ИСХОДНЫЙ КОД

Игра «Слова из слов»

Киров, 2023 г.

import pygame  
import os  
import json  
  
# Загрузка данных из файлов  
with open('levels.txt') as levels\_file:  
 levels = json.load(levels\_file)  
  
with open('users.txt') as users\_file:  
 users = json.load(users\_file)  
  
with open('settings.txt') as settings\_file:  
 volume = json.load(settings\_file)  
  
WIDTH = 1280  
HEIGHT = 720  
FPS = 30  
  
# Задаем цвета  
WHITE = (255, 255, 255)  
GRAY = (128, 128, 128)  
BLACK = (0, 0, 0)  
  
# Создаем игру и окно  
pygame.init()  
pygame.mixer.init()  
screen = pygame.display.set\_mode((WIDTH, HEIGHT))  
pygame.display.set\_caption("Слова из слов")  
clock = pygame.time.Clock()  
  
# настройка папки ассетов  
game\_folder = os.path.dirname(\_\_file\_\_)  
img\_folder = os.path.join(game\_folder, 'foto')  
snd\_folder = os.path.join(game\_folder, 'zvuki')  
click\_snd = pygame.mixer.Sound(os.path.join(snd\_folder, 'click.wav'))  
victory\_snd = pygame.mixer.Sound(os.path.join(snd\_folder, 'victory.wav'))  
  
#загрузка изображений  
bg\_img = pygame.image.load(os.path.join(img\_folder, 'bg.png')).convert()  
accept\_img = pygame.image.load(os.path.join(img\_folder, 'accept.png')).convert()  
erase\_img = pygame.image.load(os.path.join(img\_folder, 'erase.png')).convert()  
logo\_img = pygame.image.load(os.path.join(img\_folder, 'logo.png')).convert()  
back\_img = pygame.image.load(os.path.join(img\_folder, 'back.png')).convert()  
forward\_img = pygame.image.load(os.path.join(img\_folder, 'forward.png')).convert()  
  
bg\_rect = bg\_img.get\_rect()  
  
# Шаблон пользователя  
user = {  
 'name': '',  
 'progress': [[], [], [], [], [], [], [], [], [], [], [], [], [], [], []]  
}  
  
level = 0  
word = ''  
words = []  
  
result = ''  
results = []  
  
forward = None  
  
# алфавиты для контроля ввода  
allowed\_symbols = {"1", "2", "3", "4", "5", "6", "7", "8", "9", "0", "-", "q", "w", "e", "r", "t", "y", "u", "i", "o", "p", "a", "s", "d", "f", "g", "h", "j", "k", "l", "z", "x", "c", "v", "b", "n", "m", "й", "ц", "у", "к", "е", "н", "г", "ш", "щ", "з", "х", "ъ", "ф", "ы", "в", "а", "п", "р", "о", "л", "д", "ж", "э", "я", "ч", "с", "м", "и", "т", "ь", "б", "ю", "ё"}  
set\_word\_symbols = {"й", "ц", "у", "к", "е", "н", "г", "ш", "щ", "з", "х", "ъ", "ф", "ы", "в", "а", "п", "р", "о", "л", "д", "ж", "э", "я", "ч", "с", "м", "и", "т", "ь", "б", "ю", "ё"}  
  
#текстовые блоки  
class TextBlock(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, text, x, y, width, height, fz):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.text = text  
 self.height = height  
 self.font\_size = fz  
 self.width = width  
 self.image = pygame.Surface((self.width, self.height))  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (x, y)  
 self.image.fill(WHITE)  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
 f1 = pygame.font.Font(None, self.font\_size)  
 text1 = f1.render(self.text, True, BLACK)  
 self.image.blit(text1, ((self.width - text1.get\_width()) / 2, (self.height - text1.get\_height()) / 2))  
  
