```python
import pygame.font

class Button():
    def __init__(self, screen, msg):
        """Initialize button attributes."""
        self.width, self.height = 200, 50
        self.screen = screen
        self.screen_rect = screen.get_rect()
        # Set the dimensions and properties of the button. self.width, self.height = 200, 50
        self.button_color = (0, 0, 0)
        self.text_color = (255, 255, 255)
        self.font = pygame.font.SysFont(None, 48)
        # Build the button's rect object and center it.
        self.rect = pygame.Rect(0, 0, self.width, self.height)
        self.rect.center = self.screen_rect.center
        # The button message needs to be prepped only once.
        self.prep_msg(msg)

    def prep_msg(self, msg):
        """Turn msg into a rendered image and center text on the button."""
        self.msg_image = self.font.render(msg, True, self.text_color,
        self.button_color)
        self.msg_image_rect = self.msg_image.get_rect()
        self.msg_image_rect.center = self.rect.center

    def draw_button(self):
        # Draw blank button and then draw message.
        self.screen.fill(self.button_color, self.rect)
        self.screen.blit(self.msg_image, self.msg_image_rect)
```

```python
import pygame

class Background_pause(pygame.sprite.Sprite):
    def __init__(self, image_file, location):
        pygame.sprite.Sprite.__init__(self)  #call Sprite initializer
        self.image = pygame.image.load(image_file)
        self.rect = self.image.get_rect()
        self.rect.left = location
        self.rect.top = location


class Background_game(pygame.sprite.Sprite):
    def __init__(self, image_file, location):
        pygame.sprite.Sprite.__init__(self)  #call Sprite initializer
        self.image = pygame.image.load(image_file)
        self.rect = self.image.get_rect()
        self.rect.left = location
        self.rect.top = location
```

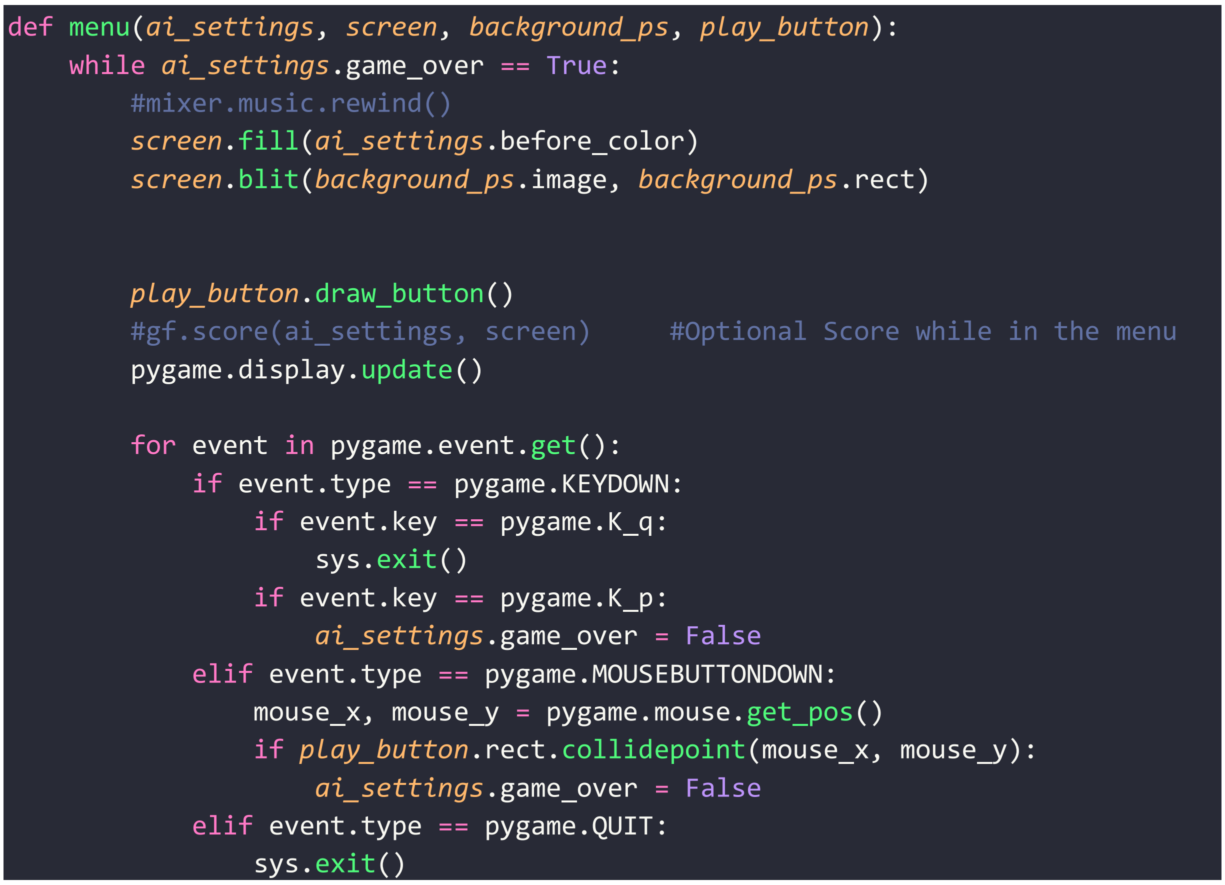
```python
import pygame

import random

class Fruit:
    def __init__(self, ai_settings, screen):
        self.ai_settings = ai_settings
        self.screen = screen

        self.image = pygame.image.load('images/apple.bmp')
        self.rect = self.image.get_rect()
        self.rect.x =  random.randint(50, ai_settings.screen_width-50)
        self.rect.y =  random.randint(30, ai_settings.screen_height-30)

        self.x = float(self.rect.x)
        self.y = float(self.rect.y)

    def blitme(self):
        """Draw the apple at its current location."""
