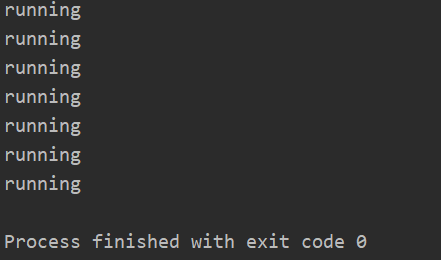
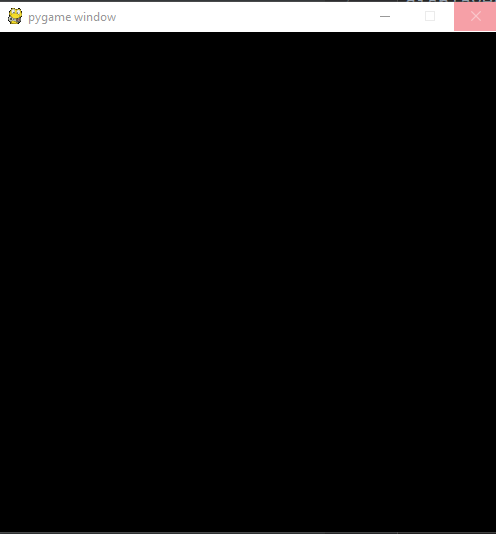
**Project Development Log – Joseph Henry**

**Prototype 1 Development (The Multiplayer):**

I first made a basic client program that opened a display window; this will be used to display the game to the players.

import pygame  
displayWidth = 500  
displayHeight = 500  
gameDisplay = pygame.display.set\_mode((displayWidth, displayHeight))  
def main():  
 run = True  
 while run:  
 print("running")  
 pygame.quit()  
main()

This worked just as expected by displaying a small box on the screen and printing the word “running” in the console continuously until the code was stopped from running.

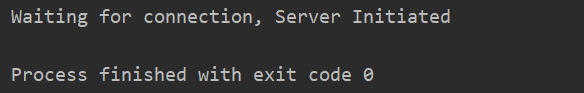
I then began work on creating a server for the multiplayer to run off. I did this by opening a socket for the clients to connect to.

import socket

# Local Ip (IPV4 FROM CMD IPCONFIG, DEVICE SPECIFIC)  
server = socket.gethostbyname(socket.gethostname())  
  
# Server Port  
port = 13010  
  
s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  
  
# The try and except are used to test to see if the port is open. If it is open it will bind the server to the port and  
# if not then the program will print out e to show that the port is in use (e for error).  
try:  
 s.bind((server, port))

except socket.error as e:  
 str(e)

# opens up port to allow us to begin to use it to connect multiple clients together. Inside the brackets limits the  
# number of possible connections  
s.listen(2)  
print("Waiting for connection, Server Initiated")

Running this code produced the response “**Waiting for connection, Server Initiated**” in the console and no errors, this means that the port I am using is open and the socket is working fine. It also shows that the port is ready to receive a connection from an external IP.

Then I made a way for the socket to handle incoming connections.

def threaded\_client(client):  
 while True:  
 data = pickle.loads(client.recv(2048))  
 # checks to see if any data is being received from the client, if not it assumes that the client is  
 # disconnected and stops running in the background  
 if data:  
 print("connection made")  
  
 else:  
 print("Lost Connection")  
 client.close()  
 break  
  
currentPlayer = 0  
while True:  
 connection, addr = s.accept()  
 print("Connected to:", addr)  
 start\_new\_thread(threaded\_client, (connection, currentPlayer))  
 currentPlayer += 1

This still printed the same text to console (“**Waiting for connection, Server Initiated**”) once the file was ran without any errors meaning it was ready to handle any incoming connections.

**Prototype 2 Development (Spawn Zone and Physics):**

**Prototype 3 Development (Map Generation):**