Explaining project 4

20191097 배호용

github : https://github.com/SGU-20191097-BHY/Project4-final-

Image source : My own(except for gun image). Sound Source : Here

1.Game play

(1)Controls

W : jump

A : move left

D : move right

R : reload current weapon

1 : switch weapon to pistol

2 : switch weapon to Assualt Rifle(AR)

3 : switch weapon to Sniper Rifle(SR)

Mouse Click : fire current weapon to mouse position.

Mouse Click when icon is red : use "concentration" skill

(2)Rules

1.Player has 500 HP.

2.Zombie have 100 HP.

3.Zombie gets close to player.

4.If zombie is too close to player, player gets 25 damage.

5.Zombie is generated every 0.5 second. At the door.

6.You can kill zombie by weapon.

7.Pistol has 40 damage. Shoot delay is 0.4 second. One magazine carries 12 bullets. Ammo is infinitely provided. Reload time is 1 sec.

8.AR has 30 damage. Shoot delay is 0.1 second. One magazine carries 30 bullets. 150 bullets(include magazine's) are provided at first. Reload time is 1 sec.

9.SR has 100 damage. Shoot delay is 1 second. One magazine carries 7 bullets. 35 bullets(include magazine's) are provided at first. SR's bullet can penetrate 2 zombies. So, one bullet can kill 3 zombies. Reload time is 1.5 sec.

10.Every weapon's bullet is flying object. Bullet flies from player to mouse-clicked position. If bullet collide with zombie or object(that can step on), bullet disappears(except for SR's) and zombie get damage.

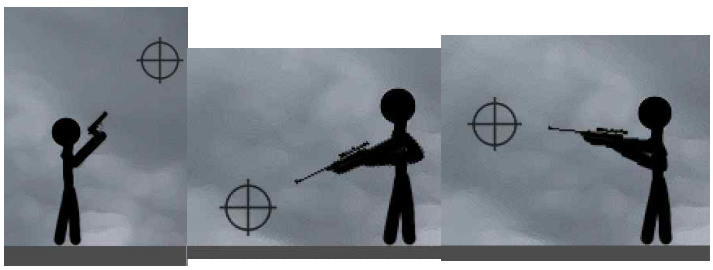
11."Concentration" is a skill. 2 conditions should be satisfied to use this skill. (1)It is not in cool down. (2)Mouse cursor is over zombie. If you use this skill, the zombie under the mouse die immediately. This skill ignores walls. Of course, 1 bullet is needed.

12.If you use "Concentration" with SR, skill is activated and one normal bullet flies

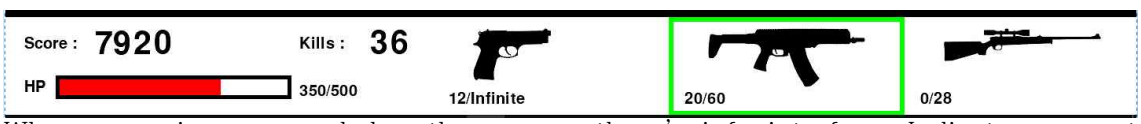
too. So, you can kill 4 zombies(1 by skill, 3 by normal bullet).
13.Damage done will be added to score. 1 zombie can give 100 score by damage.
14.One zombie kill by normal bullet pluses 100 score.
15.One zombie kill by Concentration pluses 150 score.

(3)UI

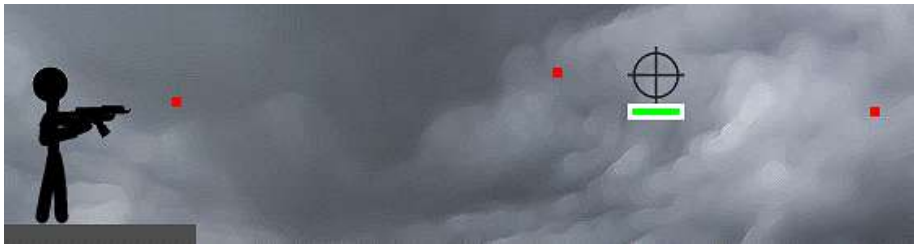
This is title screen. Center is game name. Below the name, there's last game's score. If program was just ran, it's 0. You can move player character at this time. But cannot fire.



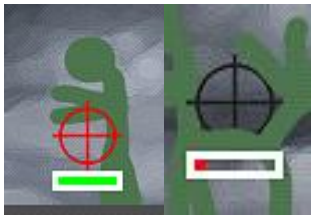
Player character aims where the mouse cursor is. Detail expression.
If you press Space, game starts.



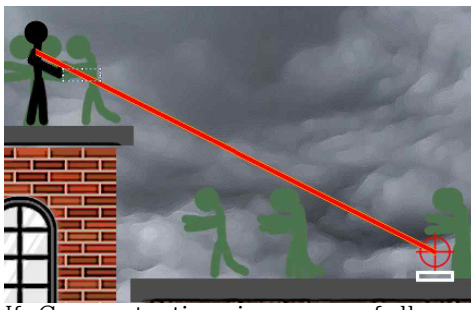
When game is now on, below the screen, there's info interface. Indicates current score, kills, player's HP and weapon's status. Currently using weapon is indicated by green rectangle.