#блок вводимого слова  
class ResBlock(pygame.sprite.Sprite):  
 def \_\_init\_\_(self):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.text = ''  
 self.height = 50  
 self.font\_size = 40  
  
 def update(self):  
 self.width = 20 + len(self.text) \* 20  
 self.image = pygame.Surface((self.width, self.height))  
 self.rect = self.image.get\_rect()  
 self.rect.centerx = WIDTH/2  
 self.rect.top = HEIGHT - (self.height + 20)  
 self.image.fill(WHITE)  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
 f1 = pygame.font.Font(None, self.font\_size)  
 text1 = f1.render(self.text, True, BLACK)  
 self.image.blit(text1, ((self.width - text1.get\_width()) / 2, (self.height - text1.get\_height()) / 2))  
  
 def set\_text(self, text):  
 self.text = text  
  
 def get\_text(self):  
 return self.text  
  
#кнопки для букв исходного слова  
class Button(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y, text, width, height, font\_size):  
 self.is\_pressed = False  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.text = text  
 self.width = width  
 self.height = height  
 self.font\_size = font\_size  
 self.image = pygame.Surface((width, height))  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (x, y)  
  
 def update(self):  
 if self.is\_pressed:  
 self.image.fill(GRAY)  
 else:  
 self.image.fill(WHITE)  
  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
 f1 = pygame.font.Font(None, self.font\_size)  
 text1 = f1.render(self.text, True, BLACK)  
 self.image.blit(text1, ((self.width - text1.get\_width()) / 2, (self.height - text1.get\_height()) / 2))  
  
 def pressed(self):  
 if not self.is\_pressed:  
 click\_snd.play()  
 global result  
 result += self.text  
 self.set\_pressed(True)  
  
 def set\_pressed(self, is\_pressed):  
 self.is\_pressed = is\_pressed  
  
#блок с изображением  
class ImageButton(pygame.sprite.Sprite):  
 def \_\_init\_\_(self, x, y, width, height, image):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.width = width  
 self.height = height  
 self.image = image  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (x, y)  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
  
#кнопка "готово"  
class AcceptButton(pygame.sprite.Sprite):  
 def \_\_init\_\_(self):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.width = 60  
 self.height = 60  
 self.image = accept\_img  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (WIDTH - (self.width + 20), HEIGHT - (self.height + 20))  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
  
 def pressed(self):  
 click\_snd.play()  
 global result  
 global level  
 global words  
 global forward  
  
 if result in words and len(results) < 40:  
 results.append(result)  
 words.remove(result)  
  
 if len(results) == 40 or len(words) == 0:  
 victory\_snd.play()  
 if level != 0:  
 forward = ImageButton(1137, 40, 103, 55, forward\_img)  
 all\_sprites.add(forward)  
  
 for i in range(len(users)):  
 if users[i]['name'] == user['name']:  
 users[i]['progress'][level - 1].append(result)  
  
 with open('users.txt', 'w') as users\_file:  
 json.dump(users, users\_file)  
  
 result = ''  
 for button in buttons:  
 button.set\_pressed(False)  
  
#кнопка "стереть"  
class EraseButton(pygame.sprite.Sprite):  
 def \_\_init\_\_(self):  
 pygame.sprite.Sprite.\_\_init\_\_(self)  
 self.width = 60  
 self.height = 60  
 self.image = erase\_img  
 self.rect = self.image.get\_rect()  
 self.rect.topleft = (WIDTH - (self.width + 100), HEIGHT - (self.height + 20))  
 self.border = pygame.rect.Rect(0, 0, self.width, self.height)  
 self.border\_color = BLACK  
 pygame.draw.rect(self.image, self.border\_color, self.border, 3)  
  
 def pressed(self):  
 click\_snd.play()  
 global result  
 result = ''  
 for button in buttons:  
 button.set\_pressed(False)  
  
#процедура для создания текстовых блоков  
def new\_tb(text, x, y, width, height, fz):  
 tb = TextBlock(text, x, y, width, height, fz)  
 all\_sprites.add(tb)  
  
#процедура для формирования сетки ответов  
def form\_results():  
 global results  
 x = 50  
 y = 150  
 for i in range(len(results)):  
 new\_tb(results[i], x, y, 145, 40, 24)  
 y += 55  
 if (i + 1) % 5 == 0:  
 y = 150  
 x += 150  
  
#процедура для вывода текста на экран  
def print\_text(surface, x, y, text, fz):  
 font = pygame.font.Font(None, fz)  
 text\_img = font.render(text, True, BLACK)  
 text\_rect = text\_img.get\_rect()  
 text\_rect.topleft = (x, y)  
 surface.blit(text\_img, text\_rect)  
  
