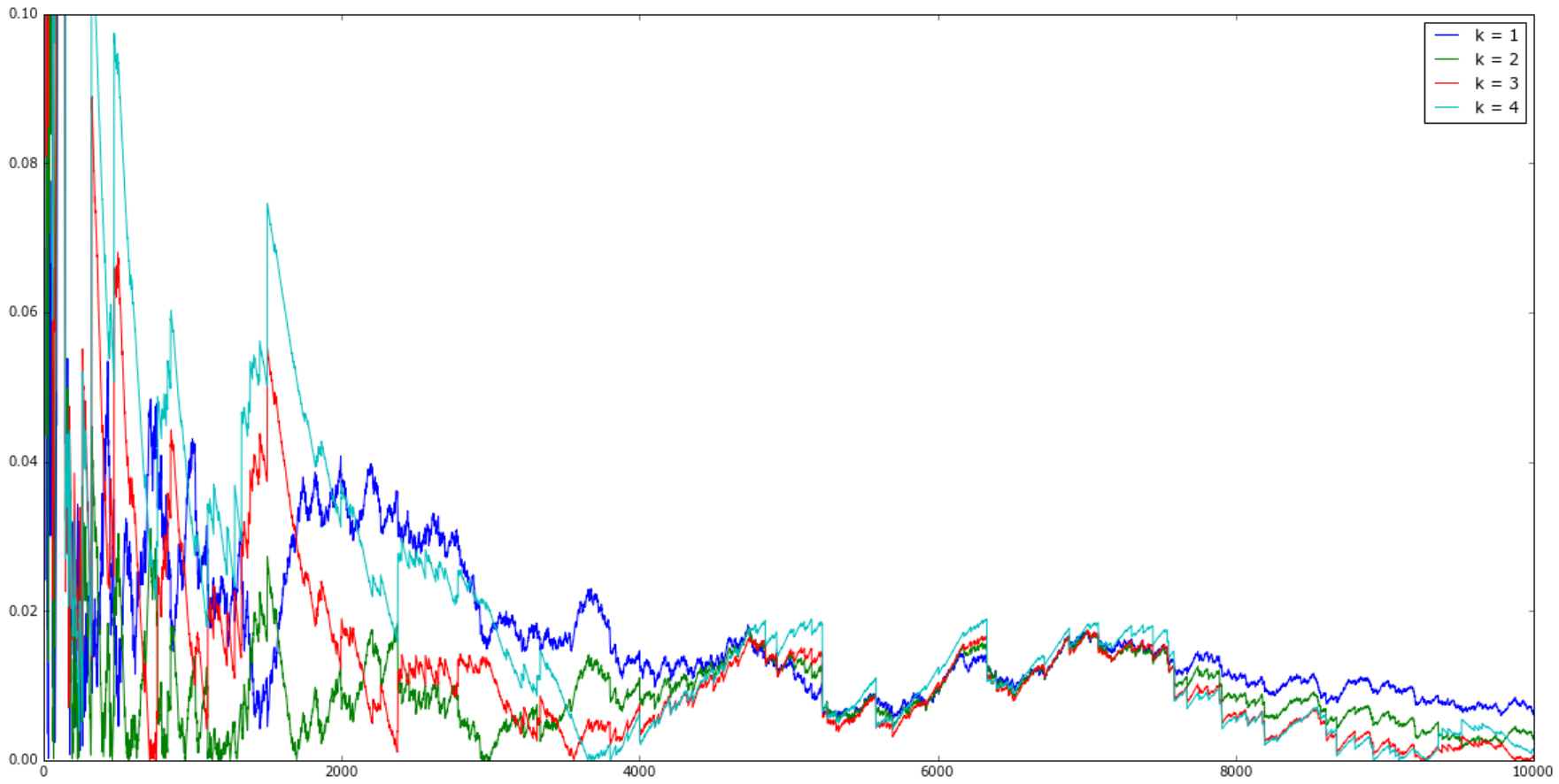
In [1]:
```
import numpy as np
import pylab
import math
import matplotlib.pyplot as plt
from scipy.stats import expon
%matplotlib inline
pylab.rc('font',**{'family':'verdana'})  # Поддержка русскоязычных надписей
```

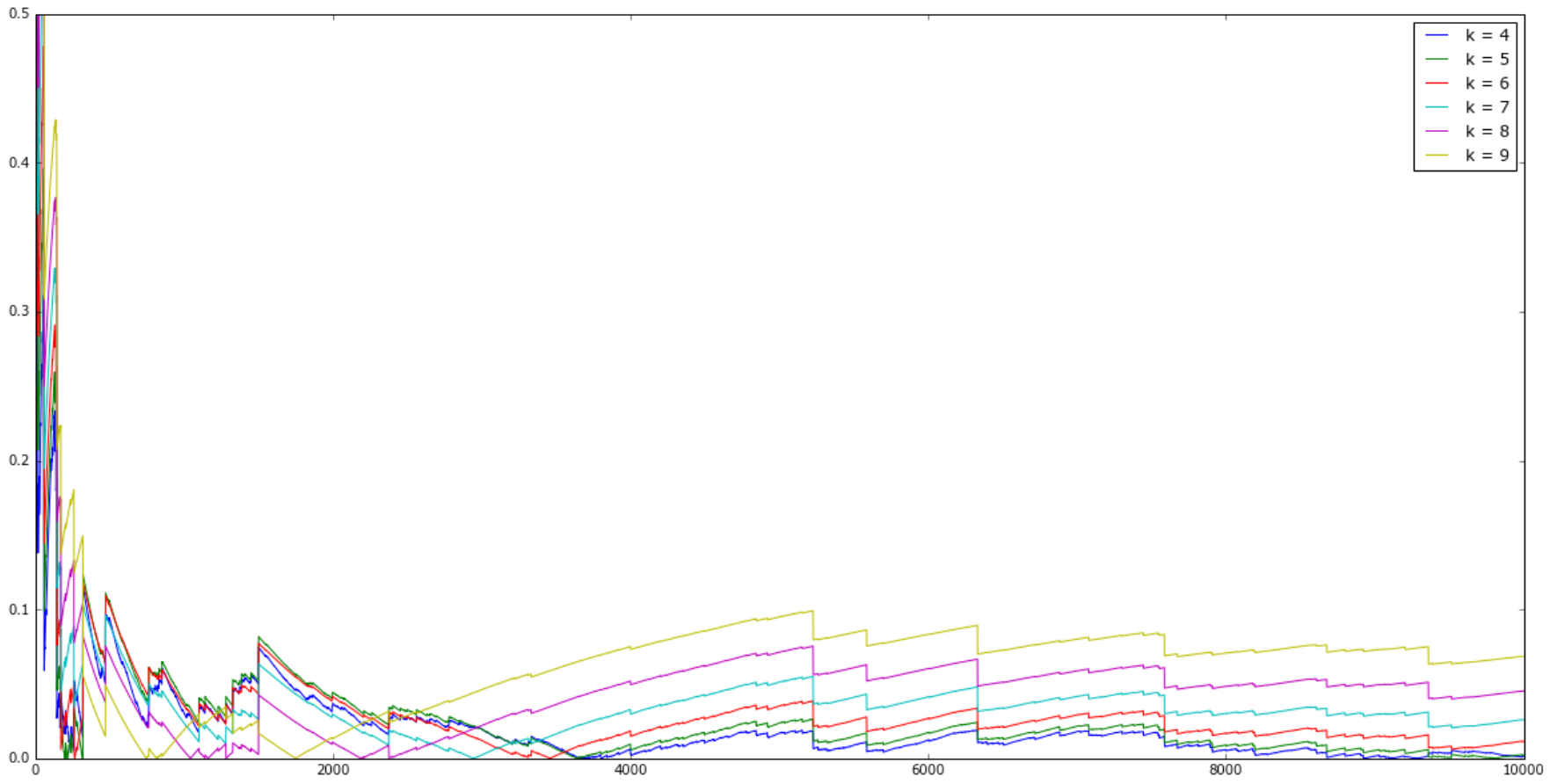
In [4]:
```
# Генерируем выборку из экспоненциального распределения с параметром 1.
maxSize = 10000
sample = expon.rvs(scale=1., size=maxSize)
```

In [5]:
```
def showEstimation(k) :
    estimation = np.array([])
    grid = np.array([]) # grid - список 1..N

    factorial = math.factorial(k)
    # Для каждого n <= N считаем значение всех оценок на выборке x[0]..x[n]
    for sampleSize in range(1, maxSize, 1):
        # Считаем оценку.
        mometK = list(map(lambda x : x**k, sample[:sampleSize]))
        value = (factorial / np.mean(mometK)) ** (1./k)
        grid = np.append(grid, sampleSize)
        estimation = np.append(estimation, abs(value - 1))

