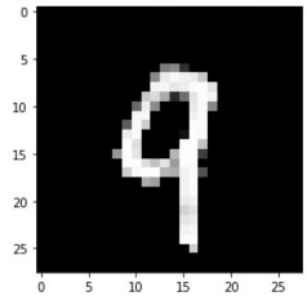
I trained a GAN using the given sample code. Then, I found two latent va riables z1 and z2, which the values and the images produced are shown below.

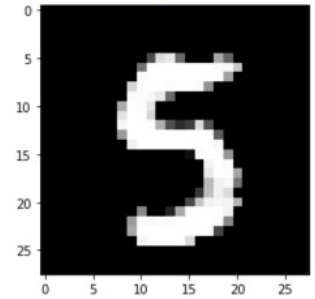
-> z1 (produced digit 9)

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
tensor([[ 1.3319, -1.7939,  0.6510, -0.0456,  0.9653,  0.4267,  0.6034,  0.3739,
          0.0371,  1.3116, -1.3073,  0.3166,  0.1823, -0.9793, -0.8949,  0.0887,
          0.6029,  1.6431,  0.9497, -0.0833, -0.6594, -0.2574,  0.6136,  0.4099,
          0.3209,  0.8264,  1.0338, -2.1359,  0.8622, -0.9737,  1.2565,  1.2629,
          0.8214, -0.0418, -0.5096,  1.6801, -0.8237,  0.0179,  0.6988, -0.2461,
          1.4188,  1.0375, -0.0650, -0.1608, -1.4040,  0.0918, -0.6466,  1.7992,
         -0.6746,  0.7195,  0.0028,  0.8255,  0.6061, -0.2376,  0.6338,  0.7979,
          0.8505,  0.8977, -1.8142, -1.6561,  0.0558, -0.3086, -0.6803, -0.2631,
          0.1198,  0.0874,  0.4023, -1.2749,  0.5639,  0.6952,  0.4274, -2.4245,
         -0.2063, -0.5090,  0.3182,  0.1857, -0.3777,  0.7154, -0.3786,  1.0571,
          1.8019, -0.8122, -0.3706,  0.7700,  0.1122, -0.0270, -0.5246, -1.2239,
          0.1856,  1.5425,  0.3172,  0.9636, -0.2757, -1.4367, -0.4300, -0.0153,
          0.5537, -1.3731, -0.3424, -0.1709]], device='cuda:0')
<matplotlib.image.AxesImage at 0x7fd9021d5490>
```

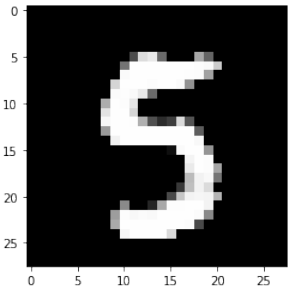
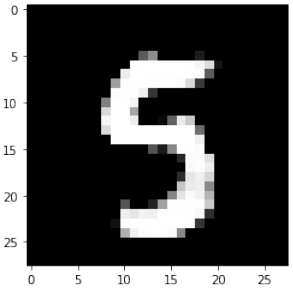
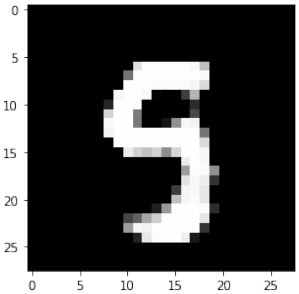
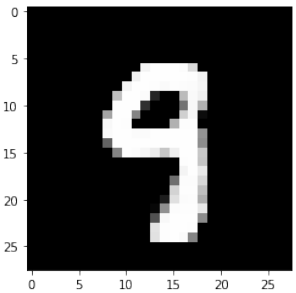
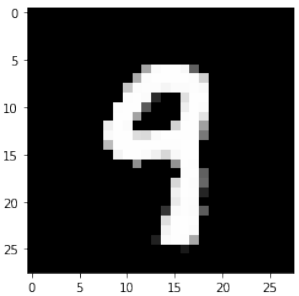
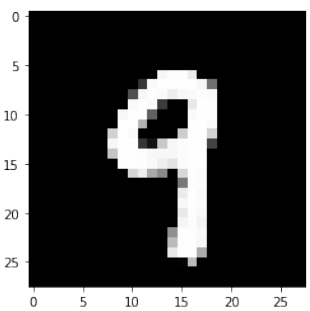


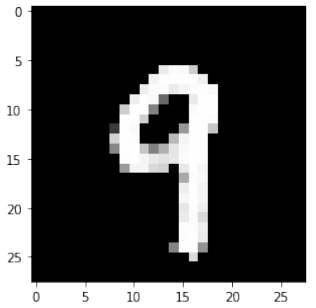
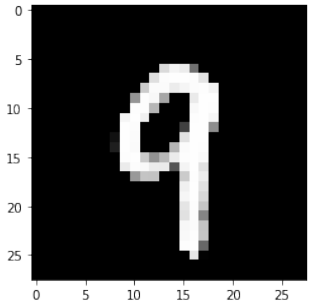
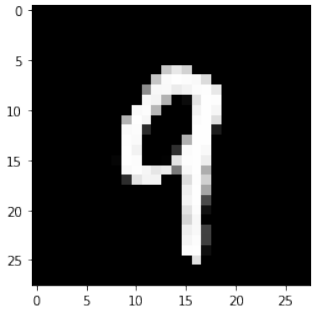
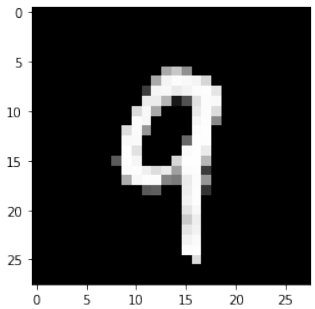
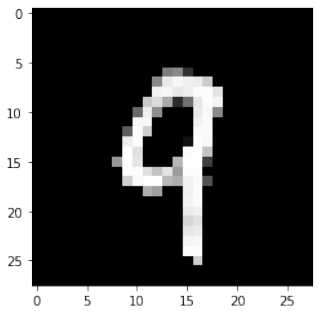
-> z2 (produced digit 5)

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
tensor([[-1.7870,  0.5879, -1.8414,  1.8812,  0.7957, -2.7532, -0.0780,  0.6262,