#процедура для вывода экрана "задать слово"  
def show\_set\_word():  
 global word  
 global words  
 global level  
  
 screen.blit(bg\_img, bg\_rect)  
  
 set\_word\_sprites = pygame.sprite.Group()  
  
 print\_text(screen, 330, 280, 'ВВЕДИТЕ ИСХОДНОЕ СЛОВО:', 56)  
  
 form = TextBlock(word, 360, 330, 560, 50, 36)  
 accept = ImageButton(950, 325, 60, 60, accept\_img)  
 back = ImageButton(40, 40, 103, 55, back\_img)  
  
 set\_word\_sprites.add(back)  
 set\_word\_sprites.add(form)  
 set\_word\_sprites.add(accept)  
  
 set\_word\_sprites.draw(screen)  
  
 pygame.display.flip()  
  
 # цикл ожидания  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 # обработка события  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 # проверка нахождения курсора на кнопке  
 if accept.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
  
 #загрузка из файла  
 with open('dictionary.txt', 'r') as dictionary\_file:  
 dictionary = json.load(dictionary\_file)  
  
 word = word.lower()  
 #поиск слова в словаре  
 found = False  
 for dictionary\_element in dictionary:  
 if dictionary\_element['source'] == word:  
 found = True  
 for type\_elem in dictionary\_element['type']:  
 words.append(type\_elem.upper())  
 break  
  
 # оповещение, если слово не найдено  
 if not found:  
 word = ''  
 form = TextBlock(word, 360, 330, 560, 50, 36)  
 set\_word\_sprites.add(form)  
  
 print\_text(screen, 500, 400, 'Слово не найдено!', 48)  
 pygame.display.flip()  
  
 time = pygame.time.get\_ticks()  
 while True:  
 now = pygame.time.get\_ticks()  
 if now - time > 1000:  
 break  
  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 330, 280, 'ВВЕДИТЕ ИСХОДНОЕ СЛОВО:', 56)  
 set\_word\_sprites.draw(screen)  
 pygame.display.flip()  
  
 continue  
  
 level = 0  
  
 # переход к экранной форме игры  
 show\_game()  
  
 form = TextBlock(word, 360, 330, 560, 50, 36)  
 set\_word\_sprites.add(form)  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 330, 280, 'ВВЕДИТЕ ИСХОДНОЕ СЛОВО:', 56)  
 set\_word\_sprites.draw(screen)  
 pygame.display.flip()  
  
 if back.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 waiting = False  
  
 # ввод слова  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_BACKSPACE:  
 word = word[:-1]  
 elif len(word) < 13 and event.unicode in set\_word\_symbols:  
 word += event.unicode  
  
 form = TextBlock(word, 360, 330, 560, 50, 36)  
 set\_word\_sprites.add(form)  
 set\_word\_sprites.draw(screen)  
  
 pygame.display.flip()  
  
#процедура для вывода экрана ввода имени  
def show\_profile():  
 global user  
 username = ''  
  
 screen.blit(bg\_img, bg\_rect)  
  
 # вывод названия игры  
 logo\_img.set\_colorkey((255, 0, 0))  
 screen.blit(logo\_img, (177, 60))  
  
 profile\_sprites = pygame.sprite.Group()  
  
 print\_text(screen, 330, 280, 'ВВЕДИТЕ ИМЯ ПОЛЬЗОВАТЕЛЯ:', 56)  
  
 form = TextBlock(username, 360, 330, 560, 50, 36)  
 accept = ImageButton(950, 325, 60, 60, accept\_img)  
  
 profile\_sprites.add(form)  
 profile\_sprites.add(accept)  
  
 profile\_sprites.draw(screen)  
  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 if accept.rect.collidepoint(mouse\_pos) and len(username) > 0:  
 with open('users.txt') as users\_file:  
 users = json.load(users\_file)  
  
 # поиск пользователя  
 found = False  
 for i in range(len(users)):  
 if users[i]['name'] == username:  
 # если найден, берем данные  
 found = True  
 user = {  
 'name': users[i]['name'],  
 'progress': users[i]['progress']  
  
 }  
  
 #если не найден, создаём нового пользователя  
 if not found:  
 user = {  
 'name': username,  
 'progress': [[], [], [], [], [], [], [], [], [], [], [], [], [], [], []]  
 }  
 # добавление нового пользователя и запись в файл  
 users.append(user)  
 with open('users.txt', 'w') as users\_file:  
 json.dump(users, users\_file)  
  
 click\_snd.play()  
 waiting = False  
  
 # ввод имени пользователя  
 if event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_BACKSPACE:  
 username = username[:-1]  
 elif len(username) < 32 and event.unicode in allowed\_symbols:  
 username += event.unicode  
  
 form = TextBlock(username, 360, 330, 560, 50, 36)  
 profile\_sprites.add(form)  
 profile\_sprites.draw(screen)  
  
 pygame.display.flip()  
  