        self.screen.blit(self.image, self.rect)
```

```python
import pygame
import time
import sys
from pygame import mixer

import game_functions as gf

class Snake:
    def __init__(self, ai_settings):
        self.x = ai_settings.screen_width/2
        self.y = ai_settings.screen_height/2
        self.x_change = 0
        self.y_change = 0
        self.rect = pygame.Rect(self.x, self.y, ai_settings.snake_width,
        ai_settings.snake_height)
        self.snake_list = []
        self.length_snake = 1
        #self.apple_sound1 = mixer.Sound("music/mary_sound1.wav")


    def move_snake(self, ai_settings):
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit()
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RIGHT and self.x_change >= 0: #Move t
he snake only if it is not moving in the opposite direction

                    #self.apple_sound1.play()
                    self.x_change = ai_settings.snake_speed
                    self.y_change = 0

                elif event.key == pygame.K_LEFT and self.x_change <= 0:
                    #apple_sound = mixer.Sound("music/mary_sound1.wav")
                    #self.apple_sound1.play()
                    self.x_change = -ai_settings.snake_speed
                    self.y_change = 0


                elif event.key == pygame.K_UP and self.y_change <= 0:
                    self.y_change = -ai_settings.snake_speed
                    self.x_change = 0
                    #apple_sound = mixer.Sound("music/mary_sound1.wav")
                    #apple_sound.play()


                elif event.key == pygame.K_DOWN and self.y_change >= 0:
                    #apple_sound = mixer.Sound("music/mary_sound1.wav")
                    #apple_sound.play()
                    self.y_change = ai_settings.snake_speed
                    self.x_change = 0

        self.rect.x += self.x_change
        self.rect.y += self.y_change
        self.x = float(self.rect.x)
        self.y = float(self.rect.y)

    def in_game(self, ai_settings, screen):
        difficulty = "hard"
        if difficulty == "hard":
            #self.x += self.x_change
            if self.rect.x >= ai_settings.screen_width - 10:
                #self.rect.x = 0
                self.reset(ai_settings, screen)
            elif self.rect.x < 0 - 5:
                #self.rect.x = ai_settings.screen_width
                self.reset(ai_settings, screen)
            #self.y += self.y_change
            if self.rect.y >= ai_settings.screen_height:
                #self.rect.y = 0
                self.reset(ai_settings, screen)
            elif self.rect.y < 0 - 5:
                #self.rect.y = ai_settings.screen_height
                self.reset(ai_settings, screen)

    def draw_snake(self, screen, ai_settings):
        for i in range(len(self.snake_list)):
            pygame.draw.rect(screen, ai_settings.snake_color,
            [self.snake_list[i][0], self.snake_list[i][1], ai_settings.snake_w
idth,
             ai_settings.snake_height])
            # x = snake_list[i][0]
            # y = snake_list[i][1]

    def snake_update(self, ai_settings, screen):
        self.snake_head = []
        self.snake_head.append(self.x)
        self.snake_head.append(self.y)
        self.snake_list.append(self.snake_head)
        if len(self.snake_list) > self.length_snake:
            del self.snake_list[0]


        for x in self.snake_list[:-1]:
            if x == self.snake_head:
                self.reset(ai_settings, screen)
```

```python
    def reset(self, ai_settings, screen):
        ai_settings.game_over = True
        ai_settings.score = 0
        gf.message("Game Over!", ai_settings.message_color,
        ai_settings, screen, 2.5, 2.5)
        pygame.display.update()
        self.x_change = 0
        self.y_change = 0
        self.rect.x = ai_settings.screen_width/2
        self.rect.y = ai_settings.screen_height/2
        self.x = float(self.rect.x)
        self.y = float(self.rect.y)
        self.length_snake = 1
        self.snake_list.clear()
        mixer.music.pause()
        game_over_sound = mixer.Sound("music/game_over.wav")
        game_over_sound.play()
        time.sleep(2)
        mixer.music.unpause()
```

```python
import pygame

class Settings():
    """A class to store all settings for Alien Invasion."""
    def __init__(self):
        """Initialize the game's settings."""