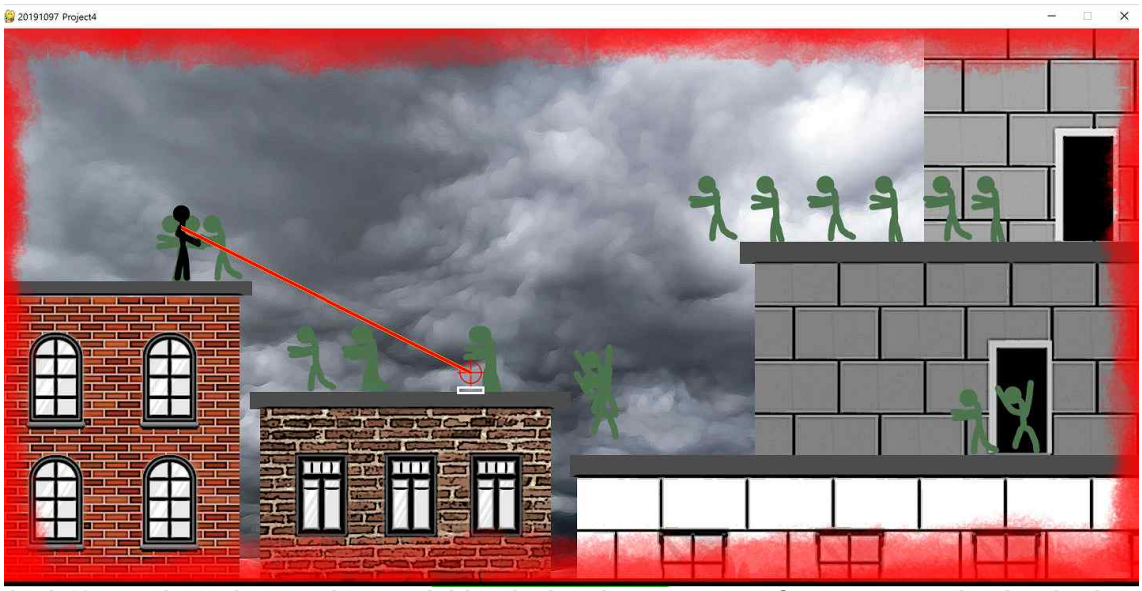
You can see bullets flying. Bullets are red tiny rectangles.



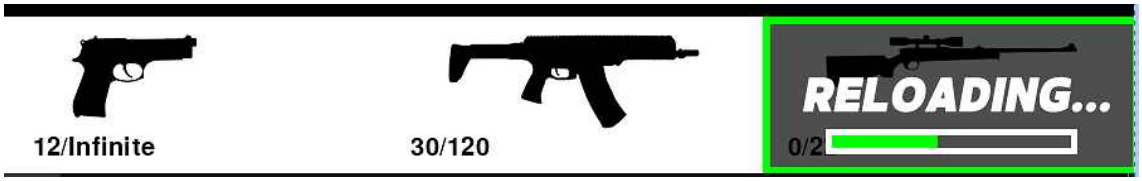
If Concentration is available, icon turns to red. If not, it's black. Below the icon is a bar that displays cool down. If cool down is not completed, red rectangle fills the empty bar. If the cool down is over, bar turns to green.



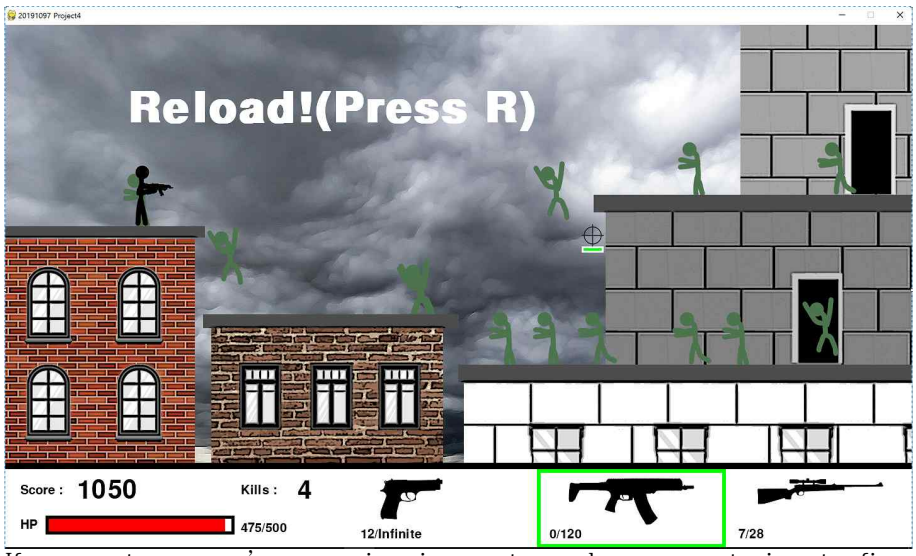
If Concentration is successfully used, red line appears between player and mouse. And more impressive gun fire sound is played. So you can check the skill use easily.



And if you hurt by zombie, red bloody border appears. So you can check whether you hurt.



When you reload a weapon, that weapon's reloading status is displayed. Also displays progress bar.



If current weapon's magazine is empty and you are trying to fire, that message is printed on the screen. Also specific sound is played. So you can easily know timing of reloading.

2.Code

(1)Before class define(Line 1 ~ 85)

Imported modules(pygame,numpy,os).

Loaded asset files.

Defined color.

Defined and set window.

Initialized pygame.

Defined fonts.

Play BGM.

(2)Class define(I'll only explain line that explain is needed.)

1)class Player – player object

```
86  class Player(pygame.sprite.Sprite):
87      def __init__(self):
88          super().__init__()
89          self.image=img_stand
90          self.rect=self.image.get_rect()
91          self.rect.x,self.rect.y=200,100
92          self.speedx=0
93          self.speedy=0
94          self.hp=500
95          self.last_hit=-600
96          self.weapon='pistol'
97          self.direction='right'
98          self.last_anim=0 #last animated tick
99          self.last_anim_frame='run1' #last ani frame
100         self.score=0
101         self.kills=0
102         self.run_sound=-800#last run_sound played tick
```

96 : current weapon

97 : moving direction(left or right)

98 : Tick of last animation frame showed. Used for running animation.

99 : Last animation's frame. Only run1 or run2. Used for running animation.

100~101 : Player's score and kills.

