In [1]:
```
import numpy as np
import pandas
import math
import seaborn as sns


np.set_printoptions(threshold=np.nan)

total = []
total.append(float(2))
total.append(float(1.1**10.0))
n_list = []
n_list.append(1)
n_list.append(10)
n = 100
i = 1

# print('n = 1')
# print('total = %.12f' % float(total[0]))
# print('n = 10')
# print('total = %.12f' % float(total[1]))


while abs(total[i] - total[i-1]) > 0.000000000001:
    total.append(float((1+1/n)**n))
    n_list.append(n)
#     print('n = %d' % n)
#     print('total = %.12f' % total[i+1])
    i = i+1
    n = n*10



vals = [round(i, 12) for i in total]

for n, val in zip(n_list, vals):
    print(n, val)



n = n/10
print('final values:')
```

```
1 2.0
10 2.5937424601
100 2.704813829422
1000 2.716923932236
10000 2.718145926825
100000 2.718268237192
1000000 2.718280469096
10000000 2.718281694132
100000000 2.718281798347
1000000000 2.718282052012
10000000000 2.718282053235
100000000000 2.718282053357
1000000000000 2.718523496037
10000000000000 2.716110034087
100000000000000 2.716110034087
final values:
n = 10000000000000
total = 2.716110034087
The code converges to 2.716110034087 instead of 2.71828183 because it eventually looses too many bits to be
accurate.
```

In [ ]: