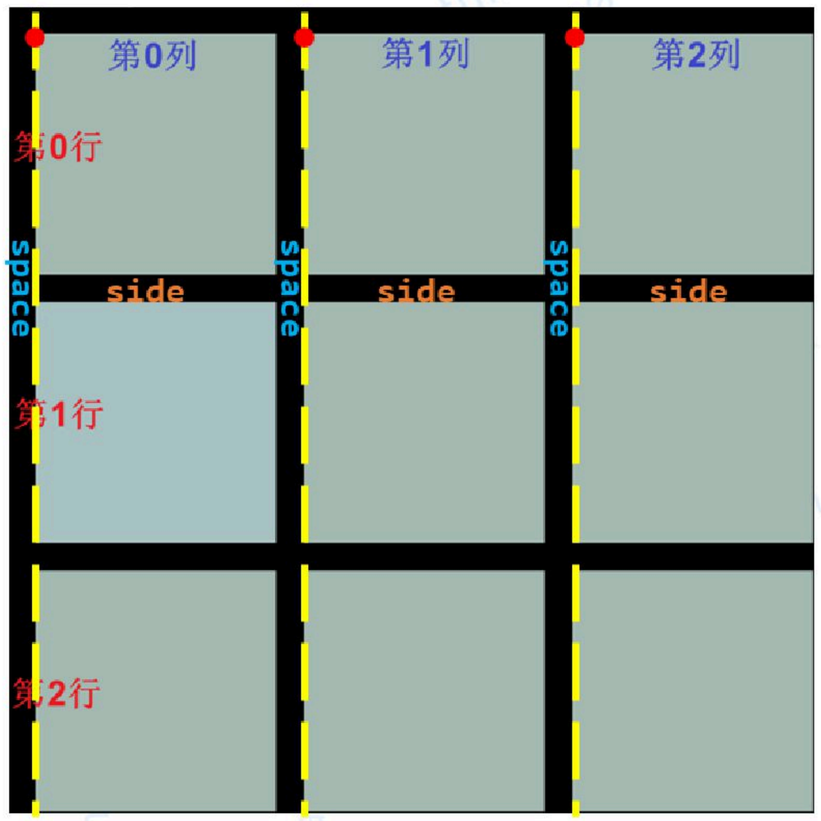


以 3*3 的方格矩阵为例

```
# 初始为3×3的矩阵
n = 3
# 计算格子边长和间距
side = 700// n * 0.9
space = 700// n * 0.1
```

