# 一、答题

1. 如何修改代码，根据你本人的姓（16\*16点阵），产生新的地图？

scene.py中Map类的stage1负责控制地图构造。代码中

1. for x in [2, 3, 6, 7, 18, 19, 22, 23]:
2. for y in [2, 3, 4, 5, 6, 7, 8, 9, 10, 17, 18, 19, 20, 21, 22, 23]:
3. brick = Brick()
4. brick.rect.left, brick.rect.top = 3 + x \* 24, 3 + y \* 24
5. brick.being = True
6. self.brickGroup.add(brick)
7. for x in [10, 11, 14, 15]:
8. for y in [2, 3, 4, 5, 6, 7, 8, 11, 12, 15, 16, 17, 18, 19, 20]:
9. brick = Brick()
10. brick.rect.left, brick.rect.top = 3 + x \* 24, 3 + y \* 24
11. brick.being = True
12. self.brickGroup.add(brick)
13. for x in [4, 5, 6, 7, 18, 19, 20, 21]:
14. for y in [13, 14]:
15. brick = Brick()
16. brick.rect.left, brick.rect.top = 3 + x \* 24, 3 + y \* 24
17. brick.being = True
18. self.brickGroup.add(brick)
19. for x in [12, 13]:
20. for y in [16, 17]:
21. brick = Brick()
22. brick.rect.left, brick.rect.top = 3 + x \* 24, 3 + y \* 24
23. brick.being = True
24. self.brickGroup.add(brick)
25. for x, y in [(11, 23), (12, 23), (13, 23), (14, 23), (11, 24), (14, 24), (11, 25), (14, 25)]:
26. brick = Brick()
27. brick.rect.left, brick.rect.top = 3 + x \* 24, 3 + y \* 24
28. brick.being = True
29. self.brickGroup.add(brick)