102 : Tick of last running sound was played. Used for running animation.

```
201     def calc_grav(self): #calculat
202         if self.speedy == 0:
203             self.speedy = 1
204             airborne=False
205         else:
206             self.speedy += .7
207             airborne=True
208             self.image=img_jump
209         return airborne
210
```

Check if object is airborne(202). If not, gravity affects object(205~207). And returns airborne boolean.

```
104     def update(self,):
105         airborne=self.calc_grav()#calculate gravity and return Tru
106         keystate = pygame.key.get_pressed()
107         run=False
108 
109         #move left/right
110         if keystate[pygame.K_a]:
111             self.direction='left'
112             self.speedx = -10
113             run=True
114         if keystate[pygame.K_d]:
115             self.direction='right'
116             self.speedx = 10
117             run=True
118         if run: #running animation
119             if airborne==False:
120                 if self.last_anim_frame=='run2':
121                     if self.last_anim+60<pygame.time.get_ticks():
122                         self.last_anim=pygame.time.get_ticks()
123                         self.last_anim_frame='run1'
124                     self.image=img_run1
125                 elif self.last_anim_frame=='run1':
126                     if self.last_anim+60<pygame.time.get_ticks():
127                         self.last_anim=pygame.time.get_ticks()
128                         self.last_anim_frame='run2'
129                     self.image=img_run2
130                 if self.run_sound+800<pygame.time.get_ticks():
131                     snd_footstep.play()
132                     self.run_sound=pygame.time.get_ticks()
```

105 : calculate gravity effect and get airborne status.

107 : object's running status.

110~117 : By key input, change direction, dx, run.

118~132 : While running, show running animation and play sound.

```
137         #actually move and adjust position not to pass platform or WINDOW
138         self.rect.x += self.speedx
139         block_hit_list = pygame.sprite.spritecollide(self, platforms, False)
140         for block in block_hit_list:
141             if self.speedx > 0:
142                 self.rect.right = block.rect.left
143             elif self.speedx < 0:
144                 self.rect.left = block.rect.right
145         if self.rect.left < 0:
146             self.rect.left=0
147         elif self.rect.right>WINDOW_WIDTH:
148             self.rect.right = WINDOW_WIDTH
149 
150         #actually move and adjust position not to pass platform or WINDOW
151         self.rect.y += self.speedy
152         platform_hit_list = pygame.sprite.spritecollide(self, platforms, False)
153         for block in platform_hit_list:
154             if self.speedy > 0:
155                 self.rect.bottom = block.rect.top
156             elif self.speedy < 0:
157                 self.rect.top = block.rect.bottom
158             self.speedy = 0
```

138~148 : Move object left or right. Make object cannot pass through platform object and window border.

151~158 : Move object up or down. Make object cannot pass through platform object and window border. And stand on platform object.

```
160         #change weapon
161         if keystate[pygame.K_1] or keystate[pygame.K_KP_1]:
162             self.weapon = 'pistol'
163             weapon.reloading=False
164         if keystate[pygame.K_2] or keystate[pygame.K_KP_2]:
165             self.weapon = 'AR'
166             weapon.reloading=False
167         if keystate[pygame.K_3] or keystate[pygame.K_KP_3]:
168             self.weapon = 'SR'
169             weapon.reloading=False
170 
171         #stand
172         if not (keystate[pygame.K_w] or keystate[pygame.K_d] or keystate[pygame.K_a]):
173             self.speedx = 0
174             self.image=img_stand
175 
176         #flip image by direction
177         if self.direction=='left':
178             self.image=pygame.transform.flip(self.image,True,False)
179 
```

161~168 : Change weapon by key input and cancel reloading.

172~174 : If any moving key is not inputed, show stand image of player.

177~178 : If direction is left, flip the image.(All player image are right oriented.)

```
180         ####Interactions####
181         #hit by zombie
182         zombie_hit_player=pygame.sprite.spritecollide(self,zombies,False)
183         if len(zombie_hit_player)>0:
184             self.hit()
185         #get medikit
186         player_get_med=pygame.sprite.spritecollide(self,medikits,False)
187         for med in player_get_med:
188             med.kill()
189             self.heal()
190         #get ARammo
191         player_get_ARammo=pygame.sprite.spritecollide(self,ARammos,False)
192         for ARa in player_get_ARammo:
193             ARa.kill()
194             weapon.ARammo+=30
195         #get SRammo
196         player_get_SRammo=pygame.sprite.spritecollide(self,SRammos,False)
197         for SRa in player_get_SRammo:
198             SRa.kill()
199             weapon.SRammo+=7
```

Interaction with zombie, medikit, SRammo, ARammo. Player get hurt, healed, SR/AR ammo by colliding. Collided medikit and ammo disappears.

```
211     def jump(self):#jump only it's ok to jump
212         self.rect.y += 2
213         platform_hit_list = pygame.sprite.spritecollide(self, platforms, False)
214         self.rect.y -= 2
215         if len(platform_hit_list) > 0 or self.rect.bottom >= WINDOW_HEIGHT:
216             if self.rect.bottom<=WINDOW_HEIGHT - 2:
217                 self.speedy = -16
218 
219     def hit(self):#hurt by zombie
220         if self.last_hit + 500 < pygame.time.get_ticks():#0.5 sec hit delay
221             self.hp-=25
222             self.last_hit = pygame.time.get_ticks()
223             snd_ouch.play()
224             snd_zombite.play()
225         if self.hp<=0:
226             self.hp=0
227             self.kill()
228             weapon.kill()
229             snd_scream.play()
```

211~217 : Check if it's ok to jump and then jump.

219~229 : Get 25 damage if 0.5 sec passed from last damage. If HP<=0, kill player.

