В программе присутствуют функции:

1. draw\_text – позволяет быстро и удобно вывести текст на нужном холсте

def draw\_text(surf, text, size, x, y):  
 font\_name = pygame.font.match\_font('arial')  
 font = pygame.font.Font(font\_name, size)  
 text\_surface = font.render(text, True, BLACK)  
 text\_rect = text\_surface.get\_rect()  
 text\_rect.midtop = (x, y)  
 surf.blit(text\_surface, text\_rect)

2)shop - необходим для торговцев для показа магазина

def shop(a):  
 fon = pygame.transform.scale(pygame.image.load('img/shop.png').convert(), (400, 200))  
 rect = fon.get\_rect()  
 screen.blit(fon, (700, 400))  
 rect.topleft = (700, 400)  
 shops = pygame.sprite.Group()  
 if a == 1:  
 heart = Heart(850, 480)  
 shops.add(heart)  
 draw\_text(screen, 'вылечиться до 100:', 18, 880, 450)  
 elif a == 0:  
 weapon = Drop(850, 500, 'weapon', 'gold weapon')  
 armor = Drop(950, 560, 'armor', 'gold')  
 draw\_text(screen, 'gold weapon:', 18, 915, 500)  
 draw\_text(screen, '200', 20, 900, 520)  
 draw\_text(screen, 'gold armor:', 18, 1020, 560)  
 draw\_text(screen, '300', 20, 1010, 580)  
 shops.add(weapon)  
 shops.add(armor)

функция создаёт новый экран и рисует табличку на которой уже расположены товары, если игрок кликнет за пределами окна, то торговля окончится и появится «эффект» от торговли в виде лечения или покупки предметов, которые выпадут рядом.

3)fight – схожа с предыдущей функцией, но разница в возможностях и условиях битвы

def fight(monster, monsterhp, monsterdamage, starthp, damage, chance, dodge):  
 fon = pygame.transform.scale(pygame.image.load('img/shop.png').convert(), (400, 200))  
 rect = fon.get\_rect()  
 icons = pygame.sprite.Group()  
 atack = Sword(910, 540)  
 shield = Shield(810, 540)  
 icons.add(atack)  
 icons.add(shield)  
 rect.topleft = (700, 400)  
 run = True

Игрок может выйти из боя только умерев или победив монстра, поэтому есть два варианта, либо лечиться +3 hp или же атаковать с определённой вероятностью

if atack.pressed(mx, my) or shield.pressed(mx, my):  
 if atack.pressed(mx, my):  
 if random.random() <= chance / 100:  
 monsterhp -= damage  
 if monsterhp <= 0:  
 global live  
 live = starthp  
 return True

урон игрок получит в зависимости от шанса доджа атаки, который повышается при надевании различных типов брони, также каждое оружие имеет шанс атаки при ударе, но его можно дополнительно увеличить надев амулет.

4) win\_screen, death\_screen, start\_screen – функции вызывающиеся при определённом результате (победе над боссом, смерти игрока во время боя или при запуске игры) позволяет делать отдельное меню со своими взаимодействиями без вреда на основную игру, ограничиваясь только фантазией как можно их украсить или сделать интерактив.

def death\_screen():  
 fon = pygame.transform.scale(pygame.image.load('img/fon.png').convert(), (WIDTH, HEIGHT))  
 screen.blit(fon, (0, 0))  
 button\_sprites = pygame.sprite.Group()  
 button1 = Button(900, 510, 'img/button2.png')  
 button\_sprites.add(button1)  
 running = True  
 while running:  
 events = pygame.event.get()  
 for event in events:  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 mx, my = pygame.mouse.get\_pos()  
 if button1.pressed(mx, my):  
 pygame.quit()  
 quit()  
 button\_sprites.draw(screen)  
 pygame.display.flip()  
 clock.tick(FPS)

При старте игры создаются две локации:

1. Улица (вне башни) на ней игрок появляется в самом начале и может подойти к трём торговцам, которые могут продать предметы или вылечить игрока

class Seller(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, sheet, columns, rows, x, y):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.frames = []  
 self.cut\_sheet(sheet, columns, rows)  
 self.cur\_frame = 0  
 self.image = pygame.transform.scale(self.frames[self.cur\_frame], (130, 170))  
 self.rect = self.rect.move(x, y)  
 self.image.set\_colorkey(BLACK)  
  
 def cut\_sheet(self, sheet, columns, rows):  
 self.rect = pygame.Rect(0, 0, sheet.get\_width() // columns,  
 sheet.get\_height() // rows)  
 for j in range(rows):  
 for i in range(columns):  
 frame\_location = (self.rect.w \* i, self.rect.h \* j)  
 self.frames.append(sheet.subsurface(pygame.Rect(  
 frame\_location, self.rect.size)))

также при клике на торговца он выводит подсказку, что он продаёт.