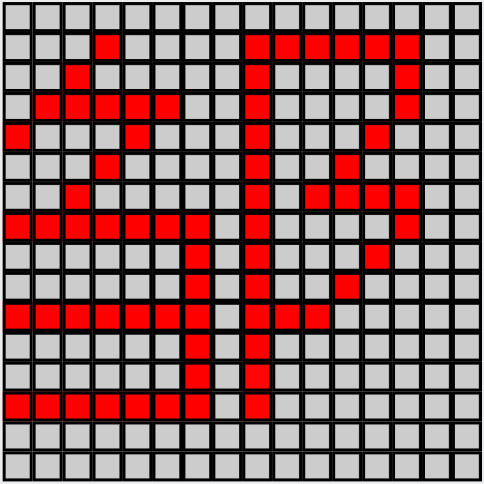
为地形坐标。如果想进行修改，将坐标改为姓氏的像素点即可。

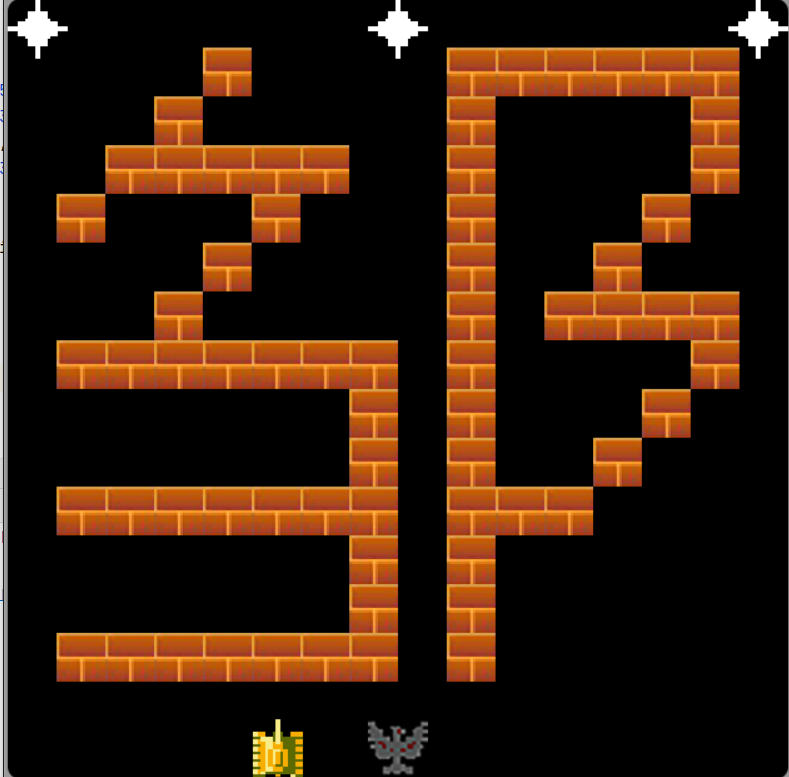
根据计算：

16\*39+3\*2=630

如果想满足16\*16的点阵，首先将图片尺寸修改为39\*39并重新加载。

接着，根据姓氏“邹”计算相应坐标点即可。





代码如下：

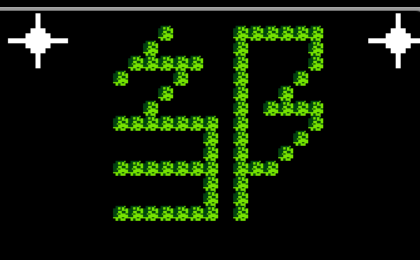
1. lst=[[4,1],[3,2],[2,3],[1,4],[3,3],[4,3],[5,3],[6,3],[5,4],[4,5],[3,6],[1,7],[2,7],[3,7],[4,7],[5,7],[6,7],[7,7],
2. [7,8],[7,9],[7,10],[7,11],[7,12],[7,13],[6,13],[5,13],[4,13],[3,13],[2,13],[1,13],[1,10],[2,10],[3,10],[4,10],
3. [5,10],[6,10],[9,1],[9,2],[9,3],[9,4],[9,5],[9,6],[9,7],[9,8],[9,9],[9,10],[9,11],[9,12],[9,13],[10,1],[11,1],
4. [12,1],[13,1],[14,1],[14,2],[14,3],[13,4],[12,5],[11,6],[12,6],[13,6],[14,6],[14,7],[13,8],[12,9],[11,10],[10,10]]
5. for itm in lst:
6. brick = Brick()
7. brick.rect.left, brick.rect.top = 3 + itm[0] \* 39, 3 + itm[1] \* 39
8. brick.being = True
9. self.brickGroup.add(brick)
10. 如何给司令部加钢板保护？

调Map类的self.protect\_home()。此函数会根据坐标在大本营附近生成钢墙。

1. def protect\_home(self):
2. for x, y in [(11, 23), (12, 23), (13, 23), (14, 23), (11, 24), (14, 24), (11, 25), (14, 25)]:
3. iron = Iron()
4. iron.rect.left, iron.rect.top = 3 + x \* 24, 3 + y \* 24
5. iron.being = True
6. self.ironGroup.add(iron)

此段代码为钢墙的碰撞检测。当检测到钢墙与子弹碰撞时，若为强力子弹，则销毁子弹本身和墙体，若为普通子弹，则仅仅销毁子弹

1. if each.bullet.stronger:
2. if pygame.sprite.spritecollide(each.bullet, stage\_data.map\_stage.ironGroup, True, None):
3. each.bullet.being = False
4. *# added, 2022.01.15*
5. stage\_data.enemybulletsGroup.remove(each.bullet)
6. else:
7. if pygame.sprite.spritecollide(each.bullet, stage\_data.map\_stage.ironGroup, False, None):
8. *# added, 2022.01.15*
9. each.bullet.being = False
10. stage\_data.enemybulletsGroup.remove(each.bullet)
11. 如何用12\*12的草地来构造游戏地图？
12. 首先将原图7\*7修改为12\*12，保存为1.png
13. 修改secnce类中的brick类图片属性为1.png
14. class Brick(pygame.sprite.Sprite):
15. def \_\_init\_\_(self):
16. pygame.sprite.Sprite.\_\_init\_\_(self)
17. self.brick = pygame.image.load('./images/scene/1.png')
18. self.rect = self.brick.get\_rect()
19. self.being = False
20. 采用以下代码即可绘制
21. lst = [[4, 1], [3, 2], [2, 3], [1, 4], [3, 3], [4, 3], [5, 3], [6, 3], [5, 4], [4, 5], [3, 6], [1, 7], [2, 7],[3, 7], [4, 7], [5, 7], [6, 7], [7, 7],[7, 8], [7, 9], [7, 10], [7, 11], [7, 12], [7, 13], [6, 13], [5, 13], [4, 13], [3, 13], [2, 13], [1, 13], [1, 10], [2, 10], [3, 10], [4, 10],[5, 10], [6, 10], [9, 1], [9, 2], [9, 3], [9, 4], [9, 5], [9, 6], [9, 7], [9, 8], [9, 9], [9,10], [9, 11], [9, 12], [9, 13], [10, 1], [11, 1],[12, 1], [13, 1], [14, 1], [14, 2], [14, 3], [13, 4], [12, 5], [11, 6], [12, 6], [13,6],[14, 6], [14, 7], [13, 8], [12, 9], [11, 10], [10, 10]]
22. for itm in lst:
23. brick=Brick()
24. brick.rect.left,brick.rect.top=3+(itm[0]+30) \* 12,3+itm[1] \* 12
25. brick.being=True
26. self.brickGroup.add(brick)



(4) 如何设置我方坦克，使其进入无敌模式？

tanks.py中设置myTank的self.protected=True。

此段代码用于检测是否坦克具有protect属性，当为false时，设置生命等属性；其余情况，则仅设置子弹被移除。

1. for tank\_player in stage\_data.mytanksGroup:
2. if pygame.sprite.collide\_rect(each.bullet, tank\_player):
3. *# 若坦克没有防护罩*
4. if not tank\_player.protected:
5. sound\_data.bang\_sound.play()
6. tank\_player.life -= 1
7. if tank\_player.life < 0:
8. stage\_data.mytanksGroup.remove(tank\_player)
9. stage\_data.tanksGroup.remove(tank\_player)
10. *# 当双打时, 两个玩家中, 可能还有一个玩家仍然没有挂掉*
11. *# 只有两个玩家都挂掉后, 才game over*
12. if len(stage\_data.mytanksGroup) < 1:
13. is\_gameover = True
14. else:
15. *# 当还有可用坦克时, 重新复位一下当前玩家的坦克*
16. tank\_player.reset()
17. each.bullet.being = False
18. stage\_data.enemybulletsGroup.remove(each.bullet)
19. break
20. 我方坦克在移动时，如何进行碰撞检测？

