In [1]:
```
import matplotlib.pyplot as plt
import numpy as np
import random
import time
def num(p,n_ill,N):
    roll = np.random.rand(N-n_ill)
    return np.sum(roll<(1-(1-p)**n_ill))
```

In [3]:
```
%%time
num(0.25, 250,1000)
```

```
Wall time: 0 ns
```

Out[3]: 750

In [7]:
```
def avg_num(iterations, p, n_ill,N):
    k=0
    for i in range(iterations):
        k+=num(p, n_ill,N)
    return k/iterations
```

In [8]:
```
avg_num(20, 0.5,1,1000)
```

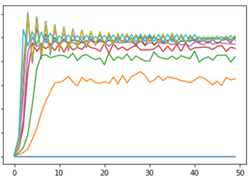
Out[8]: 497.3

In [32]:
```
def graph(iterations, n_ill,days,N, step=10):
    amount=np.empty(step)
    p=np.linspace(0.001,0.01,step)
    for i in range(step):
        avg_vect=np.ones((iterations, days))*n_ill
        for j in range(days):
            for k in range(iterations):
                avg_vect[k, j]=round(num(p[i],int(avg_vect[k, j-1]),N))
        plt.plot(range(days), np.mean(avg_vect, axis=0), label='p='+str(p[i]) )
```

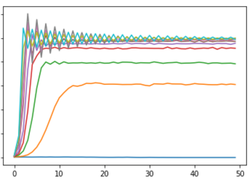
In [33]:
```
%%time

graph(1,1,50,1000)
```

```
Wall time: 35.9 ms
```

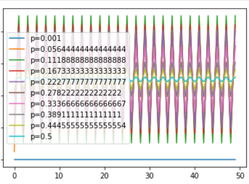


In [34]:
```
graph(20,1,50,1000)
```

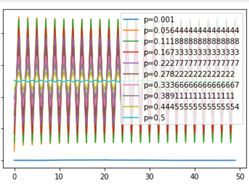


In [44]:
```
def graph(iterations, n_ill,days,N, step=10):
    amount=np.empty(step)
    p=np.linspace(0.001,0.5,step)
    for i in range(step):
        avg_vect=np.ones((iterations, days))*n_ill
        for j in range(days):
            for k in range(iterations):
                avg_vect[k, j]=round(num(p[i],int(avg_vect[k, j-1]),N))
        plt.plot(range(days), np.mean(avg_vect, axis=0), label='p='+str(p[i]) )
    plt.legend()
```

In [45]:
```
graph(1,1,50,1000)
```



In [46]:
```
graph(20,1,50,1000)
```



In [38]:
```
def limited_num(iterations, n_ill, days,N):
    step = 100
    p_c=[]
    pos=0
    amount=np.empty(step)
    p=np.linspace(0,0.004,step)
    for i in range(step):
        avg_vect=np.ones((iterations, days))*n_ill
        for j in range(days):
            for k in range(iterations):
                avg_vect[k, j]=round(num(p[i],int(avg_vect[k, j-1]),N))
        amount[i]= np.mean(avg_vect, axis=0)[-1]
    for l in range(step):
        if amount[l]==0:
            pos+=1
    pos-=1
    return p[pos]
#     plt.plot(p, amount)
```

In [39]:
```
%%time
limited_num(10, 1, 100, 1000)
```

```
Wall time: 2.15 s
```

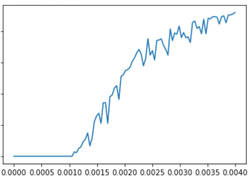
Out[39]: 0.001010101010101

In [40]:
```
def limited_avg_num(iterations, n_ill, days,N):
    step = 100

    amount=np.empty(step)
    p=np.linspace(0,0.004,step)
    for i in range(step):
        avg_vect=np.ones((iterations, days))*n_ill
        for j in range(days):
            for k in range(iterations):
                avg_vect[k, j]=round(num(p[i],int(avg_vect[k, j-1]),N))
        amount[i]= np.mean(avg_vect, axis=0)[-1]
    amount = amount/N
    plt.plot(p, amount)
```

In [41]:
```
%%time
limited_avg_num(20, 1, 100, 1000)
```

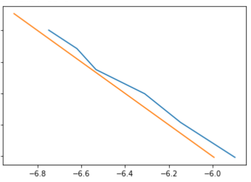
```
Wall time: 4.15 s
```



In [42]:
```
def last():
    N=np.linspace(400,1000,100)
    p_c=1-np.exp(-1/N)
    plt.plot(np.log(p_c), np.log(N))
```

In [17]:
```
%%time
p_c=[]
kol=range(400,1000,100)
for t in kol:
    p_c.append(limited_num(10, 1, 100, t))
for g in kol:
    g=1/g
plt.plot(np.log(p_c), np.log(kol))
last()
```

```
Wall time: 11 s
```



In [ ]:
```

```