# 说明：这个游戏的代码是我跟着老师学习的但是，老师把使用内容都放在一个文件里面，比较乱。我使用模块化开发，把代码和资源分开，比较简洁，是比较推荐的编程方式。

# 0.准备工作，

## 1.这个游戏需要turtle库，这个python已经自带了

## 2.这个游戏需要使用simpleaudio库，它现在也支持python3.10

### 安装simpleaudio：pip install simpleaudio，注意：如果报错说需要c++14 build tools在这里下载安装：[Microsoft C++ 生成工具 - Visual Studio](https://visualstudio.microsoft.com/zh-hans/visual-cpp-build-tools/)，安装后再运行一次：pip install simpleaudio

# 1.开发工作

## 1.在code文件夹里面新建一个game项目，然后新建一个resource文件夹在里面新建一个images文件夹和一个sounds文件夹，把游戏图片放进images文件夹里面，游戏声音文件放入sounds文件夹，然后在game文件夹下面新建一个game.py文件，如图

|  |
| --- |
|  |

## 2.编写游戏基本代码，加载游戏背景

|  |
| --- |
| import turtle  # 创建游戏窗口  game\_win = turtle.Screen()  # 设置窗口大小  game\_win.setup(600,600)  #设置窗口标题  game\_win.title("Space Invador Game")  # 设置背景图片  game\_win.bgpic('./resource/images/bg1.png')  # 游戏主循环  game\_win.mainloop() |

## 3.运行游戏，就会出现一个窗口

|  |
| --- |
|  |

## 4.turtle库的坐标系统，如图

|  |
| --- |
|  |

### 注意：他和pygame是不一样的，pygame的原点在屏幕左上角，turtle库的原点在屏幕中心，y轴的上面为+，下面为-也就是我们数学课的坐标系。这个和电脑的坐标系是不一样的。因为屏幕大小问题，我的游戏窗口设置为600x600.

## 5.创建玩家对象，除了窗口以外，所有的对象都是Turtle类创建的对象

|  |
| --- |
|  |

## 6.此时我们运行程序，就会看到游戏窗口里面有一个黑色箭头，它就是我们刚刚创建的玩家

|  |
| --- |
|  |

## 7.刚刚创建的玩家没有图片非常丑，我们给玩家设置图片，在turtle库中设置图片的方法叫shape，图片是一种“shape”

|  |
| --- |
|  |

## 8设置玩家的位置，这里和老师的可能不一样，移动位置之前，需要使用up方法把笔尖抬起来

|  |
| --- |
|  |

## 9.我们可以做一个处理，在设置玩家属性之前先把玩家隐藏，设置好属性后再显示

|  |
| --- |
| import turtle  pic\_path = './resource/images/'  # 创建游戏窗口  game\_win = turtle.Screen()  # 设置窗口大小  game\_win.setup(600,600)  #设置窗口标题  game\_win.title("Space Invador Game")  # 设置背景图片  game\_win.bgpic(pic\_path+'bg1.png')  #给turtle图片库添加图片  turtle.addshape(pic\_path+'player.gif')  # 创建玩家，它其实是一个Turtle类的对象  player = turtle.Turtle()  player.ht() #先隐藏玩家  # 给玩家设置属性  player.speed(0)  player.up() # 把笔尖抬起来，避免绘制一条直线  # 给玩家设置图片  player.shape(pic\_path+'player.gif')  player.setpos(0,-250)  # 把属性都设置好了就显示  player.st() #显示玩家  # 游戏主循环  game\_win.mainloop() |
|  |

## 10.让玩家动起来，这里需要开启turtle库监听功能而且需要根据按下的按键执行自定义的函数，为了方便使用，我们创建一个func.py模块在这里定义一些游戏需要用到的函数如这里的goLeft，我们先写一些测试代码，

|  |
| --- |
|  |

## 11.func.py里面的goLeft函数的内容如下

|  |
| --- |
| # 定义一些函数  import turtle  player = turtle.Turtle()  def goLeft():      print("moving left...") |