此代码为条件控制。

1. self.rect.top < 3 or self.rect.bottom > 630 - 3 or \
2. self.rect.left < 3 or self.rect.right > 630 - 3 or \
3. pygame.sprite.spritecollide(self, brickGroup, False, None) or \
4. pygame.sprite.spritecollide(self, ironGroup, False, None) or  \
5. pygame.sprite.spritecollide(self, tankGroup, False, None) or \
6. pygame.sprite.collide\_rect(self, myhome)

使用tanks.py中myTank的self.detect\_and\_undo()方法进行检测，原理为调用pygame.sprite.spritecollide对精灵进行检测。当检测到碰撞时，使移动状态变为反向，从而实现碰撞物体停止移动的效果。

1 其他坦克：条件为是否碰撞坦克

2 砖块：条件为是否碰撞砖块

3 铁板：条件为是否碰撞铁板

4 大本营：条件为是否超出坐标范围

1. 如何发现我方坦克被敌方子弹击中？

调用main.py中handle\_enemy\_bullet()函数。

当地方子弹撞击到我方坦克时，首先将生命值减一，如果生命值小于0，则使用remove()方法移除我方坦克。

1. 我方一辆坦克被击毁后，如何进行重新复位？

调用myTank类的reset()方法，设置坦克坐标、子弹等属性变为初始状态。

1. 敌方坦克的移动方向是如何进行随机变化？

调用enemyTank类中的random\_direction()方法随机生成方向，接着利用enemyTank类中的move()方法随机生成移动距离，并且设置墙体碰撞判断。

1. def random\_direction(self):
2. self.direction\_x, self.direction\_y = random.choice(([0, 1], [0, -1], [1, 0], [-1, 0]))
3. *# 随机移动*
4. def move(self, tankGroup, brickGroup, ironGroup, myhome):
5. *# 1%的概率, 随机改变运动方向*
6. if random.randint(0, 99) == 0:
7. self.random\_direction()
8. *# 根据坦克当前朝向, 设置其外观, self.tank对应的图片(96\*192), 包含了8张坦克(48\*48)的外观*
9. if self.direction\_x == 0 and self.direction\_y == -1:        *# UP*
10. self.tank\_0 = self.tank.subsurface((0, 0), (48, 48))
11. self.tank\_1 = self.tank.subsurface((48, 0), (48, 48))
12. elif self.direction\_x == 0 and self.direction\_y == 1:       *# DOWN*
13. self.tank\_0 = self.tank.subsurface((0, 48), (48, 48))
14. self.tank\_1 = self.tank.subsurface((48, 48), (48, 48))
15. elif self.direction\_x == -1 and self.direction\_y == 0:      *# LEFT*
16. self.tank\_0 = self.tank.subsurface((0, 96), (48, 48))
17. self.tank\_1 = self.tank.subsurface((48, 96), (48, 48))
18. elif self.direction\_x == 1 and self.direction\_y == 0:       *# RIGHT*
19. self.tank\_0 = self.tank.subsurface((0, 144), (48, 48))
20. self.tank\_1 = self.tank.subsurface((48, 144), (48, 48))
21. else:
22. raise ValueError('enemyTank class -> direction value error.')
23. 敌我坦克移动时，履带的动画特效是用哪些代码来完成的？
24. 调用 self.tank.subsurface()对图片进行切割
25. self.tank\_0 = self.tank.subsurface((0, 0), (48, 48))
26. self.tank\_1 = self.tank.subsurface((48, 0), (48, 48))

2、设置图片翻转，制作动效

1. if stage\_data.time == 5:
2. stage\_data.time = 0
3. stage\_data.is\_switch\_tank = not stage\_data.is\_switch\_tank

3、我方坦克通过切换图片制作效果

1. def handle\_mytanksGroup(stage\_data, screen):
2. if stage\_data.tank\_player1 in stage\_data.mytanksGroup:
3. *# 当玩家按下方向键, 即坦克在移动时, 通过切换两张不同的外观, 来显示坦克履带的运动的特效*
4. if stage\_data.is\_switch\_tank and stage\_data.player1\_moving:
5. screen.blit(stage\_data.tank\_player1.tank\_0, (stage\_data.tank\_player1.rect.left, stage\_data.tank\_player1.rect.top))
6. stage\_data.player1\_moving = False
7. else:
8. screen.blit(stage\_data.tank\_player1.tank\_1, (stage\_data.tank\_player1.rect.left, stage\_data.tank\_player1.rect.top))

4、敌方坦克通过切换图片制作效果

1. if stage\_data.is\_switch\_tank:
2. screen.blit(each.tank\_0, (each.rect.left, each.rect.top))
3. else:
4. screen.blit(each.tank\_1, (each.rect.left, each.rect.top))