         -0.6961,  0.9540, -2.2150,  0.4719, -0.8507,  0.0186,  0.3428,  1.2143,
          0.5695, -0.1708, -0.6461, -0.3865,  0.4549, -2.9653,  0.2762,  0.2606,
          1.6394,  0.5637, -1.5359, -0.2431,  1.0673,  0.0387,  1.0756,  0.8218,
          1.5111,  0.6655,  2.3562, -0.9202,  0.3663,  0.4345,  0.3970, -0.8745,
          0.0466,  0.0744, -0.7687,  0.0378, -0.7770, -0.4122, -0.2270, -0.9892,
          0.6165,  0.1166,  0.5393, -0.6477, -1.4745, -0.5440, -0.0752, -2.1309,
         -0.5068, -0.9801,  1.0667,  0.5967,  1.0544,  0.8656,  1.1007, -2.7266,
         -0.4097, -0.7404, -1.0311,  1.3437,  0.4624, -0.6355, -0.7126, -0.1721,
         -1.0749, -0.0557,  0.2636,  0.6530,  0.2845,  1.5408, -2.1918, -0.7971,
         -0.2013, -0.2250, -0.9191, -1.7265, -1.2418, -0.3379,  0.2381, -0.6938,
         -0.6656,  0.1905, -0.5309, -1.5759,  0.9528,  0.2214,  0.5346, -2.1369,
         -0.2121,  0.3699, -1.3213, -0.1191]], device='cuda:0')
<matplotlib.image.AxesImage at 0x7fd8ed670990>
```



Then, I generated intermediate images of the two digits by specifying z's interpolating z1 and z2 (z = a z1 + (1–a) z2 where a = 0, 0.1, …, 1.0). The generated images are shown below (ordered from a = 0 to a = 1.0)

As can be seen, the generated image tends to look like 5 when a is betw
een 0 to 0.2 and look like 9 when a is between 0.3 to 1.0. It should come
intuitively that the higher the weight of the image, the more similar it is to
the produced image. This is proved as it can be seen that if a is lower (th
e weight of z2 (digit 5) is higher), the intermediate image looks more simi

lar to 5, while if a is higher (the weight of z1 (digit 9) is higher), it looks m ore similar to 9. So by modifying the a, we can create an image which co mbines the characteristics of both digits and adjust how significant each characteristic is for the produced image. We can also modify the method to produce an image which is based on 3 or more digits.