2)Башня в которую можно войти с улицы, каждый раз при запуске игры она спавниться случайным образом имея две боковые комнаты со случайной мебелью в лице ящиков и бочек, а также с шансом 33 % может появится сундук с определённым типом вещей.

if random.choice([False, True, False]):  
 chest = Chest(x + 170, y + 110)

class Chest(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 global chest\_imgs, ARMOR, WEAPON, AMULET  
 chest\_img = random.choice(chest\_imgs)  
 yachik\_img = pygame.transform.scale(chest\_img, (75, 40))  
 self.y = y  
 self.image = yachik\_img  
 if chest\_img == chest\_imgs[0]:  
 self.drop = random.choice(WEAPON)  
 self.type = 'Weapon'  
 elif chest\_img == chest\_imgs[1]:  
 self.drop = random.choice(ARMOR)  
 self.type = 'Armor'  
 else:  
 self.drop = random.choice(AMULET)  
 self.type = 'Amulet'  
 self.image.set\_colorkey(MEBEL)  
 self.rect = self.image.get\_rect()  
 self.rect.center = (x, y)

Также для усложнения геймплея на каждом этаже в центральной комнате присутствует монстр (всего 4 вида), если игрок находится рядом с ним то вызывается функция fight и начинается бой.

class Monsters(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, sheet, columns, rows, x, y, name):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.frames = []  
 self.cut\_sheet(sheet, columns, rows)  
 self.cur\_frame = 0  
 self.image = pygame.transform.scale(self.frames[self.cur\_frame], (400, 400))  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = x, y  
 self.image.set\_colorkey(BLACK)  
 if name == 'Sans':  
 self.name = 'Sans'  
 self.hp = 200  
 self.damage = 10  
 elif name == 'Гоблин Рикардо':  
 self.name = 'Гоблин Рикардо'  
 self.hp = 100  
 self.damage = 9  
 elif name == 'Одноглаз':  
 self.name = 'Одноглаз'  
 self.hp = 100  
 self.damage = 8  
 elif name == 'Грыб':  
 self.name = 'Грыб'  
 self.hp = 80  
 self.damage = 8

def update(self):  
 self.cur\_frame = (self.cur\_frame + 1) % len(self.frames)  
 self.image = pygame.transform.scale(self.frames[self.cur\_frame], (400, 400))  
 self.image.set\_colorkey(BLACK)  
 if self.rect.centerx - 30 <= player.rect.centerx <= self.rect.centerx + 30:  
 if self.rect.centery == 870:  
 global live, money  
 if fight(self.name, self.hp, self.damage,  
 live, player.damage, player.chance + player.dop\_chance, player.dodge):  
 money += random.randint(1, 15)  
 self.kill()  
 else:  
 death\_screen()

При победе над монстром у игрока убавляется хп, которое он потерял во время боя и выдаётся случайное количество монет.

Для победы в игре нужно пройти 24 этажа и на последнем победить главного босса, он мало отличается от монстров просто выделен в отдельный класс для удобства обработки и редактирования боя конкретно с финальным боссом, также ограничиваясь только фантазией и временем, при победе над ним вызывается функция win\_screen и игрока оповещают о победе.

class Boss(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 global boss\_imgs  
 self.index = 0  
 self.image = pygame.transform.scale(boss\_imgs[0], (400, 360))  
 self.image.set\_colorkey(BLACK)  
 self.rect = self.image.get\_rect()  
 self.rect.center = (x, y)  
 self.damage = 14  
 self.name = 'хранитель башни'  
 self.hp = 300  
  
 def update(self):  
 self.index += 1  
 if self.index >= len(boss\_imgs):  
 self.index = 0  
 self.image = pygame.transform.scale(boss\_imgs[self.index], (400,300))  
 self.image.set\_colorkey(BLACK)  
 if self.rect.centerx - 30 <= player.rect.centerx <= self.rect.centerx + 30:  
 if self.rect.centery == 835:  
 global live  
 if fight(self.name, self.hp, self.damage,  
 live, player.damage, player.chance + player.dop\_chance, player.dodge):  
 win\_screen()  
 else:  
 death\_screen()

Игру можно удобно дополнять новыми сундуками и соответственно новым дропов из них, дополняя только sql базу данных указывая картинку и характеристики предмета.

Игра была создана, основываясь на геймплее старых флеш игр 2000-ых