(10) 我方坦克进入无敌模式时，保护罩的绘制是用哪些代码来完成的？

1、当处于保护模式时，设置保护罩图片

1. if stage\_data.tank\_player1.protected:
2. screen.blit(stage\_data.tank\_player1.protected\_mask1, (stage\_data.tank\_player1.rect.left, stage\_data.tank\_player1.rect.top))
3. TankWar的代码中，是如何记录上，下，左，右的方向的？
4. 初始化键盘监听事件
5. key\_pressed = pygame.key.get\_pressed()
6. *# 处理玩家一相关键盘事件*
7. handle\_player1\_events(key\_pressed, sound\_data, stage\_data)
8. *# 处理玩家二相关键盘事件*
9. handle\_player2\_events(key\_pressed, sound\_data, stage\_data)
10. 判断对应按键是否按下，并处理键盘监听事件
11. def handle\_player1\_events(key\_pressed, sound\_data, stage\_data):
12. *# 用户只要有按住方向键, 则认为tank处于moving状态, 所以设置player1\_moving为true*
13. if key\_pressed[pygame.K\_w]:
14. stage\_data.tanksGroup.remove(stage\_data.tank\_player1)
15. stage\_data.tank\_player1.move\_up(stage\_data.tanksGroup, stage\_data.map\_stage.brickGroup,
16. stage\_data.map\_stage.ironGroup, stage\_data.myhome)
17. stage\_data.tanksGroup.add(stage\_data.tank\_player1)
18. stage\_data.player1\_moving = True
19. elif key\_pressed[pygame.K\_s]:
20. stage\_data.tanksGroup.remove(stage\_data.tank\_player1)
21. stage\_data.tank\_player1.move\_down(stage\_data.tanksGroup, stage\_data.map\_stage.brickGroup,
22. stage\_data.map\_stage.ironGroup, stage\_data.myhome)
23. stage\_data.tanksGroup.add(stage\_data.tank\_player1)
24. stage\_data.player1\_moving = True
25. elif key\_pressed[pygame.K\_a]:
26. stage\_data.tanksGroup.remove(stage\_data.tank\_player1)
27. stage\_data.tank\_player1.move\_left(stage\_data.tanksGroup, stage\_data.map\_stage.brickGroup,
28. stage\_data.map\_stage.ironGroup, stage\_data.myhome)
29. stage\_data.tanksGroup.add(stage\_data.tank\_player1)
30. stage\_data.player1\_moving = True
31. elif key\_pressed[pygame.K\_d]:
32. stage\_data.tanksGroup.remove(stage\_data.tank\_player1)
33. stage\_data.tank\_player1.move\_right(stage\_data.tanksGroup, stage\_data.map\_stage.brickGroup, stage\_data.map\_stage.ironGroup, stage\_data.myhome)
34. stage\_data.tanksGroup.add(stage\_data.tank\_player1)
35. stage\_data.player1\_moving = True
36. TankWar的代码中，如何进行图片的加载？

1、创建图片路径列表

1. self.bullets = ['./images/bullet/bullet\_up.png', './images/bullet/bullet\_down.png', './images/bullet/bullet\_left.png', './images/bullet/bullet\_right.png']

2、使用 pygame.image.load()加载图片

1. self.bullet = pygame.image.load(self.bullets[0])
2. TankWar的代码中，如何从加载的图片中取某个子图片？

使用subsurface()方法截取图片

1. self.tank\_1 = self.tank.subsurface((48, 0), (48, 48))
2. 如何处理我方坦克的上、下、左或者右的移动？

修改坐标，并在此坐标创建新对象。

1、监听键盘事件，并移除对象

1. stage\_data.tanksGroup.remove(stage\_data.tank\_player1)
2. stage\_data.tank\_player1.move\_down(stage\_data.tanksGroup, stage\_data.map\_stage.brickGroup,

2、调用对应移动函数，修改坐标

1. *# 向上*
2. def move\_up(self, tankGroup, brickGroup, ironGroup, myhome):
3. return self.do\_move(tankGroup, brickGroup, ironGroup, myhome,
4. 0, -1,
5. (0, 0), (48, 48),
6. (48, 0), (48, 48))
7. *# 向下*
8. def move\_down(self, tankGroup, brickGroup, ironGroup, myhome):
9. return self.do\_move(tankGroup, brickGroup, ironGroup, myhome,
10. 0, 1,
11. (0, 48), (48, 48),
12. (48, 48), (48, 48))
13. *# 向左*
14. def move\_left(self, tankGroup, brickGroup, ironGroup, myhome):
15. return self.do\_move(tankGroup, brickGroup, ironGroup, myhome,
16. -1, 0,
17. (0, 96), (48, 48),
18. (48, 96), (48, 48))
19. *# 向右*
20. def move\_right(self, tankGroup, brickGroup, ironGroup, myhome):
21. return self.do\_move(tankGroup, brickGroup, ironGroup, myhome,
22. 1, 0,
23. (0, 144), (48, 48),
24. (48, 144), (48, 48))
25. 判断是否可以移动
26. def try\_move(self):
27. self.rect = self.rect.move(self.speed \* self.direction\_x, self.speed \* self.direction\_y)
28. *# 撤销移动(通过反向移动来实现)*
29. def undo\_move(self):
30. self.rect = self.rect.move(self.speed \* -self.direction\_x, self.speed \* -self.direction\_y)
31. def detect\_and\_undo(self, tankGroup, brickGroup, ironGroup, myhome):
32. *# 是否可以移动*
33. is\_move = True
34. *# 判断是否超出地图四个边界, 及是否与砖墙, 钢墙, 其他坦克及大本营碰撞*
35. if self.rect.top < 3 or self.rect.bottom > 630 - 3 or \
36. self.rect.left < 3 or self.rect.right > 630 - 3 or \
37. pygame.sprite.spritecollide(self, brickGroup, False, None) or \
38. pygame.sprite.spritecollide(self, ironGroup, False, None) or  \
39. pygame.sprite.spritecollide(self, tankGroup, False, None) or \
40. pygame.sprite.collide\_rect(self, myhome):
41. self.undo\_move()
42. is\_move = False
43. return is\_move
44. def do\_move(self, tankGroup, brickGroup, ironGroup, myhome, dx, dy, x0y0, w0h0, x1y1, w1h1):
45. *# 改变朝向*
46. self.direction\_x, self.direction\_y = dx, dy
47. *# 根据朝向, 重新设置坦克外观*
48. self.tank\_0 = self.tank.subsurface(x0y0, w0h0)
49. self.tank\_1 = self.tank.subsurface(x1y1, w1h1)
50. *# 先移动后判断*
51. self.try\_move()
52. return self.detect\_and\_undo(tankGroup, brickGroup, ironGroup, myhome)