#процедура для вывода экрана игры  
def show\_game():  
 global word  
 global level  
 global words  
 global results  
 global forward  
  
 resblock = ResBlock()  
 erase\_btn = EraseButton()  
 accept\_btn = AcceptButton()  
 all\_sprites.add(resblock)  
 all\_sprites.add(erase\_btn)  
 all\_sprites.add(accept\_btn)  
  
 back = ImageButton(40, 40, 103, 55, back\_img)  
 all\_sprites.add(back)  
  
 # вывод кнопки "далее"  
 if level != 0 and (len(results) == 40 or len(words) == 0):  
 forward = ImageButton(1137, 40, 103, 55, forward\_img)  
 all\_sprites.add(forward)  
  
 word = word.upper()  
 letter\_width = 80  
 letter\_spacing = 20  
 letter\_font\_size = 52  
  
 #вывод букв исходного слова  
 x = (WIDTH - len(word) \* letter\_width - (len(word) - 1) \* letter\_spacing) / 2  
 for letter in word:  
 button = Button(x, 504, letter, letter\_width, letter\_width, letter\_font\_size)  
 all\_sprites.add(button)  
 buttons.add(button)  
 x += letter\_width + letter\_spacing  
  
 waiting = True  
 while waiting:  
 # Держим цикл на правильной скорости  
 clock.tick(FPS)  
 # Ввод процесса (события)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 for button in buttons:  
 if button.rect.collidepoint(mouse\_pos):  
 button.pressed()  
  
 if accept\_btn.rect.collidepoint(mouse\_pos):  
 accept\_btn.pressed()  
  
 if erase\_btn.rect.collidepoint(mouse\_pos):  
 erase\_btn.pressed()  
  
 #обнуление переменных при выходе с уровня  
 if back.rect.collidepoint(mouse\_pos):  
 global result  
 result = ''  
 results = []  
 word = ''  
 words = []  
 click\_snd.play()  
 waiting = False  
  
 #переход на следующий уровень  
 if forward != None and forward.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 forward = None  
  
 level += 1  
 word = levels[level - 1]['word']  
 words = levels[level - 1]['words']  
  
 results = user['progress'][level - 1]  
  
 # удаление уже введенныз слов из списка возможных  
 for i in range(len(results)):  
 if results[i] in words:  
 words.remove(results[i])  
  
 all\_sprites.empty()  
 buttons.empty()  
  
 all\_sprites.add(resblock)  
 all\_sprites.add(erase\_btn)  
 all\_sprites.add(accept\_btn)  
 all\_sprites.add(back)  
  
 if level != 0 and (len(results) == 40 or len(words) == 0):  
 forward = ImageButton(1137, 40, 103, 55, forward\_img)  
 all\_sprites.add(forward)  
  
 word = word.upper()  
 letter\_width = 80  
 letter\_spacing = 20  
 letter\_font\_size = 52  
  
 x = (WIDTH - len(word) \* letter\_width - (len(word) - 1) \* letter\_spacing) / 2  
 for letter in word:  
 button = Button(x, 504, letter, letter\_width, letter\_width, letter\_font\_size)  
 all\_sprites.add(button)  
 buttons.add(button)  
 x += letter\_width + letter\_spacing  
  
 # Обновление  
 resblock.set\_text(result)  
 all\_sprites.update()  
 form\_results()  
  
 # Рендеринг  
 screen.blit(bg\_img, bg\_rect)  
 all\_sprites.draw(screen)  
 print\_text(screen, 490, 50, 'УРОВЕНЬ ' + str(level), 72)  
 # После отрисовки всего, переворачиваем экран  
 pygame.display.flip()  
  
 all\_sprites.empty()  
 buttons.empty()  
  
#процедура для вывода экрана "выбора уровня"  
def show\_levels():  
 global words  
 global word  
 global level  
 global results  
  
 screen.blit(bg\_img, bg\_rect)  
  
 blocks = pygame.sprite.Group()  
 buttons = pygame.sprite.Group()  
  
 back = ImageButton(40, 40, 103, 55, back\_img)  
 buttons.add(back)  
  
 x = 295  
 y = 165  
  
 # поиск последнего выполненного уровня  
 last\_level = 1  
 for i in range(1, len(user['progress'])):  
 if user['progress'][i] == []:  
 last\_level = i  
 break  
  