## 12，测试一下，先需要在游戏的开头导入func模块里面的所有函数

|  |
| --- |
|  |
|  |

### 可以看到，控制台有输出，说明监听正常

## 13.我们把玩家左右移动的函数都实现了，然后在game模块里面添加←和→方向键的按下事件处理

|  |
| --- |
|  |
|  |

### 为了方便函数的调用，我们把player的创建代码剪切到func模块里面了，现在就可以非常方便的左右移动玩家了

|  |
| --- |
|  |
|  |
|  |

## 14.创建敌人，需要注意的是必须将敌人放入一个敌人列表里面，而且必须用随机方法来计算敌人出现的位置

|  |
| --- |
|  |

## 15.敌人动起来

### 1）在func模块里面编写一个enemy\_move方法

|  |
| --- |
| # 让敌人动起来的方法  def enemy\_move():      inv\_step = 2 # 左右移动步长      sink\_step = 40 # y轴下沉步长，注意turtle是使用数学坐标系而不是屏幕坐标系      go\_back = False      # 敌人是组件运动的需要一个循环来实现      while True:          for inv in inv\_list:              x = inv.xcor()              x += inv\_step              inv.setx(x)              # 越界处理              if x > 250 or x < -250:                  # 往反方向动                  go\_back = True                  # print(go\_back)          if go\_back:              inv\_step = inv\_step \* -1              go\_back = False              # 碰到我们设置的限定边界需要往下沉              for inv in inv\_list:                  y = inv.ycor()                  y -= sink\_step                  inv.sety(y) |

### 2）在game模块里面调用敌人移动方法

|  |
| --- |
|  |

## 16.子弹的创建，先在func模块里面编写一个创建子弹的函数，需要在外面定义一个全局变量bomb=None，然后在函数里面使用global bomb来使用这个全局变量

|  |
| --- |
|  |

### 然后在game模块里面调用这个函数创建子弹

|  |
| --- |
| # import turtle  from func import \*  # 创建游戏窗口  game\_win = turtle.Screen()  # 设置窗口大小  game\_win.setup(600,600)  #设置窗口标题  game\_win.title("Space Invador Game")  # 设置背景图片  game\_win.bgpic(pic\_path+'bg1.png')  #给turtle图片库添加图片  turtle.addshape(pic\_path+'player.gif')  # 创建玩家，它其实是一个Turtle类的对象  # player = turtle.Turtle()  player.ht() #先隐藏玩家  # 给玩家设置属性  player.speed(0)  player.up() # 把笔尖抬起来，避免绘制一条直线  # 给玩家设置图片  player.shape(pic\_path+'player.gif')  player.setpos(0,-250)  # 把属性都设置好了就显示  player.st() #显示玩家  # 玩家移动  #1.开启turtle库的监听功能  turtle.listen()  # 按下←键的处理  turtle.onkey(goLeft,'Left') # goLeft是自定义函数，在func.py模块里面定义  # 按下→键的处理  turtle.onkey(goRight,'Right') # goRight是自定义函数，在func.py模块里面定义  # 创建子弹，这个版本的游戏只有一颗子弹  create\_bomb()  # 调用创建敌人的方法  create\_enemies()  # 让敌人动起来  enemy\_move()  # 游戏主循环  game\_win.mainloop() |

## 17子弹发射，子弹现在是隐藏状态，我们需要添加按键监听，设置当用户按下空格键就发射子弹，我们在func模块里面定义一个fire方法，用来发射子弹，其实这个方法只做这几个事情：设置子弹的位置并且显示子弹，将是否是发射状态的标记改为True，注意：一定要当is\_fired为False才继续发射操作，否则效果不好