4、添加新对象，并设置当前为移动状态

1. stage\_data.tanksGroup.add(stage\_data.tank\_player1)
2. stage\_data.player1\_moving = True
3. 哪种颜色的敌方坦克被击中后，会出现奖励(food)？是用哪些代码来处理的？

红色。当颜色为红色时，生成food。

1. if each.is\_red:
2. myfood = food.Food()
3. myfood.generate()
4. stage\_data.myfoodsGroup.add(myfood)
5. each.is\_red = False
6. 开机界面是由哪些代码来处理的？

调用show\_start\_interface()函数实现。

1. def show\_start\_interface(screen, width, height):
2. tfont = pygame.font.Font('./font/simkai.ttf', width//4)
3. cfont = pygame.font.Font('./font/simkai.ttf', width//20)
4. title = tfont.render(u'坦克大战', True, (255, 0, 0))
5. content1 = cfont.render(u'按1键进入单人游戏', True, (0, 0, 255))
6. content2 = cfont.render(u'按2键进入双人人游戏', True, (0, 0, 255))
7. trect = title.get\_rect()
8. trect.midtop = (width/2, height/4)
9. crect1 = content1.get\_rect()
10. crect1.midtop = (width/2, height/1.8)
11. crect2 = content2.get\_rect()
12. crect2.midtop = (width/2, height/1.6)
13. screen.blit(title, trect)
14. screen.blit(content1, crect1)
15. screen.blit(content2, crect2)
16. pygame.display.update()
17. while True:
18. for event in pygame.event.get():
19. if event.type == QUIT:
20. sys.exit()
21. elif event.type == pygame.KEYDOWN:
22. if event.key == pygame.K\_1:
23. return 1
24. if event.key == pygame.K\_2:
25. return 2
26. 关卡界面由哪些代码进行处理？如何修改代码，使关卡界面的显示时间持续5秒左右？

1、调用show\_switch\_stage()实现。

1. def show\_switch\_stage(screen, width, height, stage):
2. bg\_img = pygame.image.load("./images/others/background.png")
3. screen.blit(bg\_img, (0, 0))
4. font = pygame.font.Font('./font/simkai.ttf', width//10)
5. content = font.render(u'第%d关' % stage, True, (0, 255, 0))
6. rect = content.get\_rect()
7. rect.midtop = (width/2, height/2)
8. screen.blit(content, rect)
9. pygame.display.update()
10. delay\_event = pygame.constants.USEREVENT + 3
11. pygame.time.set\_timer(delay\_event, 1000, loops=1)
12. while True:
13. for event in pygame.event.get():
14. if event.type == QUIT:
15. sys.exit()
16. if event.type == delay\_event:
17. return

2、将以下代码

pygame.time.set\_timer(delay\_event, 1000, loops=1)

修改为

pygame.time.set\_timer(delay\_event, 5000, loops=1)

即可实现暂停5秒。

1. 如何处理我方坦克的开火？
2. 按键监听空格按下
3. if key\_pressed[pygame.K\_SPACE]:
4. if not stage\_data.tank\_player1.bullet.being:
5. sound\_data.fire\_sound.play()
6. stage\_data.tank\_player1.shoot()
7. 首先播放内置的开火声音
8. self.fire\_sound = pygame.mixer.Sound("./audios/fire.wav")
9. self.fire\_sound.set\_volume(1)