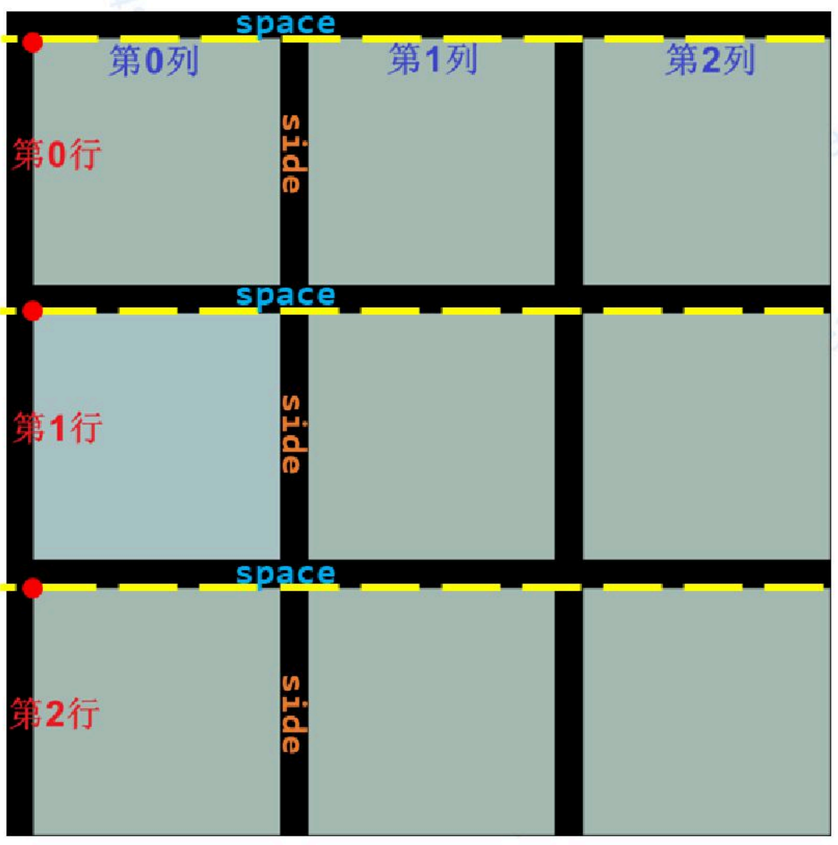
1.3 创建方格对象

【创建方格】



方格的x坐标

列	方格的x坐标
第0列	space
第1列	space + side + space
第2列	space + side + space + side + space
…	…
第col列	space + col * （side + space）



方格的y坐标

行	方格的y坐标
第0行	space
第1行	space + side + space
第2行	space + side + space + side + space
…	…
第row行	space + row * （side + space）

方格坐标

(space + col * (side + space), space + row * (side + space))

已知每个方格都有自己的坐标和颜色，创建字典info存储每个方格的信息

```
info = {
      'pos': (方格坐标),
      'color': (颜色元组)
      }
```

1.3 创建方格对象

```
# 根据格子数量计算格子边长和间距
side = 700// n * 0.9
space = 700// n * 0.1
for row in range(n):    # 遍历行
    for col in range(n):    # 遍历列
        info = {
            'pos': (space + col * (side + space),
            space + row * (side + space)),
            # 'color': 待定
            }
```

如何创建矩形对象?

```
Rect(info['pos'], (side, side))
```

创建列表 grids ，存储每个方格信息

```
grids = []    # 创建列表grids，后面存储每个方格信息
for row in range(n):    # 遍历行
    for col in range(n):    # 遍历列
        info = {
            'pos': (space + col * (side + space),
            space + row * (side + space)),
            # 'color': 待定
            }
        grids.append(info)
```

1.3 创建方格对象

我们此时想要绘制方格矩阵看看效果，只需要任意给矩形设置一个颜色即可

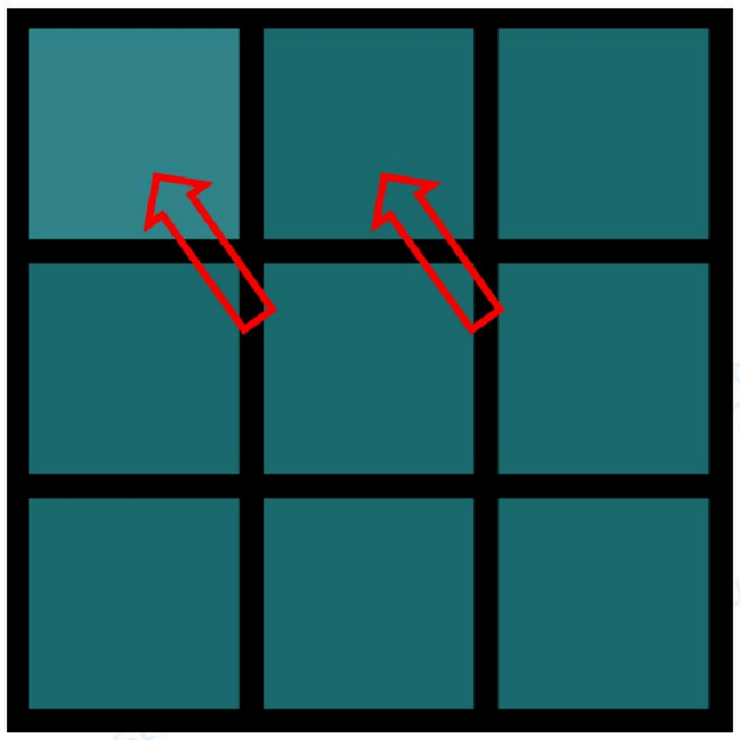
```
def draw():
    screen.blit('火眼金睛.png',(0,0))
    for grid in grids:
        screen.draw.filled_rect(Rect(grid['pos'], (side, side)), 'white')
```