|  |
| --- |
|  |

## 然后在game模块里面的监听事件定义fire方法

|  |
| --- |
|  |

### 现在的效果是：游戏启动，有玩家和敌人，子弹看不到，当玩家按下空格键，子弹出现在玩家上面一点的位置但是还不会动

|  |
| --- |
|  |
|  |

## 18.子弹移动，我们可以在敌人移动的函数里面添加子弹移动的功能，前提是需要判断一下is\_fired是否为True，如果是，子弹移动，如果不是，子弹不动

|  |
| --- |
| # 让敌人动起来的方法，这个方法又会让子弹动起来  def enemy\_move():      global bomb,is\_fired      inv\_step = 2 # 左右移动步长      sink\_step = 40 # y轴下沉步长，注意turtle是使用数学坐标系而不是屏幕坐标系      go\_back = False      # 敌人是组件运动的需要一个循环来实现      while True:          bomb\_step = 20 # 子弹位移          for inv in inv\_list:              x = inv.xcor()              x += inv\_step              inv.setx(x)              # 越界处理              if x > 250 or x < -250:                  # 往反方向动                  go\_back = True                  # print(go\_back)          if go\_back:              inv\_step = inv\_step \* -1              go\_back = False              # 碰到我们设置的限定边界需要往下沉              for inv in inv\_list:                  y = inv.ycor()                  y -= sink\_step                  inv.sety(y)          if is\_fired: #如果is\_fired为True，子弹需要动             y = bomb.ycor()             y += bomb\_step             bomb.sety(y)             # 子弹飞出边界，需要把is\_fired设置为False             if y > 250:                 is\_fired = False                 # 重新设置子弹位置                 bomb.setpos(-300,-300)                 bomb.ht() # 把子弹隐藏 |