        # Screen settings
        self.screen_width = 600
        self.screen_height = 600
        self.bg_color = (126,200,80) #Green

        # Snake settings
        self.snake_color = 0, 0, 0 #Black
        self.snake_width = 15
        self.snake_height = 15
        self.snake_speed = 10

        #Score settings
        self.score = 0
        self.score_font = pygame.font.SysFont("comicsansms", 20)
        self.score_color = (255, 255, 255) #White
        self.message_style = pygame.font.SysFont("bahnschrift", 35)
        self.message_color = (255, 0, 0)

        #Menu settings
        self.before_color = (255, 255, 255)


        self.game_over = True
```

```python
import random

import pygame
from pygame import mixer

import sys

def snake_and_fruit(snake, fruit, ai_settings):
    #What happends when tha snake eats the fruit

    if snake.rect.colliderect(fruit.rect):
        #If there is music I lower it and play eating sound
        mixer.music.set_volume(0.1)
        mixer.music.pause()
        apple_sound = mixer.Sound("music/apple_sound4.wav")
        apple_sound.play()

        #New spawn point for the fruit
        fruit.rect.x =  random.randint(30, ai_settings.screen_width-30)
        fruit.rect.y =  random.randint(30, ai_settings.screen_height-30)

        for x in snake.snake_list[:-1]:
            while fruit.rect.x == x[0] or fruit.rect.y == x[1]:
                fruit.rect.x =  random.randint(30, ai_settings.screen_width-
30)
                fruit.rect.y =  random.randint(30, ai_settings.screen_height-
30)


        fruit.x = float(fruit.rect.x)
        fruit.y = float(fruit.rect.y)

        snake.length_snake += 2  #How fast the snake grows
        ai_settings.score += 1   #Increase the score
        mixer.music.unpause()
        mixer.music.set_volume(0.5)

def score(ai_settings, screen):
    #Display score function
    value = ai_settings.score_font.render("Score: " + str(ai_settings.score),
True, ai_settings.score_color)
    screen.blit(value, [10, 10])

def message(msg, color,ai_settings, screen,w ,h):
    #Display message function
    mesg = ai_settings.message_style.render(msg, True, color)
    screen.blit(mesg, [ai_settings.screen_width / w, ai_settings.screen_height
 / h])
```



```python
def menu(ai_settings, screen, background_ps, play_button):
    while ai_settings.game_over == True:
        #mixer.music.rewind()
        screen.fill(ai_settings.before_color)
        screen.blit(background_ps.image, background_ps.rect)


        play_button.draw_button()
        #gf.score(ai_settings, screen)     #Optional Score while in the menu
        pygame.display.update()

        for event in pygame.event.get():
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_q:
                    sys.exit()
                if event.key == pygame.K_p:
                    ai_settings.game_over = False
            elif event.type == pygame.MOUSEBUTTONDOWN:
                mouse_x, mouse_y = pygame.mouse.get_pos()
                if play_button.rect.collidepoint(mouse_x, mouse_y):
                    ai_settings.game_over = False
            elif event.type == pygame.QUIT:
                sys.exit()
```

```python
import pygame
import game_functions as gf

from settings import Settings
from the_snake import Snake
from the_fruit import Fruit
from button import Button
from background import Background_pause
from background import Background_game
from pygame import mixer

pygame.mixer.pre_init(44100, -16, 1, 512)
pygame.init()
ai_settings = Settings()
screen = pygame.display.set_mode((ai_settings.screen_width, ai_settings.screen
_height))
snake = Snake(ai_settings)
fruit = Fruit(ai_settings, screen)

clock = pygame.time.Clock()


#background_game = Background_game("images/grass_bg.png", [0][0])
background_ps = Background_pause("images/snake_bg2.png", [0][0])
play_button = Button(screen, "Play")
pygame.display.set_caption("Snake Game")
#game_over = True

mixer.music.load("music/snake_song.mp3")
mixer.music.play(loops = -1, start = 5)
mixer.music.set_volume(0.5)


#pygame.mixer.Channel(0).play(pygame.mixer.Sound("music/snake_song.mp3"))
#pygame.mixer.Channel(1).play(pygame.mixer.Sound('music\game_over.mp3'))

while True:

    mixer.music.rewind()
    gf.menu(ai_settings, screen, background_ps, play_button)

    snake.move_snake(ai_settings)
    snake.in_game(ai_settings, screen)

    screen.fill(ai_settings.bg_color)
    #screen.blit(background_game.image, background_game.rect)
    gf.score(ai_settings, screen)
    fruit.blitme()
    snake.snake_update(ai_settings,screen)
    snake.draw_snake(screen, ai_settings)


    pygame.display.update()

    gf.snake_and_fruit(snake, fruit, ai_settings)


    clock.tick(30)
```