任务三新增代码：

```
...
def draw():
    screen.blit('火眼金睛.png',(0,0))
    for grid in grids:  # (新增)绘制方格矩阵
        screen.draw.filled_rect(Rect(grid['pos'], (side, side)), 'white')
def update():
    pass
# (新增)初始为 3X3 的矩阵
n = 3
# (新增)根据格子数量计算格子边长和间距
side = 700// n * 0.9
space = 700// n * 0.1
# (新增)
grids = []    # 创建列表grids，后面存储每个方格信息
for row in range(n):   # 遍历行
    for col in range(n):    # 遍历列
        info = {
            'pos':(space + col * (side + space), space + row * (side + space)),
            'color': 'white'   # 测试用，后续及时更新对应颜色
            }
        grids.append(info)
pgzrun.go()
```

1.4 准备颜色

每一轮方格共有两种颜色，一种基础色，还有一个颜色轻微不同的异色



接下来我们需要做两件事情：

1. 生成随机基础色
2. 生成轻微异色

【生成随机基础色】

颜色元组（r，g，b），想要生成随机颜色，只需要生成随机的r、g、b参数即可

```
import random
...
def update():
    pass
def base_color():
    """生成随机基础颜色"""
    return ( random.randint(0, 255),
             random.randint(0, 255),
             random.randint(0, 255) )
```

【生成轻微异色】

想要生成与基础色轻微不同的随机颜色，也就是在基础色的r、g、b值上随机增加一个微小的随机值。

1.4 准备颜色

```
def dif_color(base):
    """
    参数:
    base: 表示基础颜色的元组，格式为（r,g,b）
    """
    r = base[0] + random.randint(20, 30)
    g = base[1] + random.randint(20, 30)
    b = base[2] + random.randint(20, 30)
    return (r, g, b)
```

base[0]、base[1]、base[2]的值在 0~255 范围内，随机增加 20~30，可能会出现r、g、b值超过255的情况

min()函数

返回给定参数的最小值，参数可以为序列

例：

min(2, 5, 0)　　返回值为 0

对 r、g、b 值用 min() 函数处理，如果值超过255则设置为255

```
def dif_color(base):
    """
    参数:
    base: 表示基础颜色的元组，格式为（r,g,b）
    """
    r = min(base[0] + random.randint(20, 30), 255)
    g = min(base[1] + random.randint(20, 30), 255)
    b = min(base[2] + random.randint(20, 30), 255)
    return (r, g, b)
```

1.4 准备颜色

任务四新增代码：

```
import random     # (新增)

...

def update():
    pass

def base_color():      # (新增)
    """生成随机基础颜色"""
    return ( random.randint(0, 255),
             random.randint(0, 255),
             random.randint(0, 255) )


def dif_color(base):     # (新增)
    """生成轻微异色"""
    r = min(base[0] + random.randint(20, 30), 255)
    g = min(base[1] + random.randint(20, 30), 255)
    b = min(base[2] + random.randint(20, 30), 255)
    return (r, g, b)
```

生成颜色的函数已经备好，
如何随机选择一个方块设置为异色？
我们下节课继续探索~

完整代码

```
import pgzrun
import random

WIDTH = 1000  # 窗口的宽度
HEIGHT = 725  # 窗口的高度

def draw():
    screen.blit('火眼金睛.png', (0, 0))
    for grid in grids:
        screen.draw.filled_rect(Rect(grid['pos'], (side, side)), 'white')
def update():
    pass
def base_color():
    """生成随机基础颜色"""
    return (random.randint(0, 255),
            random.randint(0, 255),
            random.randint(0, 255))
def dif_color(base):
    """生成轻微异色"""
    r = min(base[0] + random.randint(20, 30), 255)
    g = min(base[1] + random.randint(20, 30), 255)
    b = min(base[2] + random.randint(20, 30), 255)
    return (r, g, b)

# 初始为3×3的矩阵
n = 3
# 根据格子数量计算格子边长和间距
side = 700 // n * 0.9
space = 700 // n * 0.1
# 创建列表grids，后面存储每个方格信息
grids = []
for row in range(n):  # 遍历行
    for col in range(n):  # 遍历列
        info = {
            'pos': (space + col * (side + space), space + row * (side + space)),
            'color': 'white'  # 测试用，后续及时更新对应颜色
            }
        grids.append(info)

# 启动游戏
pgzrun.go()
```