## 19.子弹消灭敌人，如何在turtle库里面判断敌人中弹了?使用一个distance方法。我们可以在移动敌人的同时判断敌人是否中弹。修改enemy\_move函数

|  |
| --- |
|  |

## 注意：为了更方便我们理解代码，我把enemy函数改名为game\_proc

|  |
| --- |
|  |
|  |

## 20.添加游戏得分，在func模块里面添加一个显示得分的函数show\_score

|  |
| --- |
|  |

## 我们在func模块里面定义一个游戏得分初始值是0

|  |
| --- |
|  |

## 然后在game模块里面调用这个函数，需要在func模块里面创建一个pen全局变量，然后修改游戏game模块的代码

|  |
| --- |
|  |
|  |

### 效果：

|  |
| --- |
|  |

## 21.在游戏中更新分数，在打中敌人的时候处理需要重置敌人的位置和子弹的位置，就是需要修改并且显示分数

|  |
| --- |
|  |

## 效果：

|  |
| --- |
|  |
|  |
|  |

## 22.设置游戏结束的标准，其实很简单，就是只要又一个敌人下降到和玩家一样的高度，玩家就输掉游戏。我们先在func模块里面定义一个全局变量game\_over默认是False

|  |
| --- |
|  |

## 然后在game\_proc函数里面声明它是全局变量

|  |
| --- |
|  |

## 在敌人下降的代码里面添加判断游戏是否结束的代码，如果是把game\_over设置为True

|  |
| --- |
|  |

## 然后在while循环的一开始，就需要根据game\_over来判断是否退出循环

|  |
| --- |
|  |

## 23.设置音效，需要使用simpleaudio库，我们先把它导进来

|  |
| --- |
|  |

## 定义一个子弹发射的音效对象和一个爆炸的音效对象

|  |
| --- |
|  |

## 然后在fire方法里面调用laser对象的play方法播放音效

|  |
| --- |
|  |

## 在打中敌人的地方播放爆炸音效

|  |
| --- |
|  |

## 在game over的位置有需要播放爆炸音效

|  |
| --- |
|  |

### 此致，这个游戏开发完成，当然这个游戏还是比较简陋，以后有机会我想扩展它一下，下面是游戏的代码

## func.py

|  |
| --- |
| **# 定义一些函数**  **import turtle**  **import random**  **import simpleaudio as sa**  **pic\_path = './resource/images/' #图片文件夹路径**  **sound\_path = './resource/sounds/' #音效文件路径**  **player = turtle.Turtle()**  **player\_step = 15**  **num = 6 #敌人数量**  **inv\_list = [] # 敌人列表**  **bomb = None**  **is\_fired = False**  **score = 0 #游戏得分**  **pen = turtle.Turtle() # 用来写分数的笔对象**  **game\_over = False**  **#创建发射子弹和爆炸的音效**  **laser = sa.WaveObject.from\_wave\_file(sound\_path+'laser.wav') # 发射子弹**  **explosion = sa.WaveObject.from\_wave\_file(sound\_path+'explosion.wav') # 爆炸的音效**  **#创建敌人的方法**  **def create\_enemies():**  **turtle.addshape(pic\_path+'inv.gif') # 添加敌人图片到turtle库**  **for i in range(num):**  **inv = turtle.Turtle()**  **# 隐藏敌人**  **inv.ht()**  **# 设置敌人属性**  **inv.speed(0)**  **inv.up() # 使他移动时不会绘制直线**  **inv.shape(pic\_path+'inv.gif')**  **# 利用随机数计算敌人出现的位置**  **x = random.randint(-200,200)**  **y = random.randint(100,200)**  **inv.setpos(x,y)**  **inv.st() # 显示敌人**  **inv\_list.append(inv) # 把敌人添加到敌人列表**  **# 玩家往左移动**  **def goLeft():**  **# print("moving left...")**  **x = player.xcor() #获取玩家当前的坐标**  **x -= player\_step # 把它的坐标减去移动步长，因为往左边所有是减**  **# 越界处理**  **if x < -250:**  **x= -250**  **player.setx(x) # 重新设置玩家的x坐标，就实现了水平运动**  **# 玩家往右移动**  **def goRight():**  **# print("moving left...")**  **x = player.xcor() #获取玩家当前的坐标**  **x += player\_step # 把它的坐标减去移动步长，因为往左边所有是减**  **# 越界处理**  **if x > 250:**  **x= 250**  **player.setx(x) # 重新设置玩家的x坐标，就实现了水平运动**  **# def enemy\_move():**  **#游戏主逻辑函数，实现敌人和子弹移动，子弹打中敌人，和加分功能**  **def game\_proc():**  **global bomb,is\_fired,score,pen,game\_over**  **inv\_step = 2 # 左右移动步长**  **sink\_step = 40 # y轴下沉步长，注意turtle是使用数学坐标系而不是屏幕坐标系**  **go\_back = False**  **# 敌人是组件运动的需要一个循环来实现**  **while True:**  **if game\_over: #如果这个标志位True，就需要终止循环**  **# 在屏幕中间显示：Game Over**  **over\_pen = turtle.Turtle()**  **over\_pen.color('red')**  **over\_pen.up()**  **over\_pen.ht()**  **over\_pen.write('Game Over',align='center',font=('Arial',18,'bold'))**  **explosion.play() # 游戏结束相当于大本营给炸了，有需要播放爆炸音效**  **break**  **bomb\_step = 20 # 子弹位移**  **for inv in inv\_list:**  **x = inv.xcor()**  **x += inv\_step**  **inv.setx(x)**  **# 越界处理**  **if x > 250 or x < -250:**  **# 往反方向动**  **go\_back = True**  **# 用距离来判断敌人是否中弹**  **if inv.distance(bomb) < 15: # 如果距离小于15，我们就认为子弹打中敌人**  **# 打中后处理**  **#1.把敌人复位到尽量靠上面**  **inv.setpos(0,285)**  **# 播放爆炸音效**  **explosion.play()**  **# 2.把is\_fired设置为False这样子才能继续发射子弹**  **is\_fired = False**  **# 3.重置子弹的位置并且隐藏子弹**  **bomb.setpos(-300,-300)**  **bomb.ht()**  **# 更新分数并且显示**  **score += 10**  **show\_score(pen,score)**  **# 判断游戏是否结束**  **if inv.ycor() < -280:**  **game\_over = True**  **if go\_back:**  **inv\_step = inv\_step \* -1**  **go\_back = False**  **# 碰到我们设置的限定边界需要往下沉**  **for inv in inv\_list:**  **y = inv.ycor()**  **y -= sink\_step**  **inv.sety(y)**  **if is\_fired: #如果is\_fired为True，子弹需要动**  **y = bomb.ycor()**  **y += bomb\_step**  **bomb.sety(y)**  **# 子弹飞出边界，需要把is\_fired设置为False**  **if y > 250:**  **is\_fired = False**  **# 重新设置子弹位置**  **bomb.setpos(-300,-300)**  **bomb.ht() # 把子弹隐藏**  **def create\_bomb():**  **global bomb**  **bomb = turtle.Turtle()**  **bomb.ht()**  **bomb.up()**  **bomb.speed(0)**  **bomb.shape('triangle') # 使用turtle库自带的图片**  **bomb.color('yellow') # 设置颜色**  **bomb.shapesize(0.5,0.5) # 设置子弹尺寸**  **bomb.seth(90)**    **def fire(): # 发射子弹功能**  **global bomb,is\_fired,laser**  **if not is\_fired: # 设置不能连续发射，因为这样子视觉效果不好，子弹发射出去又被拉回来。**  **bomb.setpos(player.xcor(),player.ycor()+20) # 把子弹移动到玩家飞机的上方**  **bomb.st()**  **is\_fired = True**  **# 播放音效**  **laser.play()**  **# 显示游戏得分的函数**  **def show\_score(pen,score):**  **pen.color('white')**  **pen.speed(0)**  **pen.up()**  **pen.ht() #笔不需要显示**  **pen.setpos(-270,255)**  **pen.clear() # 在写新分数之前，需要去除旧分数，否则两个分数重叠在一起**  **pen.write(f"得分：{score}",align='left',font=("Arial",14,'normal'))** |

