1) **Python program to generate Powerball numbers (include in canvas submission)**
   (25 points)

2) **Screenshot of 10 weeks of output**
   (25 points)

```
❯ python3 Powerball Jacob Boicken.py
[50, 54, 6, 34, 66],     [16]
[18, 9, 33, 16, 64],     [25]
[8, 12, 11, 47, 22],     [24]
[31, 17, 48, 61, 9],     [20]
[31, 39, 14, 51, 62],    [5]
[33, 46, 68, 4, 60],     [25]
[11, 63, 34, 5, 1],      [5]
[42, 20, 51, 7, 10],     [18]
[30, 48, 49, 17, 25],    [23]
[60, 48, 35, 18, 24],    [22]
```

3) **Explanation of how to rig the Lottery**
   (25 points)

   We could set the seed of the RNG for a specific drawing. This seed is simply the position in the pseudo random sequence. This will then generate the same sequence of numbers so. So we set the seed for one day and then pick the numbers that it generates.

   If we only know the only seed being used, we could then determine where the lotto is in its sequence. From there, we can determine the future drawings' numbers.

4) **Explanation of how to increase the randomness of the simulated Lottery**
   (25 points)

   I could base the seed off of the current time (since epoch). This prevents the same seed being reused and starts at a different point in the sequence each time the powerball is run.

   I could generate random numbers from using /dev/urandom or /dev/random. This would use the OS to generate random numbers. These both are affected by the entropy from noise in the OS and drivers. Which should make it difficult to find a pattern and predict future numbers.

   (I could also get a Raspberry Pi, a radiation detector, and some uranium or radon to create a random number generator.)