 # создание сетки уровней  
 for i in range(last\_level):  
 tb = TextBlock(str(i + 1), x, y, 90, 90, 56)  
 blocks.add(tb)  
 x += 150  
  
 if (i + 1) % 5 == 0:  
 x = 295  
 y += 150  
  
 buttons.draw(screen)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 for block in blocks:  
 # переход на выбранный уровень  
 if block.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 level = int(block.text)  
 word = levels[level - 1]['word']  
 words = levels[level - 1]['words']  
  
 results = user['progress'][level - 1]  
  
 for i in range(len(results)):  
 if results[i] in words:  
 words.remove(results[i])  
  
 show\_game()  
  
 blocks.empty()  
  
 last\_level = 1  
 for i in range(1, len(user['progress'])):  
 if user['progress'][i] == []:  
 last\_level = i  
 break  
  
 x = 295  
 y = 165  
  
 for i in range(last\_level):  
 tb = TextBlock(str(i + 1), x, y, 90, 90, 56)  
 blocks.add(tb)  
 x += 150  
  
 if (i + 1) % 5 == 0:  
 x = 295  
 y += 150  
  
 screen.blit(bg\_img, bg\_rect)  
 buttons.draw(screen)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
 if back.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 waiting = False  
  
#процедура для вывода экрана "играть"  
def show\_play():  
 options = ['ПРОДОЛЖИТЬ', 'ЗАДАТЬ СЛОВО']  
 blocks = pygame.sprite.Group()  
 buttons = pygame.sprite.Group()  
  
 screen.blit(bg\_img, bg\_rect)  
  
 # реализация пунктов меню  
 for i in range(len(options)):  
 tb = TextBlock(options[i], 455, 240 + (i \* 150), 370, 90, 56)  
 blocks.add(tb)  
  
 back = ImageButton(40, 40, 103, 55, back\_img)  
 buttons.add(back)  
  
 buttons.draw(screen)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 for block in blocks:  
 if block.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 match block.text:  
 case 'ПРОДОЛЖИТЬ':  
 show\_levels()  
  
 screen.blit(bg\_img, bg\_rect)  
 buttons.draw(screen)  
 blocks.draw(screen)  
 pygame.display.flip()  
  
 case 'ЗАДАТЬ СЛОВО':  
 show\_set\_word()  
  
 screen.blit(bg\_img, bg\_rect)  
 buttons.draw(screen)  
 blocks.draw(screen)  
 pygame.display.flip()  
  
 if back.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 waiting = False  
  
#процедура для вывода правил  
def show\_rules():  
 surf = pygame.surface.Surface((640, 360))  
 surf.fill((229, 229, 229))  
 rules = [  
 'Механика игры состоит в составлении максимального количества',  
 'слов из представленного игроку исходного слова.',  
 '',  
 'После составления всех возможных слов появится возможность',  
 'перехода на следующий уровень.',  
 '',  
 'Слова, получаемые в результате, должны удовлетворять следующим',  
 'требованиям:',  
 ' -имя существительное;',  
 ' -именительный падеж;',  
 ' -минимальная длина - три буквы.'  
 ]  
  
 for i in range(len(rules)):  
 print\_text(surf, 30, 70 + (i \* 20), rules[i], 24)  
  
 screen.blit(surf, (320, 180))  
 pygame.draw.rect(screen, BLACK, pygame.rect.Rect(320, 180, 640, 360), 3)  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 click\_snd.play()  
 waiting = False  
  
#процедура для отрисовки ползунка  
def draw\_slider():  
 pygame.draw.line(screen, BLACK, (570, 270), (840, 270), 6)  
 pygame.draw.line(screen, BLACK, (570, 285), (570, 255), 6)  
 pygame.draw.line(screen, BLACK, (840, 285), (840, 255), 6)  
  
 pygame.draw.line(screen, BLACK, (560, 235), (580, 235), 6)  
  
 pygame.draw.line(screen, BLACK, (830, 235), (850, 235), 6)  
 pygame.draw.line(screen, BLACK, (840, 225), (840, 245), 6)  
  