2)class Weapon - player's hand.

```
236 class Weapon(pygame.sprite.Sprite):
237     def __init__(self, player):
238         super().__init__()
239         self.weapon='pistol'
240         self.image=img_pistol_arm
241         self.rect=self.image.get_rect()
242         self.rect.centerx=player.rect.x+player.image.get_width()/2
243         self.rect.centery=player.rect.y+player.image.get_height()/4
244         self.degree=0
245 
246         #magazine's bullet of each weapons
247         self.pistolmagazine=12
248         self.ARmagazine=30
249         self.SRmagazine=7
250 
251         #left ammo of each weapons
252         self.ARammo=120
253         self.SRammo=28
254 
255         ##reload
256         self.reloading=False #True:now reloading
257         self.reload_start=0 #reload started tick
258 
259         #last tick of each weapon's fire
260         self.last_pistol=0
261         self.last_AR=0
262         self.last_SR=0
263 
264         ##concentration skill
265         self.concentration=False #True:now available
266         self.last_conc=-2000 #tick of skill last used
```

239~241 : Init weapon and image, rect. Player's first weapon is pistol, so it's OK
242~243 : Set position of weapon to player character's shoulder position.
244 : Initialize degree to 0. Degree of aim line. I'll explain soon.
247~249 : The number of bullets of each weapon's magazine. Initialized as full magazine.
252~253 : The number of bullets that are not in magazine and can use later. Reload magazine by these bullets.
256 : True means now reloading, False is not.
257 : Tick that reload was started.
265 : True means 2 condition for Concentration satisfied.
266 : Tick of last concentration use.

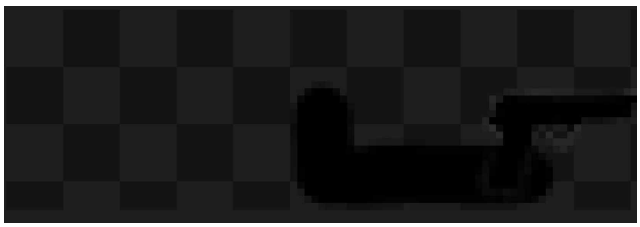
```
268     def update(self,):
269         #updating weapon image
270         if player.weapon=='pistol':
271             self.weapon='pistol'
272             self.image=img_pistol_arm
273         elif player.weapon=='AR':
274             self.weapon='AR'
275             self.image=img_AR_arm
276         else:
277             self.weapon='SR'
278             self.image=img_SR_arm
```

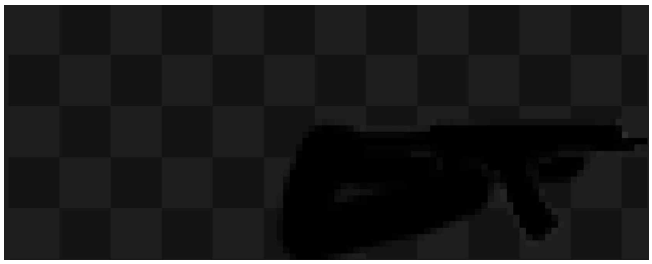
Update weapon and its image correspond to player's.

```
280         #get mouse position and degree between horizon line(1) from hand and aim line
281         m_pos = pygame.mouse.get_pos()
282         m_x = m_pos[0]
283         m_y = m_pos[1]
284           #line2# pygame.draw.line(screen,BLACK,[weapon.rect.centerx,weapon.rect.cent
285           #line1# pygame.draw.line(screen,BLACK,[weapon.rect.centerx,weapon.rect.cent
286         degree=np.arctan(np.abs(m_y-self.rect.centery)/np.abs(m_x-self.rect.centerx))
287         degree=np.rad2deg(degree)
288 
289         #flip and roatate according to player's sight(mouse position)
290         if m_x>=self.rect.centerx:
291             if m_y>=self.rect.centery:
292                 degree*=-1
293             self.image=pygame.transform.rotate(self.image,degree)
294         else:
295             self.image=pygame.transform.flip(self.image,False,True)
296             if m_y>=self.rect.centery:
297                 degree+=180
298             else:
299                 degree=180-degree
300             self.image=pygame.transform.rotate(self.image,degree)
301         self.degree=degree
302 
303         # Move arm image position to player's xy
304         self.rect=self.image.get_rect()
305         self.rect.center=[player.rect.centerx,player.rect.centery-20]
```

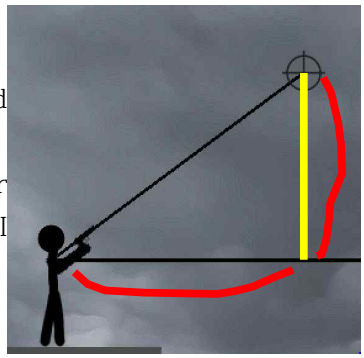
Rotating weapon image correspond to mouse position. I used transform.rotate.

Since this method rotates image by it's center, I used these images.





These images' center is shoulder. And these image's center go to player's shoulder position(304~305). So, position adjusting due to rotation's rect change is not needed.



281~287 : Get mouse position and get degree of right two black line.
I calculated tan first and used arctan to get degree(286). And converted radian to degree(287).
290~300 : Rotate and flip image by mouse position and player position. Those degree adjusting succeed by trial and failure. So, I don't know why that's correct.

```
307     def shoot(self): #generate bullet
308         if not self.reloading: #cannot shoot while reloading
309             now=pygame.time.get_ticks()
310             if self.weapon=='pistol':
311                 if self.last_pistol+400<now:#0.4sec delay
312                     if self.pistolmagazine>0:
313                         if self.concentration==False: #normal
314                             a=Bullet(self)
315                             bullets.add(a)
316                             self.last_pistol=now
317                             self.pistolmagazine -= 1
318                             snd_pistolshot.play()
319                         else: #skill
320                             self.last_pistol=now
321                             self.pistolmagazine -= 1
322                             self.last_conc=now
323                             aim.concentrate()
324                             snd_pistolcon.play()
325                     else: #need reload
326                         screen.blit(text_reload,[200,100])
327                         snd_nobullet.play()
328 
329             elif self.weapon=='AR':
330                 if self.last_AR+100<now: #0.1sec delay
```

shoot method generates bullet. Cannot shoot while reloading(308).