3、调用myTank类的shoot()方法

1. def shoot(self):
2. *# 在发出去的一发子弹, 没有到达边界以前, 不能再次发射*
3. self.bullet.being = True
4. stronger = self.bullet.stronger
5. *# 根据坦克的当前朝向和位置, 设置子弹的方向和位置, 子弹出现在坦克前方不远处*
6. self.bullet.turn(self.direction\_x, self.direction\_y)
7. if self.direction\_x == 0 and self.direction\_y == -1:   *# UP*
8. self.bullet.rect.left = self.rect.left + 20
9. self.bullet.rect.bottom = self.rect.top - 1
10. elif self.direction\_x == 0 and self.direction\_y == 1:  *# DOWN*
11. self.bullet.rect.left = self.rect.left + 20
12. self.bullet.rect.top = self.rect.bottom + 1
13. elif self.direction\_x == -1 and self.direction\_y == 0:  *# LEFT*
14. self.bullet.rect.right = self.rect.left - 1
15. self.bullet.rect.top = self.rect.top + 20
16. elif self.direction\_x == 1 and self.direction\_y == 0:   *# RIGHT*
17. self.bullet.rect.left = self.rect.right + 1
18. self.bullet.rect.top = self.rect.top + 20
19. else:
20. raise ValueError('myTank class -> direction value error.')
21. *# 坦克的等级提升后, 子弹的速度会加快, 提到2级及以上等级后, 可碎钢板*
22. if self.level == 0:
23. self.bullet.speed = 8
24. self.bullet.stronger = False
25. elif self.level == 1:
26. self.bullet.speed = 12
27. self.bullet.stronger = False
28. elif self.level == 2:
29. self.bullet.speed = 12
30. self.bullet.stronger = True
31. elif self.level == 3:     *# BUG, 图片只提供了0到2级的, 没有3级坦克的图片*
32. self.bullet.speed = 16
33. self.bullet.stronger = True
34. else:
35. raise ValueError('myTank class -> level value error.')
36. *# added, 2021.01.16*
37. if stronger:
38. self.bullet.stronger = stronger
39. 每一关对应的数据(坦克数量和地图等)是哪个类来管理的？

由main.py中的StageData类来管理。

(20) 敌方坦克出场时的特效在哪里处理？

由StageData类中以下代码处理。

1. appearance\_img = pygame.image.load("./images/others/appear.png").convert\_alpha()
2. self.appearances = []
3. self.appearances.append(appearance\_img.subsurface((0, 0), (48, 48)))
4. self.appearances.append(appearance\_img.subsurface((48, 0), (48, 48)))
5. self.appearances.append(appearance\_img.subsurface((96, 0), (48, 48)))
6. main()函数的游戏主循环的相关代码主要完成了哪些事情？
7. 初始化地方坦克信息，数量等属性
8. 初始化键盘监听事件
9. 绘制墙、树等地图
10. 初始化食物奖励、子弹等
11. 初始化关卡信息
12. 事件(消息)循环如何处理各种event？

使用pygame.event.get()获取事件队列，将event.type作为判断条件，进行不同情况的处理。

1. while True:
2. for event in pygame.event.get():
3. if event.type == QUIT:
4. sys.exit()
5. if event.type == delay\_event:
6. return
7. 如何播放声音？
8. 使用pygame.mixer.Sound(path).add\_sound.set\_volume(1)初始化并设置音量
9. class SoundData:
10. def \_\_init\_\_(self):
11. self.add\_sound = pygame.mixer.Sound("./audios/add.wav")
12. self.add\_sound.set\_volume(1)
13. 使用play()方法播放音乐
14. sound\_data = SoundData()
15. num\_player = show\_start\_interface(screen, 630, 630)
16. sound\_data.start\_sound.play()
17. game over 时，如何退出main()函数的游戏主循环？
18. 预置is\_gameover变量作为条件
19. 当符合游戏结束的条件时将is\_gameover修改为false并使用break跳出循环
20. if is\_gameover:
21. break
22. *# 敌方库存坦克数为0 且战场上坦克也为0*
23. if stage\_data.enemytanks\_total < 1 and stage\_data.enemytanks\_now < 1:
24. is\_gameover = False
25. break
26. 将main.py中的clock.tick(60)改为clock.tick(6)或者clock.tick(180)会有什么结果？

clock.tick(60)为控制游戏刷新率函数，参数为60代表一秒刷新60次。

1. 当参数为6：画面一秒钟刷新6次，比参数为60更为卡顿。
2. 当参数为180：画面一秒刷新180次，速度比60快一倍。
3. 我方坦克获得的保护罩的奖励如何在8秒后消失？

将handle\_food类中的代码修改为以下代码：

1. if myfood.kind == 4:
2. sound\_data.add\_sound.play()
3. for t in stage\_data.mytanksGroup:
4. t.protected = True
5. pygame.time.set\_timer(stage\_data.noprotectMytankEvent, 8000, loops=1)
6. 我方坦克如何获得各种奖励(food)？如何修改代码，使得所有生成的敌方坦克都为红色，且奖励都为保护罩？
7. Food类中的属性foods定义了可生成的food种类
8. self.foods = [self.food\_bomb, self.food\_clock, self.food\_gun, self.food\_iron, self.food\_protect, self.food\_star, self.food\_tank]