2.强化练习

1．下面哪个函数是用于生成随机基础色的？（　　）

A. draw()

B. update()

C. base_color()

D. dif_color()

2．代码中 min()函数 的作用是什么？（　　）

```
def dif_color(base):
    r = min(base[0] + random.randint(20, 30), 255)
    g = min(base[1] + random.randint(20, 30), 255)
    b = min(base[2] + random.randint(20, 30), 255)
    return (r, g, b)
```

A．限制方格的边长

B．计算方格位置

C．限制颜色数值在0-255范围内

D．设置颜色为(255,255,255)

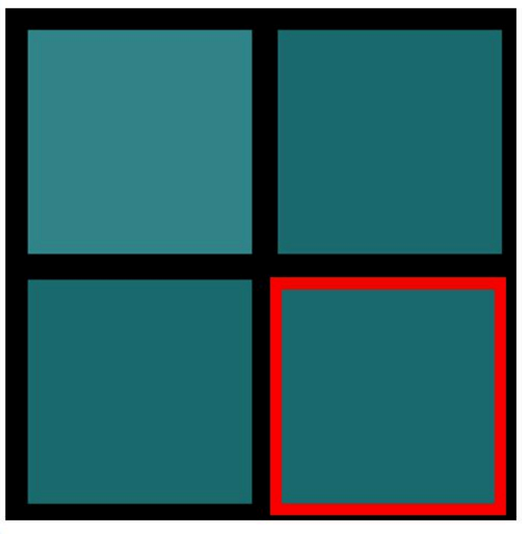
3．已知方块边长为20，间距为5，写出红框圈起来的方块坐标？（　　）

A. (20, 20)

B. (30, 30)

C. (40, 40)

D. (50, 50)



2.强化练习

4. 已知字典 jd = {'哈尔滨':['冰雪大世界','侵华日军第七三一部队罪证陈列馆', '太阳岛','中央大街'], '沈阳':['沈阳故宫','辽宁省博物馆','张学良旧居','清昭陵'], '长春':['伪满皇宫博物院','净月潭','长影世纪城','长春一汽']}，字典jd的长度是？（　　）

A. 3

B. 4

C. 12

D. 15

5. 用 Rect(100, 200, 50, 100) 创建了一个矩形，这个矩形的宽高分别是？（　　）

A. 100 200

B. 200 50

C. 50 100

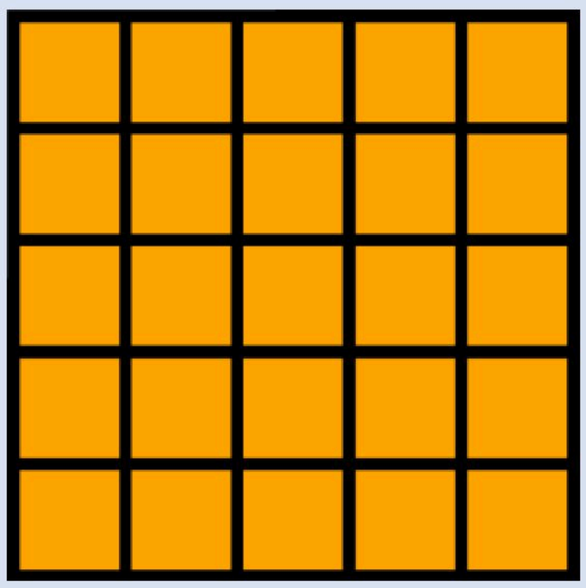
D. 100 100

3.术语箱

grid	格子	update	更新
row	行	rect	矩形（rectangle的缩写）
column	列		

4.课后挑战

五阶橙色矩阵



要求如下：

1. 方块边长为40，间距为5
2. 绘制五阶矩阵方块
3. 方块为橙色‘orange’