310 : check weapon
311 : check if shot delay has passed. -> if not, nothing happens.
312 : check if magazine is not empty. -> if not, cannot shoot(325~327)
313 : if it's not concentration shot. -> 314~318 : minus 1 from magazine. 1 Bullet.
391~324 : if it's concentration shot, minus 1 from magazine, 0 Bullet. Activate skill, update last skill used tick.
329~365 : Repeating above mechanism for AR and SR. I'll pass explain.

```
367     def reload(self): #reload
368         now=pygame.time.get_ticks()
369         if self.weapon=='pistol':
370             if self.pistolmagazine<12:
371                 if self.reload_start + 1000 < now:
372                     self.pistolmagazine=12
373                     self.reloading=False
374             else:
375                 self.reloading=False
376         elif self.weapon=='AR':
377             if self.ARmagazine<30:
378                 if self.ARammo>0:
379                     if self.reload_start + 1000 < now:
380                         needed=30-self.ARmagazine
381                         if needed>=self.ARammo:
382                             self.ARmagazine+=self.ARammo
383                             self.ARammo=0
384                         else:
385                             self.ARmagazine=30
386                             self.ARammo-=needed
387                         self.reloading=False
388                 else: #no ammo left
389                     self.reloading=False
390             else: #magazine full
391                 self.reloading=False
```

369~375 : Reload pistol. Look mechanism of AR(376~391).
376 : Check current weapon
377 : Check if magazine is not full. If full, can't reload(390~391).
378 : Check if any ammo left. If not, can't reload(388~389).
379 : Check if reloading time is done.
381~383 : Reload that can't fill magazine full because of ammo shortage.
384~386 : Normal reload.
387 : Reload completed.
392~407 : Code for reloading SR. Same as mechanism of AR. Skip.

3)class Bullet - bullet object

412~413 : Get weapon and it's degree.
416~418 : Set bullet rect.
421~422 : Set bullet position to weapon's.
425~427 : Means N zombies can get damage by one bullet. Only SR is 3 and the other is 1.

```
409 class Bullet(pygame.sprite.Sprite):
410     def __init__(self, weapon):
411         super().__init__()
412         self.weapon=weapon.weapon#get weapon
413         self.degree=weapon.degree#get degree
414 
415         #bullet:tiny red square
416         self.image = pygame.Surface([6, 6])
417         self.image.fill(RED)
418         self.rect = self.image.get_rect()
419 
420         #get weapon's position
421         self.rect.centerx=weapon.rect.centerx
422         self.rect.centery=weapon.rect.centery
423         self.dx=0
424         self.dy=0
425         self.penetrate=1 #give damage to N zombies
426         if self.weapon=='SR':
427             self.penetrate=3
428 
```

```
429     def update(self):
430         #bullet move
431         self.get_dxy()
432         self.rect.centerx+=self.dx
433         self.rect.centery+=self.dy
434 
435         #bullet and zombie collide
436         bullet_hit_zombie=pygame.sprite.spritecollide(self,zombies,False)
437         for hit_zombie in bullet_hit_zombie:
438             hit_zombie.hit(self)
439             self.penetrate-=1
440             if self.penetrate==0:
441                 self.kill()
442                 break #no multi hit
443 
444         bullet_hit_platform=pygame.sprite.spritecollide(self,platforms,False)
445         if len(bullet_hit_platform)>0:
446             self.kill()
447 
448         #delete when go out of window
449         if self.rect.centerx>WINDOW_WIDTH or self.rect.centerx<0 or self.rect.centery>WINDOW_HEIGHT or self.rect.centery<0:
450             self.kill()
451 
452     def get_dxy(self):
453         #get dxy by cos and sin. For any direction, same speed
454         self.dx=np.cos(np.deg2rad(-self.degree))
455         self.dy=np.sin(np.deg2rad(-self.degree))
456         self.dx*=40
457         self.dy*=40
```

452~457 : Get dx, xy of bullet. Used sin and cos to go have same velocity at any degree. I multiplied 40 to speed up bullet.

431~433 : Move bullet

436~442 : If bullet and zombie collided, zombie get damage. Also bullet disappears if its penetration is over.

444~446 : If bullet and platform collided, bullet disappears.

449~450 : If bullet get out of the window, bullet disappears.

4)class Zombie - zombie object.

```
459 class Zombie(pygame.sprite.Sprite):
460     def __init__(self, image):
461         super().__init__()
462         self.image=image
463         self.rect=self.image.get_rect()
464         self.hp=100
465         self.speedx=0
466         self.speedy=0
467 
468         #as same as player's
469         self.last_anim=0
470         self.last_anim_frame='run1'
471 
472         #set random spawn point
473         spawn=np.random.randint(0,2)
474         if spawn==0:
475             self.rect.x=1360
476             self.rect.y=150
477         elif spawn==1:
478             self.rect.x=1280
479             self.rect.y=425
480 
481         self.direction='right'
482         self.knockback=False
483         self.last_hit=0
```

469~470, 481 : Same as player's

473~479 : Determine spawn point. One door(in game map) is selected randomly.

482 : True==zombie is now knock-backed.