2、generate()方法可随机生成不同种类的food和food对应的坐标

1. def generate(self):
2. self.kind = random.randint(0, 6)
3. self.food = pygame.image.load(self.foods[self.kind]).convert\_alpha()
4. self.rect = self.food.get\_rect()
5. self.rect.left, self.rect.top = random.randint(100, 500), random.randint(100, 500)
6. self.being = True
7. 使得所有生成的敌方坦克都为红色，且奖励都为保护罩
   1. 修改food类的generate()方法为如下代码
8. def generate(self):
9. *#self.kind = random.randint(0, 6)*
10. self.kind = 4
11. self.food = pygame.image.load(self.foods[self.kind]).convert\_alpha()
12. self.rect = self.food.get\_rect()
13. self.rect.left, self.rect.top = random.randint(100, 500), random.randint(100, 500)
14. self.being = True
    1. 修改tanks.py，最后加上self.is\_red = True
15. class enemyTank(pygame.sprite.Sprite):
16. def \_\_init\_\_(self, x=None, kind=None, is\_red=None):
17. pygame.sprite.Sprite.\_\_init\_\_(self)
18. *# 用于给刚生成的坦克播放出生特效*
19. self.born = True
20. *# 90个滴答? 根据tick()函数来决定真正所需的时间, 如果tick(60), 即每秒60个滴答(界面刷新), 则出生特效大概需要1.5秒*
21. self.times = 90
22. *# 敌方坦克的种类编号: 从0到3*
23. if kind is None:
24. self.kind = random.randint(0, 3)
25. else:
26. self.kind = kind
27. *# 所有敌方坦克: 共有4种kind, 同一kind, 内部也分为4种不同的颜色*
28. self.tanks1 = ['./images/enemyTank/enemy\_1\_0.png', './images/enemyTank/enemy\_1\_1.png', './images/enemyTank/enemy\_1\_2.png', './images/enemyTank/enemy\_1\_3.png']
29. self.tanks2 = ['./images/enemyTank/enemy\_2\_0.png', './images/enemyTank/enemy\_2\_1.png', './images/enemyTank/enemy\_2\_2.png', './images/enemyTank/enemy\_2\_3.png']
30. self.tanks3 = ['./images/enemyTank/enemy\_3\_0.png', './images/enemyTank/enemy\_3\_1.png', './images/enemyTank/enemy\_3\_2.png', './images/enemyTank/enemy\_3\_3.png']
31. self.tanks4 = ['./images/enemyTank/enemy\_4\_0.png', './images/enemyTank/enemy\_4\_1.png', './images/enemyTank/enemy\_4\_2.png', './images/enemyTank/enemy\_4\_3.png']
32. self.tanks = [self.tanks1, self.tanks2, self.tanks3, self.tanks4]
33. *# 是否携带食物(红色的坦克携带食物)*
34. if is\_red is None:
35. self.is\_red = random.choice((True, False, False, False, False))
36. else:
37. self.is\_red = is\_red
38. *# added*
39. self.is\_red = True
40. 整个游戏界面重绘时，哪条语句将整个界面清空？
41. screen.blit(bg\_img, (0, 0))
42. 游戏窗口标题栏上的提示信息是由哪些代码实现的？
43. pygame.display.set\_caption("坦克大战: 第{0}关, 我方库存坦克数 {1}, 敌方库存坦克数 {2}".format(
44. stage\_data.stage,
45. stage\_data.tank\_player1.life,
46. stage\_data.enemytanks\_total))

(30) 如何把游戏改为共6关，每关的界面依次为“澳 门 科 技 大 学”？

1、修改NUM\_OF\_STAGES属性为6，即可修改为6关

1. NUM\_OF\_STAGES = 6

2、界面标题修改

(1)修改show\_switch\_stage函数前半部分如下

1. def show\_switch\_stage(screen, width, height, stage):
2. bg\_img = pygame.image.load("./images/others/background.png")
3. screen.blit(bg\_img, (0, 0))
4. font = pygame.font.Font('./font/simkai.ttf', width//10)
5. content = font.render(stage, True, (0, 255, 0))
   1. 修改main函数前半部分如下
6. def main():
7. global is\_gameover
8. pygame.init()
9. pygame.mixer.init()
10. screen = pygame.display.set\_mode((630, 630))
11. pygame.display.set\_caption("坦克大战")
12. *# 加载图片*
13. bg\_img = pygame.image.load("./images/others/background.png")
14. *# 加载音效*
15. sound\_data = SoundData()
16. *# 开始界面*
17. num\_player = show\_start\_interface(screen, 630, 630)
18. *# 播放游戏开始的音乐, 此函数播放音乐时, 不影响界面刷新*
19. sound\_data.start\_sound.play()
20. *# 关卡*
21. stage = 0
22. *#num\_stage = NUM\_OF\_STAGES*
23. *# 游戏是否结束*
24. is\_gameover = False
25. *# 时钟*
26. clock = pygame.time.Clock()
27. *# 主循环*
28. tittlelist=["澳","门","科","技","大","学"]
29. while not is\_gameover:
30. *# 关卡*
31. stage += 1
32. if stage > NUM\_OF\_STAGES:
33. break
34. show\_switch\_stage(screen, 630, 630, tittlelist[stage])

