**CMSC 150 Fall 2022**

**Game (200 points available)**

**Part 1 Hand-in: Midnight Thursday November 17**

Create a game containing all the following elements. If you want to create a different type of game which does not contain all of these elements, **talk to me or email me first** about it.

**Due on November 175h: (70 points)**

1. Description of your game, how you play, how it is scored, and how it gets harder with time. (10 points)

My game will allow the user to use the keyboard to jump and squat to avoid various obstacles as they appear in the game. You play by simply using the up and down arrows on the keyboard to make your character jump and squat at the right time. If your character does this at the wrong time and makes contact with one of the obstacles, the game is over. The speed of the obstacles appearing will increase over time, and the player will earn points by lasting longer. The amount of points earned will also increase over time.

1. Initial program with your initial plan for background, sprite images shown on it (not necessarily working yet), sounds, etc.. This may change as you go forward and that is ok

(10 points)

self.player\_sprite = arcade.Sprite("zombie\_walk7.png", PLAYER\_SCALING)

arcade.set\_background\_color(arcade.color.SKY\_BLUE)

self.standard = arcade.load\_sound("arcade\_resources\_sounds\_hit2.wav")

This first code is the character. I have it set to a zombie currently, but I’d like to have multiple characters the player can choose to use.

The second code is the background. I have set the background color to sky blue, and it will simulate the character in the great outdoors with the sunny, light blue sky.

The third code adds a preliminary sound I’d like to use and activate each time the player jumps. It doesn’t work yet, but I plan on integrating it later on in the code.

1. Description of how each sprite will move and all user interactions with the game

(10 points)

The player’s character will move forward automatically across the screen with no user interaction on the keyboard. The user will push the up key on the keyboard to make the character jump over the barriers on the lower level (ground) that would otherwise hit the character’s legs on the screen. The user will push the down key on the keyboard to make the character lean forward (squat) under the barriers in the air that would otherwise hit the character’s head on the screen. These two controls are the only interaction the player has with the keyboard in the game.

1. Pseudo code for the main and each sprite class you plan to implement (10 points)

As stated in our meeting, the Pseudo code step listed here is mainly to see that I have a plan for my game. I have attached the code I have so far below. I still need to add gravity, add a moving camera, and get my blocks in the correct position. It’s going to be a fun journey, and I’m excited to make this framework and structure a reality. It will require at least two – maybe three – phases to the game: game intro, game running, and possibly game over.

1. Program that displays the background, initial text with information from step 1, sprites (even if they are not yet moving correctly), and score (even if it is not updating correctly yet). (30 points)

Here is what I have so far conceptually for my game. I have a long way to go, but I’ve included some preliminary code for the moving camera, blocks, and sprites. I also have a game score, initial text to be shown, and a general background color to be shown throughout the game. Please let me know if this isn’t sufficient.

import arcade  
  
SPRITE\_SCALING = 0.5  
  
SCREEN\_WIDTH = 800  
SCREEN\_HEIGHT = 600  
SCREEN\_TITLE = "Sprite Move with Walls Example"  
GAME\_INTRO = 1  
GAME\_RUNNING = 2  
GAME\_OVER = 3  
MOVEMENT\_SPEED = 5  
  
  
class MyGame(arcade.Window):  
 *""" Main application class. """* def \_\_init\_\_(self, width, height, title):  
 *"""  
 Initializer  
 """* super().\_\_init\_\_(width, height, title)  
  
 # Sprite lists  
 self.coin\_list = None  
 self.wall\_list = None  
 self.player\_list = None  
  
 # Set up the player  
 self.player\_sprite = None  
 self.physics\_engine = None  
  
 #Sound  
 self.standard = arcade.load\_sound("arcade\_resources\_sounds\_hit2.wav")  
  
 def setup(self):  
 *""" Set up the game and initialize the variables. """* # Sprite lists  
 self.player\_list = arcade.SpriteList()  
 self.wall\_list = arcade.SpriteList()  
  