483 : Tick of last hit by bullet.

```
485     def update(self,player):
486         airborne=self.calc_grav()
487         if self.knockback:
488             if self.last_hit + 250 > pygame.time.get_ticks():
489                 self.image=img_zombie_knockback
490             else:
491                 self.knockback=False
492         else:
493             if airborne==False:
494                 if self.last_anim_frame=='run2':
495                     if self.last_anim+90<pygame.time.get_ticks():
496                         self.last_anim=pygame.time.get_ticks()
497                         self.last_anim_frame='run1'
498                     self.image=img_zombie_walk1
499                 elif self.last_anim_frame=='run1':
500                     if self.last_anim+90<pygame.time.get_ticks():
501                         self.last_anim=pygame.time.get_ticks()
502                         self.last_anim_frame='run2'
503                     self.image=img_zombie_walk2
```

553~562 : Define calc_grav method. Same as player's

564~771 : Define jump method. Same as player's

487~491 : Show knock-back image for 0.25sec if zombie got hit.

492~503 : Show zombie walking image. Same as player's.

```
505         if player.rect.centerx >= self.rect.centerx:
506                     self.speedx=3
507                     self.direction='right'
508         else:
509                     self.speedx=-3
510                     self.direction='left'
511 
512         self.rect.x+=self.speedx
513 
514         block_hit_list = pygame.sprite.spritecollide(self, platforms, False)
515         for block in block_hit_list:
516             if not self.knockback:
517                 if self.speedx > 0:
518                     self.rect.right = block.rect.left
519                 elif self.speedx < 0:
520                     self.rect.left = block.rect.right
521                 self.jump()
522             else:
523                 if self.speedx > 0:
524                     self.rect.left = block.rect.right
525                 elif self.speedx < 0:
526                     self.rect.right = block.rect.left
527 
528         growl=np.random.randint(0,5000)
529         if growl==1:
530             snd_zom1.play()
531         elif growl==2:
532             snd_zom2.play()
533         elif growl==3:
534             snd_zom3.play()
```

505~510 : change zombie's direction and speedx. So, zombie comes to player.

514~526 : Make zombie cannot pass through platform. If there's platform on the way, zombie jumps(521).

528~534 : Rarely play zombie sound. More zombie, more sound play.

```
538         self.rect.y += self.speedy
539
540         platform_hit_list = pygame.sprite.spritecollide(self, platforms, False)
541         for block in platform_hit_list:
542             # Reset our position based on the top/bottom of the object.
543             if self.speedy > 0:
544                 self.rect.bottom = block.rect.top
545             elif self.speedy < 0:
546                 self.rect.top = block.rect.bottom
547             # Stop our vertical movement
548             self.speedy = 0
549
550         if self.direction=='right':
551             self.image=pygame.transform.flip(self.image,True,False)
552
553         if gameover:
554             self.kill()
```

538~548 : Step on platform.

550~551 : Flip image regard as direction

553~554 : Erase all zombie when game ended.

```
574     def hit(self,bullet):
575         if bullet.weapon=='pistol':
576             self.hp-=40
577             player.score+=40
578         elif bullet.weapon=='AR':
579             self.hp-=30
580             player.score+=30
581         else:
582             self.hp-=100
583             player.score+=100
584
585         self.last_hit=pygame.time.get_ticks()
586         self.knockback=True
587
588         if self.direction=='right':
589             self.rect.x -= 40
590         else:
591             self.rect.x += 40
```

575~583 : Get damage by weapon and plus score.

585~591 : Give knock-back

```
596             if self.hp<=0:
597                 player.score+=self.hp
598                 z=Zombiedie(self)
599                 all_sprites.add(z)
600                 player.score+=100
601                 player.kills+=1
602 
603                 drop=np.random.randint(0,40)
604                 if drop==0:
605                     ss=SRammo(self)
606                     SRammos.add(ss)
607                     all_sprites.add(ss)
608                 elif drop==1 or drop==2:
609                     mm=ARammo(self)
610                     ARammos.add(mm)
611                     all_sprites.add(mm)
612                 elif drop==3:
613                     m=Medikit(self)
614                     medikits.add(m)
615                     all_sprites.add(m)
616                 self.kill()
617                 snd_zomdie.play()
```

Code for when zombie dies.

597 : plus damage score.

598~599 : generate animation showing object.

600~601 : plus kill score and kills.

603~615 : randomly drop item.

5)class Zombiedie - for zombie dying animation.

```
619 class Zombiedie(pygame.sprite.Sprite):#zombie dying animation
620     def __init__(self,zombie):
621         super().__init__()
622         self.image=img_zombie_die1
623         self.rect=self.image.get_rect()
624         self.direction=zombie.direction
625         if self.direction=='right':
626             self.rect.x=zombie.rect.left
627             self.image=pygame.transform.flip(self.image,True,False)
628         else:
629             self.rect.x=zombie.rect.left
630 
631         self.rect.bottom=zombie.rect.bottom
632         self.last_anim=pygame.time.get_ticks()
633         self.frame=1
634         self.location=[self.rect.centerx, self.rect.bottom]
635         self.dy=0
```

Get zombies' position and direction. And flip image according to direction. Other things are same as above all animating codes.

```
637     def update(self):
638         self.dy+=.7
639         self.rect.bottom += self.dy
640         platform_hit_list = pygame.sprite.spritecollide(self, platforms, False)
641         for block in platform_hit_list:
642             self.rect.bottom = block.rect.top
643             self.dy = 0
644 
645         if self.frame==1:
646             if self.last_anim+300<pygame.time.get_ticks():
647                 self.image=img_zombie_die2
648                 if self.direction=='right':
649                     self.image=pygame.transform.flip(self.image,True,False)
650                 self.rect=self.image.get_rect()
651                 self.rect.bottom=self.location[1]
652                 self.rect.centerx=self.location[0]
653                 self.frame=2
654                 self.last_anim=pygame.time.get_ticks()
655 
656         elif self.frame==2:
657             if self.last_anim+300<pygame.time.get_ticks():
658                 self.kill()
```

638~643 : apply gravity, step on platform.

645~658 : show dying animation and kill().

