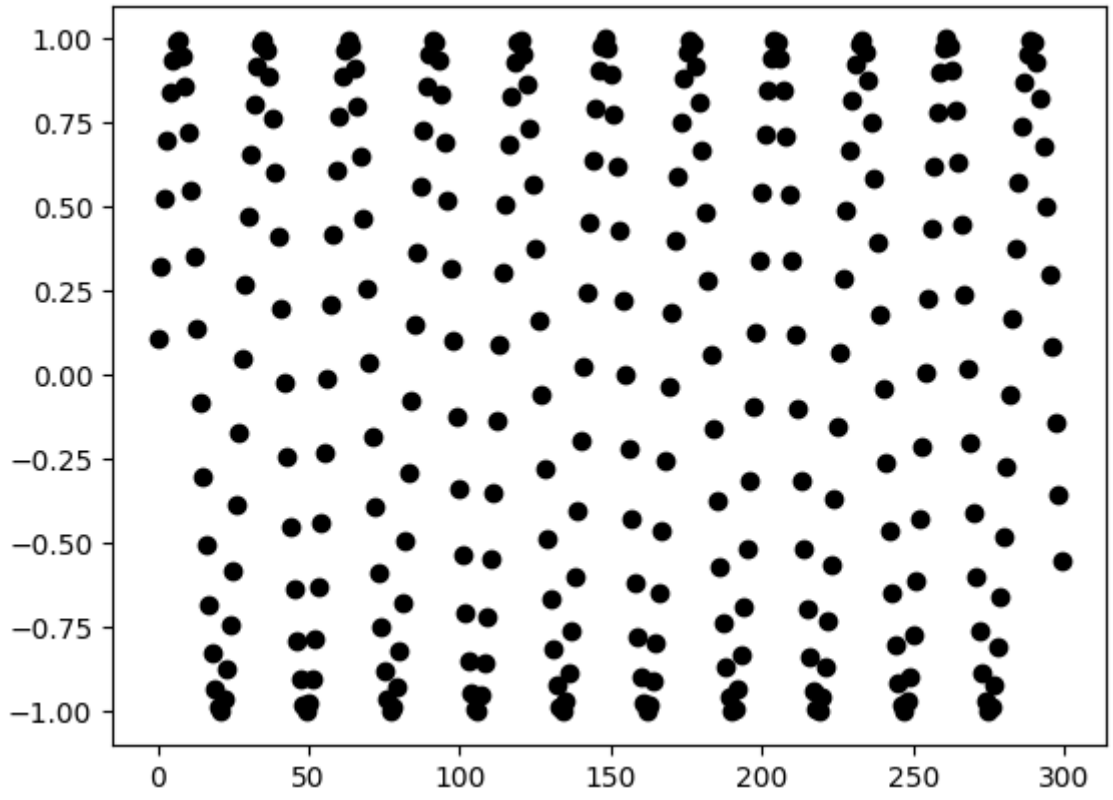
In [2]:

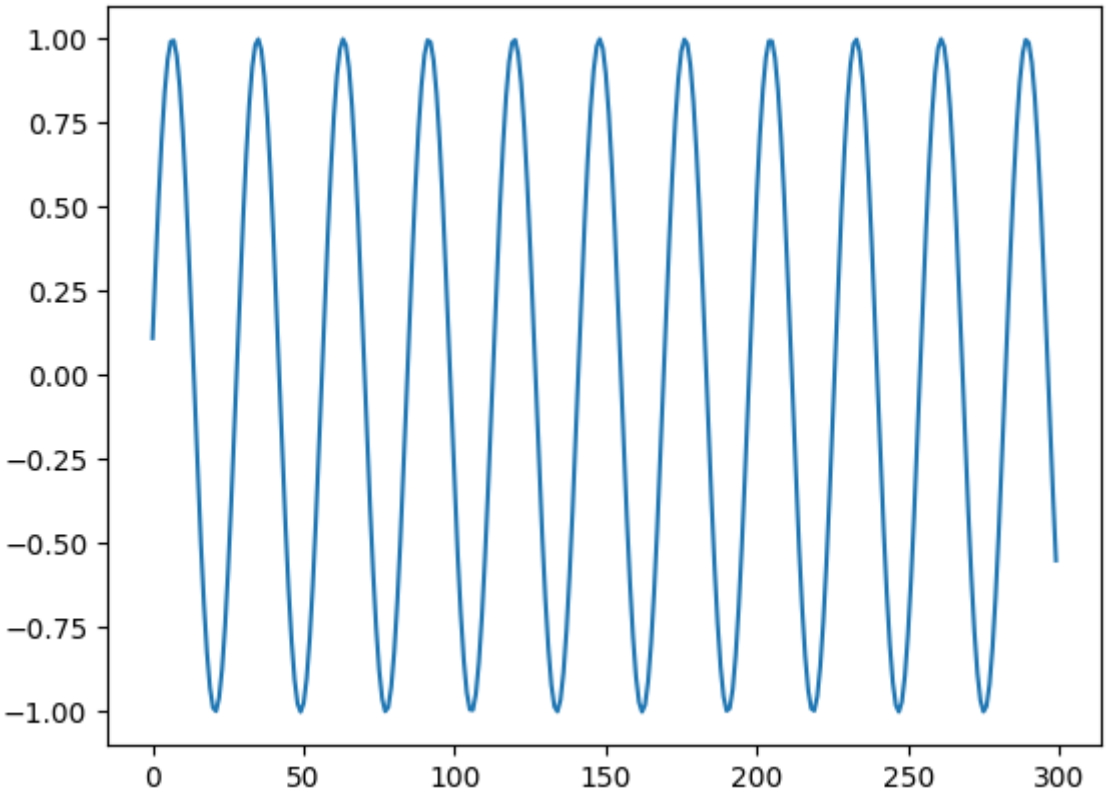
```
1 import numpy as np
2 import math
3 import matplotlib.pyplot as plt
```