 # Set up the player  
 self.player\_sprite = arcade.Sprite(":resources:images/animated\_characters/female\_person/femalePerson\_idle.png",  
 SPRITE\_SCALING)  
 self.player\_sprite.center\_x = 50  
 self.player\_sprite.center\_y = 64  
 self.player\_sprite.angle = 0  
 self.player\_sprite.change\_x = 1  
 self.player\_list.append(self.player\_sprite)  
 self.current\_state = GAME\_INTRO  
  
 # -- Set up the walls  
 # Create a row of boxes  
 for x in range(173, 650, 64):  
 wall = arcade.Sprite(":resources:images/tiles/boxCrate\_double.png",  
 SPRITE\_SCALING)  
 wall.center\_x = x  
 wall.center\_y = 0  
 self.wall\_list.append(wall)  
  
 # Create a column of boxes  
 for y in range(273, 500, 64):  
 wall = arcade.Sprite(":resources:images/tiles/boxCrate\_double.png",  
 SPRITE\_SCALING)  
 wall.center\_x = 465  
 wall.center\_y = y  
 self.wall\_list.append(wall)  
  
 self.physics\_engine = arcade.PhysicsEngineSimple(self.player\_sprite,  
 self.wall\_list)  
  
 # Set the background color  
 arcade.set\_background\_color(arcade.color.SKY\_BLUE)  
  
  
 if self.current\_state == GAME\_INTRO:  
 arcade.draw\_text("Welcome to Running Man", SCREEN\_WIDTH // 3, SCREEN\_HEIGHT // 2 + 30, arcade.color.BLACK,  
 25)  
 arcade.draw\_text("Use the Up Key and Down Key to jump and squat to avoid the obstacles", SCREEN\_WIDTH // 3, SCREEN\_HEIGHT // 2, arcade.color.BLACK,  
 25)  
 arcade.draw\_text("Press the space bar to continue.", SCREEN\_WIDTH // 3, (SCREEN\_HEIGHT // 2 - 30),  
 arcade.color.BLACK, 25)  
  
 elif self.current\_state == GAME\_RUNNING:  
 self.draw\_game()  
  
 else:  
 ##End game  
 self.draw\_game\_over()  
  
 def draw\_game(self):  
 self.on\_draw()  
  
  
 def on\_draw(self):  
 *"""  
 Render the screen.  
 """* # This command has to happen before we start drawing  
 self.clear()  
  
 # Draw all the sprites.  
 self.wall\_list.draw()  
 self.player\_list.draw()  
  
 def on\_key\_press(self, key, modifiers):  
 *"""Called whenever a key is pressed. """* if key == arcade.key.UP:  
 self.player\_sprite.change\_y = MOVEMENT\_SPEED  
 elif key == arcade.key.DOWN:  
 self.player\_sprite.angle = 270  
 elif key == arcade.key.LEFT:  
 self.player\_sprite.change\_x = -MOVEMENT\_SPEED  
 elif key == arcade.key.RIGHT:  
 self.player\_sprite.change\_x = MOVEMENT\_SPEED  
  
 def on\_key\_release(self, key, modifiers):  
 *"""Called when the user releases a key. """* if key == arcade.key.UP or key == arcade.key.DOWN:  
 self.player\_sprite.angle = 0  
 self.player\_sprite.change\_y = 0  
 elif key == arcade.key.LEFT or key == arcade.key.RIGHT:  
 self.player\_sprite.change\_x = 0  
  
 def on\_update(self, delta\_time):  
 *""" Movement and game logic """* # Call update on all sprites (The sprites don't do much in this  
 # example though.)  
 self.physics\_engine.update()  
  
  
def main():  
 *""" Main function """* window = MyGame(SCREEN\_WIDTH, SCREEN\_HEIGHT, SCREEN\_TITLE)  
 window.setup()  
 arcade.run()  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()