# 代码分析

除资源文件外，程序主要为以下模块：

bullet.py：子弹基础属性模块

food.py：物品奖励基础属性模块

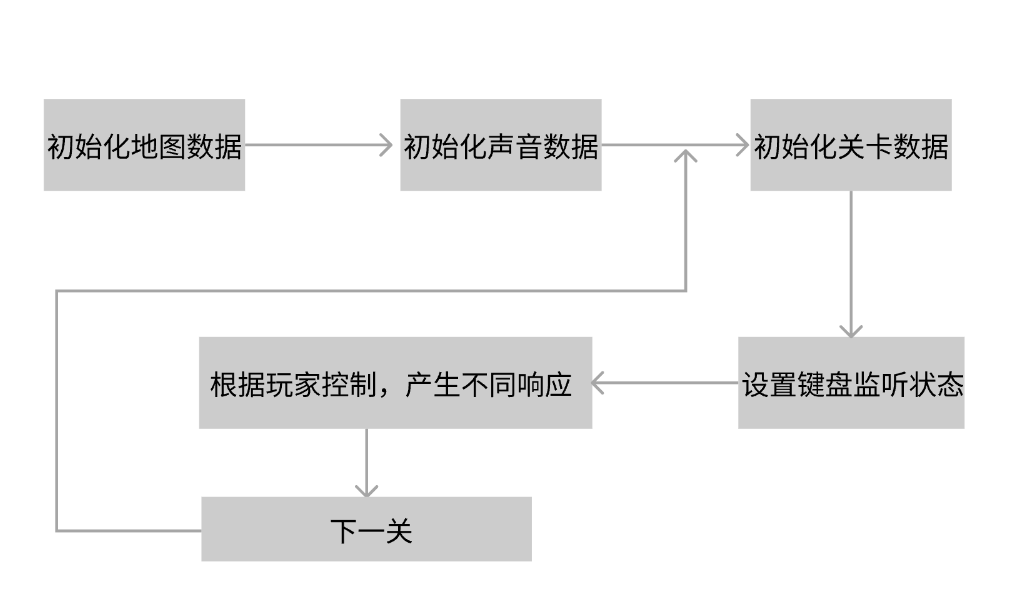
home.py：基地基础属性模块

main.py：程序入口

scene.py：地图属性模块

tanks.py：坦克属性模块

整个程序的框架大致如下：



## 一、bullet.py：子弹基础属性模块

类Bullet包含一个初始化方法、两个成员方法。

\_\_init\_\_()：类的初始化方法，用于初始化子弹的速度、外观等数据。

Ture()：以坐标向量的方式记录转向，通过修改direction\_x 和direction\_y，从而实现修改子弹。

Move()：用于实现当子弹运动超出范围时设置子弹消亡。

## 二、food.py：物品奖励基础属性模块

类Food包含一个初始化方法、一个成员方法。

\_\_init\_\_()：类的初始化方法，用于初始化奖励类型、奖励图标等数据。

generate()：用于生成奖励物品的坐标、种类。

## 三、home.py：基地基础属性模块

类Home包含一个初始化方法、一个成员方法。

\_\_init\_\_()：类的初始化方法，用于初始化大本营坐标、大本营图标等数据。

set\_dead()：用于设置大本营为摧毁状态。

## 四、scene.py：地图属性模块

该文件定义了6个类。

Brick：定义了地图砖块的初始化属性

Iron：定义了地图钢墙的初始化属性

Ice：定义了地图冰的初始化属性

River：定义了地图河流的初始化属性

Tree：定义了地图树的初始化属性

Map：定义了地图的初始化属性及地图样式

\_\_init\_\_()：初始化地图物品属性

set\_stage()：设置当前关卡采用的地铁

stage1()：地图一样式

stage2()：地图二样式

protect\_home()：设置主基地包围钢墙

该地图的设计模式参考了unity的tilemap，以瓦片地图的构造方式形成地图。

## 五、tanks.py：坦克属性模块

该文件定义了2个类。

myTank：定义了我方坦克的初始化数据

\_\_init\_\_()：初始化我方坦克外观、子弹、坐标等属性

Shoot()：设置子弹发射后的坐标、强度等属性

up\_level()：设置坦克升级

down\_level()：设置坦克降级

try\_move，undo\_move，detect\_and\_undo，do\_move，move\_up，move\_down，move\_left，move\_right：控制坦克移动、碰撞检测

Reset()：重置坦克为初始状态

enemyTank：定义了敌方坦克初始化数据

\_\_init\_\_

Shoot：设置子弹发射后的坐标、强度等属性

try\_move，undo\_move，random\_direction，move：设置地方坦克随机移动、碰撞检测

Reload：重置坦克被射击后的状态

## 六、main.py：程序入口

结构如下：

show\_start\_interface：函数定义了开始界面显示的字符

show\_end\_interface：函数定义了结束界面显示的字符

show\_switch\_stage：函数定义了切换关卡界面显示的字符

handle\_mytanksGroup：函数定义了我方坦克移动时候的动画特效、保护罩特效

handle\_enemytanksGroup：函数定义了敌方坦克出生时候的动画特效、暂停奖励特效

handle\_mybullet：函数定义了我方子弹攻击时撞击不同对象时的处理逻辑

handle\_enemy\_bullet：函数定义了敌方子弹攻击时撞击不同对象时的处理逻辑

handle\_food：函数定义了我方拣取物品奖励时的处理逻辑

handle\_player1\_events：玩家1的按键监听

handle\_player2\_events：玩家2的按键监听

SoundData：该类存放了初始化的声音数据

StageData：该类存放了初始化的关卡数据

Main：主程序的运行逻辑

## 总结

通过阅读坦克大战的源码，成功学习了pygame模块的使用方法，深刻的学习了如何使用pygame进行游戏开发。