6)class Platform - object that can step on

```
660 class Platform(pygame.sprite.Sprite):
661     def __init__(self, x,y,img):
662         super().__init__()
663         self.image = img
664         self.rect = self.image.get_rect()
665         self.rect.x = x
666         self.rect.y = y
```

Skip explain.

7)class SRammo, ARammo, Medikit - item objects.

```
668 class SRammo(pygame.sprite.Sprite):
669     def __init__(self, zombie):
670         super().__init__()
671         self.image=img_SRammo
672         self.rect=self.image.get_rect()
673         self.rect.center=zombie.rect.center
674         self.dy=-4
675 
676     def update(self):
677         self.dy+=.7
678         self.rect.bottom += self.dy
679 
680         platform_hit_list = pygame.sprite.spritecollide(self, platforms, False)
681         for block in platform_hit_list:
682             self.rect.bottom = block.rect.top
683             self.dy = 0
684 
685         if gameover:
686             self.kill()
```

677~683 : apply gravity, step on platform.

ARammo(688~706), Medikit(708~726) are copy&paste of SRammo. Skip.

8)class Aim - mouse

```
725 class Aim(pygame.sprite.Sprite):
726     def __init__(self):
727         super().__init__()
728         self.image=img_black_aim
729         self.rect=self.image.get_rect()
730 
731     def update(self):
732         self.rect.center=pygame.mouse.get_pos()
733         pygame.mouse.set_visible(False)
734         zombie_aimed = pygame.sprite.spritecollide(self, zombies, False)
735         if len(zombie_aimed)>0:
736             if weapon.last_conc+1000<pygame.time.get_ticks():
737                 self.image=img_red_aim
738                 weapon.concentration=True
739             else:
740                 self.image=img_black_aim
741                 weapon.concentration=False
742         else:
743             weapon.concentration=False
744             self.image=img_black_aim
```

732 : update aim position to mouse position.

734~744 : if mouse is over any zombie and concentration's cool down done, icon turns to red and concentration is available.

```
746     def concentrate(self):
747         zombie_aimed = pygame.sprite.spritecollide(self, zombies, False)
748         for zomb in zombie_aimed:
749             pygame.draw.line(screen,ORANGE,weapon.rect.center,pygame.mouse.get_pos(),6)
750             pygame.draw.line(screen,RED,weapon.rect.center,pygame.mouse.get_pos(),4)
751             zomb.kill()
752             player.kills+=1
753             zd=Zombiedie(zomb)
754             all_sprites.add(zd)
755             player.score+=150
756             player.score+=zomb.hp
757             weapon.concentration=False
758             weapon.last_conc=pygame.time.get_ticks()
759             break
```

Code for skill 'Concentration'.

Draw line between mouse and player(749,750), kill zombie(751~753) and get score(752,755,756), init weapon class' attribute related with concentration(757,758).

(3)Global variables out of loop.

```
761 zombies=pygame.sprite.Group()
762 platforms=pygame.sprite.Group()
763 bullets=pygame.sprite.Group()
764 medikits=pygame.sprite.Group()
765 ARammos=pygame.sprite.Group()
766 SRammos=pygame.sprite.Group()
767 all_sprites = pygame.sprite.Group()
768
769 #add platforms
770 p1=Platform(0,320,img_building1)
771 p2=Platform(320,460,img_building2)
772 p3=Platform(725,540,img_building3)
773 p4=Platform(940,270,img_platform1)
774 for i in [p1,p2,p3,p4]:
775     platforms.add(i)
776     all_sprites.add(i)
777
778 player=Player()
779 weapon=Weapon(player)
780 aim=Aim()
781 all_sprites.add(player)
782 all_sprites.add(weapon)
783 all_sprites.add(aim)
784
785 #text render
786 text_title=scorefont.render("Concentrate to Survive",True,BLACK)
787 text_score=UIfont.render("Score :",True,BLACK)
788 text_kills=UIfont.render("Kills :",True,BLACK)
789 text_pressstart=scorefont.render("Press Space to start",True,BLACK)
790
791 is_shooting=False #mouse click
792 gameover=True #one game
793 lastscore=0 #for title last score
794 lastSpawn=0 #for zombie spawn
795 done=False #for pygame loop
```

761~767 : define sprite groups.

770~783 : generate platform, player, weapon, aim objects and add in to group.

786 : render text

791 : True means mouse is clicked.

792 : True means game is over.

793 : Score of previous game.

794 : tick of last zombie spawned.

(4)Inside the loop – actual game running

```
797 while not done:
798     #draw basic graphics(backgrounds)
799     screen.fill(WHITE)
800     screen.blit(img_sky,[0,0])
801     screen.blit(img_building4,[940,270])
802     screen.blit(img_building5,[1173,0])
803     pygame.draw.line(screen,BLACK,[0,700],[1450,700],10)
804     text_lastscore=scorefont.render(str(lastscore),True,BLACK)
805 
806     if gameover:#gameover display
807         for event in pygame.event.get():
808             if event.type == pygame.QUIT:
809                 done = True
810             elif event.type == pygame.KEYDOWN:
811                 if event.key==pygame.K_SPACE:
812                     gameover=False #game start
813         screen.blit(text_title,[500,300])
814         screen.blit(text_score,[500,400])
815         screen.blit(text_lastscore,[590,390])
816         screen.blit(text_pressstart,[520,750])
817         all_sprites.update()
818         all_sprites.draw(screen)
819     else: #game started
```

799~804 : blit images. render last score.