## game.py

|  |
| --- |
| **from func import \***  **# 创建游戏窗口**  **game\_win = turtle.Screen()**  **# 设置窗口大小**  **game\_win.setup(600,600)**  **#设置窗口标题**  **game\_win.title("Space Invader Game")**  **# 设置背景图片**  **game\_win.bgpic(pic\_path+'bg1.png')**  **#给turtle图片库添加图片**  **turtle.addshape(pic\_path+'player.gif')**  **# 创建玩家，它其实是一个Turtle类的对象**  **# player = turtle.Turtle()**  **player.ht() #先隐藏玩家**  **# 给玩家设置属性**  **player.speed(0)**  **player.up() # 把笔尖抬起来，避免绘制一条直线**  **# 给玩家设置图片**  **player.shape(pic\_path+'player.gif')**  **player.setpos(0,-250)**  **# 把属性都设置好了就显示**  **player.st() #显示玩家**  **# 玩家移动**  **#1.开启turtle库的监听功能**  **turtle.listen()**  **# 按下←键的处理**  **turtle.onkey(goLeft,'Left') # goLeft是自定义函数，在func.py模块里面定义**  **# 按下→键的处理**  **turtle.onkey(goRight,'Right') # goRight是自定义函数，在func.py模块里面定义**  **# 创建子弹，这个版本的游戏只有一颗子弹**  **create\_bomb()**  **# 添加游戏得分显示**  **show\_score(pen,score)**  **# 当用户按下空格键时发发射子弹**  **turtle.onkey(fire,'space')**  **# 调用创建敌人的方法**  **create\_enemies()**  **# 游戏主要业务逻辑：让敌人动起来，子弹打中敌人，加分等等功能**  **game\_proc()**  **# 游戏主循环**  **game\_win.mainloop()** |

# 麦叔其他游戏编程

猫和老鼠： [【Python游戏】跟麦叔学Python - 猫和老鼠\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av79334599/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)  
圣诞礼物： [Python小游戏 - 圣诞礼物来袭\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av80179874/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)

双人乒乓球： [用Python开发双人对战乒乓球小游戏\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av80942357/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)

新年时钟： [Python程序员打开新年的方法 - 开发1个时钟小应用\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av81912317/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)

迷宫： [用Python开发迷宫游戏，玩转Python面向对象「难度 L3」\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av83923241/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)

飞机大战： [【Python游戏】1小时开发飞机大战游戏-Pygame版本【课程编号210】\_哔哩哔哩\_bilibili](https://www.bilibili.com/video/av88696599/?vd_source=77c27fc74aec7a1c6789f9d7680904aa)