#процедура для вывода экрана "настройки"  
def show\_settings():  
 global volume  
 start = 570  
 end = 840  
  
 screen.blit(bg\_img, bg\_rect)  
 buttons = pygame.sprite.Group()  
 back = ImageButton(40, 40, 103, 55, back\_img)  
 buttons.add(back)  
  
 buttons.draw(screen)  
  
 print\_text(screen, 400, 250, 'ЗВУК', 64)  
 draw\_slider()  
  
 # рисование ползунка  
 circle = pygame.surface.Surface((30, 30))  
 circle.fill(WHITE)  
 circle.set\_colorkey(WHITE)  
 circle\_rect = circle.get\_rect()  
 pygame.draw.circle(circle, BLACK, circle\_rect.center, 15)  
 circle\_rect.center = (start + (end - start) \* volume, 270)  
  
 screen.blit(circle, circle\_rect)  
  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 if back.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 waiting = False  
  
 mouse\_pos = pygame.mouse.get\_pos()  
 mouse\_buttons = pygame.mouse.get\_pressed()  
 # ограничение отзывчивой зоны ползунка  
 if mouse\_pos[0] in range(555, 855) and mouse\_pos[1] in range(255, 285) and mouse\_buttons[0]:  
 circle\_rect.center = (mouse\_pos[0], 270)  
  
 if circle\_rect.centerx < start:  
 circle\_rect.centerx = start  
  
 if circle\_rect.centerx > end:  
 circle\_rect.centerx = end  
  
 screen.blit(bg\_img, bg\_rect)  
 buttons.draw(screen)  
 print\_text(screen, 400, 250, 'ЗВУК', 64)  
 draw\_slider()  
 screen.blit(circle, circle\_rect)  
  
 # вычисление громкости  
 volume = (circle\_rect.centerx - start) / (end - start)  
 click\_snd.set\_volume(volume)  
 victory\_snd.set\_volume(volume)  
  
 # запись громкости в файл настроек  
 with open('settings.txt', 'w') as settings\_file:  
 json.dump(volume, settings\_file)  
  
 pygame.display.flip()  
  
#процедура для вывода экрана меню  
def show\_menu():  
 global user  
 options = ['ИГРАТЬ', 'ПРАВИЛА', 'НАСТРОЙКИ', 'ВЫХОД']  
  
 blocks = pygame.sprite.Group()  
  
 screen.blit(bg\_img, bg\_rect)  
  
 print\_text(screen, 1010, 40, 'ИГРОК:', 36)  
  
 #реализация пунктов меню  
 for i in range(len(options)):  
 tb = TextBlock(options[i], 455, 90 + (i \* 150), 370, 90, 64)  
 blocks.add(tb)  
  
 user\_block = TextBlock(user['name'], 1110, 35, 120, 30, 36)  
 blocks.add(user\_block)  
  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
 waiting = True  
 while waiting:  
 clock.tick(FPS)  
 for event in pygame.event.get():  
 if event.type == pygame.QUIT:  
 pygame.quit()  
 if event.type == pygame.MOUSEBUTTONDOWN:  
 if event.button == pygame.BUTTON\_LEFT:  
 mouse\_pos = pygame.mouse.get\_pos()  
 for block in blocks:  
 if block.rect.collidepoint(mouse\_pos):  
 click\_snd.play()  
 match block.text:  
 case 'ИГРАТЬ':  
 show\_play()  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 1010, 40, 'ИГРОК:', 36)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
 case 'ПРАВИЛА':  
 show\_rules()  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 1010, 40, 'ИГРОК:', 36)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
 case 'НАСТРОЙКИ':  
 show\_settings()  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 1010, 40, 'ИГРОК:', 36)  
 blocks.draw(screen)  
  
 pygame.display.flip()  
 case 'ВЫХОД':  
 pygame.quit()  
  
 if block.text == user['name']:  
 show\_profile()  
 screen.blit(bg\_img, bg\_rect)  
 print\_text(screen, 1010, 40, 'ИГРОК:', 36)  
  
 blocks.remove(user\_block)  
 user\_block = TextBlock(user['name'], 1110, 35, 120, 30, 36)  
 blocks.add(user\_block)  
  
 blocks.draw(screen)  
  
 pygame.display.flip()  
  
all\_sprites = pygame.sprite.Group()  
buttons = pygame.sprite.Group()  
  
show\_profile()  
show\_menu()  
  
pygame.quit()