In [3]:
```
%matplotlib inline

theta = 1 / 9  # angle in radians
t = np.arange(0, 300, 1)  # (start, stop, step)
y = np.sin((2 * t + 1) * theta)

print("Scatter plot")
plt.plot(t, y, "o", color="black")
plt.show()

print("Linear interpolation")
plt.plot(t, y)
plt.show()
```

```
Scatter plot
```




```
Linear interpolation
```



In [4]:
```
for n in range(1, 20):
    N = 2**n
    theta = np.arcsin(np.sqrt(1 / N))
    t = math.floor(np.pi / (4 * theta))
    p = np.sin((2 * t + 1) * theta) ** 2
    print("%d\t %12.10f" % (N, p))
```

```
2        0.5000000000
4        1.0000000000
8        0.9453125000
16       0.9613189697
32       0.9991823155
64       0.9965856808
128      0.9956198657
256      0.9999470421
512      0.9994480262
1024     0.9994612447
2048     0.9999968478
4096     0.9999453461
8192     0.9999157752
16384    0.9999997811
32768    0.9999868295
65536    0.9999882596
131072   0.9999992587
262144   0.9999978382
524288   0.9999997279
```

In [5]:

```
# We'll start the loop with n=2 to stay within the domain of arcs
# (Note that we can't have 4 solutions when n=1.)

for n in range(2, 20):
    N = 2**n
    old_theta = np.arcsin(np.sqrt(1 / N))
    new_theta = np.arcsin(np.sqrt(4 / N))
    t = math.floor(np.pi / (4 * old_theta))
    p = np.sin((2 * t + 1) * new_theta) ** 2
    print("%d\t %12.10f" % (N, p))
```

```
4         1.0000000000
8         0.5000000000
16        0.2500000000
32        0.0122070313
64        0.0203807689
128       0.0144530758
256       0.0000705058
512       0.0019310741
1024      0.0023009083
2048      0.0000077506
4096      0.0002301502
8192      0.0003439882
16384     0.0000007053
32768     0.0000533810
65536     0.0000472907
131072    0.0000030066
262144    0.0000086824
524288    0.0000010820
```

In [6]:
```
s = 7  # Number of solutions. This can be any positive integer.

# The loop starts with the smallest value of n that allows s solu
for n in range(math.ceil(math.log2(s)), math.ceil(math.log2(s)) +
    N = 2**n
    theta = np.arcsin(np.sqrt(s / N))
    t = math.floor(np.pi / (4 * theta))
    p = np.sin((2 * t + 1) * theta) ** 2
    print("%d\t %12.10f" % (N, np.sin((2 * t + 1) * theta) ** 2))
```

```
8        0.8750000000
16       0.6835937500
32       0.9877929688
64       0.9869401455
128      0.9933758959
256      0.9942813445
512      0.9977678832
1024     0.9999963373
2048     0.9999257666
4096     0.9983374778
8192     0.9995465664
16384    0.9995822234
32768    0.9999531497
65536    0.9998961946
131072   0.9999998224
262144   0.9999745784
524288   0.9999894829
1048576  0.9999939313
2097152  0.9999979874
4194304  0.9999986243
```

In [ ]:
```
1
```