806~818 : Title display. If space pressed, game start(810~812).

```
819     else: #game started
820         now=pygame.time.get_ticks()
821         for event in pygame.event.get():
822             if event.type == pygame.QUIT:
823                 done = True
824             elif event.type == pygame.MOUSEBUTTONDOWN:
825                 is_shooting=True
826             elif event.type == pygame.MOUSEBUTTONUP:
827                 is_shooting=False
828             elif event.type == pygame.KEYDOWN and weapon.reloading==False:
829                 if event.key==pygame.K_r:
830                     weapon.reload_start=pygame.time.get_ticks()
831                     if player.weapon=='pistol':
832                         snd_pistolreload.play()
833                     elif player.weapon=='AR' and weapon.ARammo>0:
834                         snd_ARreload.play()
835                     elif player.weapon=='SR' and weapon.SRammo>0:
836                         snd_SRreload.play()
837                     weapon.reloading=True
838 
839         if weapon.reloading==True:
840             weapon.reload()
```

824~827 : if clicked, is_shooting=True, if not, is_shooting=False

828~840 : reload weapon.

```
842         all_sprites.update()
843
844         zombies.update(player)
845         zombies.draw(screen)
846         bullets.update()
847         bullets.draw(screen)
848         all_sprites.draw(screen)
849
850         if is_shooting:
851             weapon.shoot()
852
853         if lastSpawn+500<now: #spawn zombie every 0.5sec
854             lastSpawn=pygame.time.get_ticks()
855             a=Zombie(img_zombie_walk1)
856             zombies.add(a)
857
858         pygame.draw.line(screen,BLACK,[0,700],[1450,700],10)
859
860         if player.last_hit+300>pygame.time.get_ticks(): #blo
861             screen.blit(img_blood_border,[0,0])
862
```

853~856 : spawn a zombie every 0.5 second.

858 : Line between game board and below UI.

860~861 : Show blood splatter border if player got hit.

```
863         #Indicate HP info
864         text_hp=UIfont.render("HP",True,BLACK)
865         text_playerHP=UIfont.render(str(player.hp)+"/500",True,BLACK)
866         pygame.draw.rect(screen,RED,[70,780,300*player.hp/500,30],0)
867         pygame.draw.rect(screen,BLACK,[70,780,300,30],5)
868         screen.blit(text_hp,[30,785])
869         screen.blit(text_playerHP,[380,790])
870
871         #Indicate score info
872         text_currnetscore=scorefont.render(str(player.score),True,BLACK)
873         screen.blit(text_score,[30,730])
874         screen.blit(text_currnetscore,[120,720])
875
876         #Indicate kills info
877         text_currnetkills=scorefont.render(str(player.kills),True,BLACK)
878         screen.blit(text_kills,[380,730])
879         screen.blit(text_currnetkills,[470,720])
```

Indicates information of HP, score, kills.

```
881             #Indicate weapon info
882                 #weapon images
883             screen.blit(img_pistol,[600,720])
884             screen.blit(img_AR,[900,720])
885             screen.blit(img_SR,[1200,720])
886                 #weapon ammo
887             text_pistol=UIfont.render(str(weapon.pistolmagazine)+'/Infinite',True,BLACK)
888             screen.blit(text_pistol,[570,800])
889             text_AR=UIfont.render(str(weapon.ARmagazine)+'/'+str(weapon.ARammo),True,BLACK)
890             screen.blit(text_AR,[870,800])
891             text_SR=UIfont.render(str(weapon.SRmagazine)+'/'+str(weapon.SRammo),True,BLACK)
892             screen.blit(text_SR,[1170,800])
893                 #show selected weapon and reloading status
894             now=pygame.time.get_ticks()
895             if player.weapon=='pistol':
896                 if weapon.reloading:
897                     screen.blit(img_reloading,[550,706])
898                     pygame.draw.rect(screen,GREEN,[600,795,200*(now-weapon.reload_start)/1000,20])
899                     pygame.draw.rect(screen,WHITE,[600,795,200,20],5)
900                 pygame.draw.rect(screen,GREEN,[550,706,300,125],6)
901             elif player.weapon=='AR':
902                 if weapon.reloading:
903                     screen.blit(img_reloading,[850,706])
904                     pygame.draw.rect(screen,GREEN,[900,795,200*(now-weapon.reload_start)/1000,20])
905                     pygame.draw.rect(screen,WHITE,[900,795,200,20],5)
906                 pygame.draw.rect(screen,GREEN,[850,706,300,125],6)
907             else:
908                 if weapon.reloading:
909                     screen.blit(img_reloading,[1150,706])
910                     pygame.draw.rect(screen,GREEN,[1200,795,200*(now-weapon.reload_start)/1500,20])
911                     pygame.draw.rect(screen,WHITE,[1200,795,200,20],5)
912                 pygame.draw.rect(screen,GREEN,[1150,706,300,125],6)
```

Indicates weapon information.

887~982 : weapon's ammo status.

894~912 : show selected weapon and reloading image, reloading progress bar.

```
914             #Indicate concentrate cooldown
915             if weapon.last_conc+990>now:
916                 pygame.draw.rect(screen,RED,[aim.rect.x,aim.rect.bottom,35*(now-weapon.last_conc)/990,7],0)
917             else:
918                 pygame.draw.rect(screen,GREEN,[aim.rect.x,aim.rect.bottom,35,7],0)
919             pygame.draw.rect(screen,WHITE,[aim.rect.x,aim.rect.bottom,35,10],3)
920
921             if player.hp<=0:#init sprites and go to gameover display
922                 lastscore=player.score#save last score. will be displayed on gameover display
923                 zombies=pygame.sprite.Group()
924                 bullets=pygame.sprite.Group()
925                 SRammos=pygame.sprite.Group()
926                 ARammos=pygame.sprite.Group()
927                 medikits=pygame.sprite.Group()
928                 player=Player()
929                 weapon=Weapon(player)
930                 all_sprites.add(player)
931                 all_sprites.add(weapon)
932                 gameover=True
933                 zombies.update(player)
934
935         pygame.display.flip()
936
937         clock.tick(60)
938
939     pygame.quit()
```

915~919 : Indicates concentration cool down progress bar.

921~933 : When player dead, initialize sprites and gameover=True.