    # Строим график зависимости оценки от n.
    plt.plot(grid, estimation, label="k = " + str(k))
```

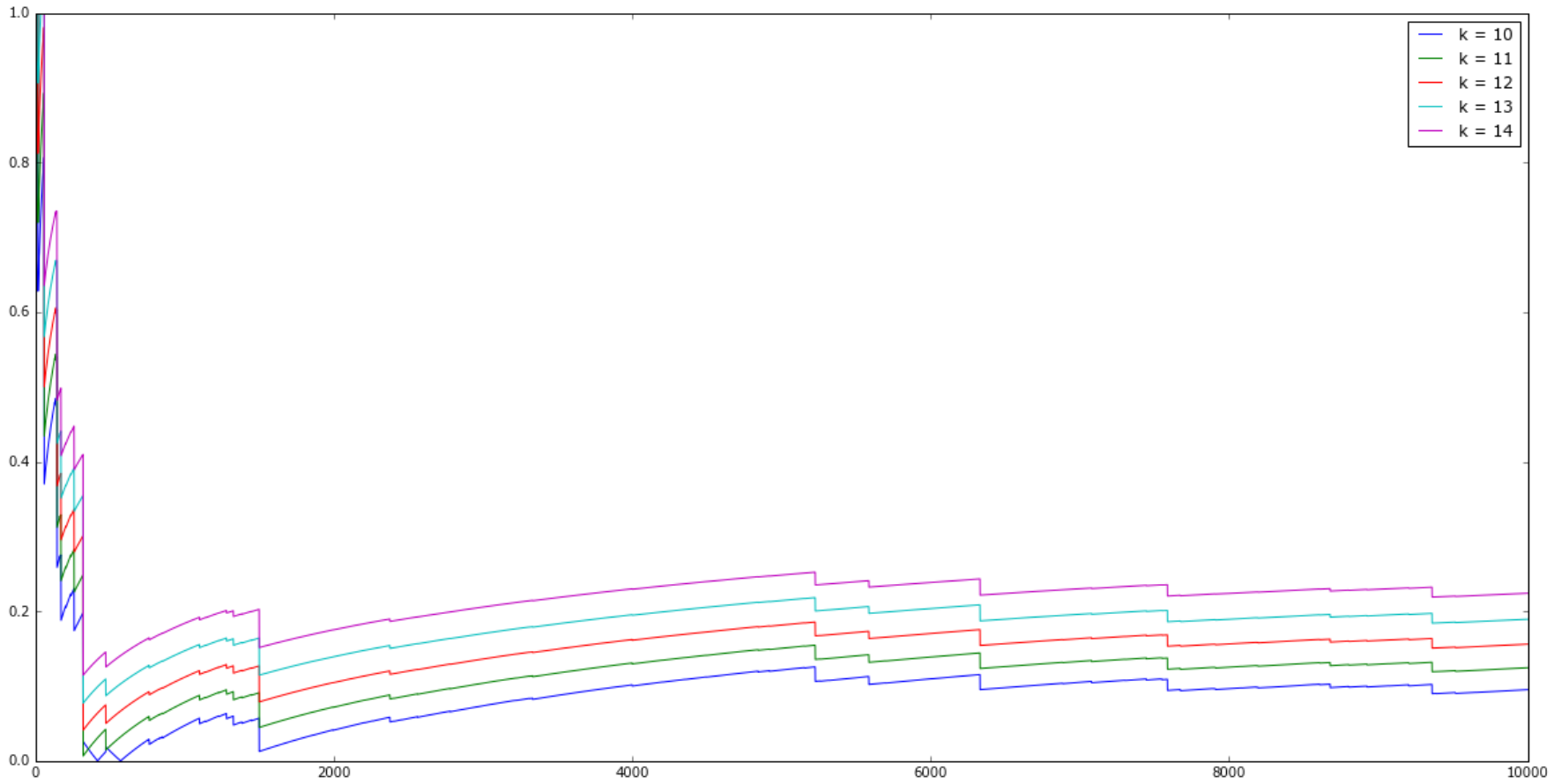
In [6]:
```
# Строим график для значений k от 1 до 4.
plt.figure(figsize=(20, 10))
for k in range(1, 5, 1) :
    showEstimation(k)
plt.ylim(0, 0.1)
plt.legend()
plt.show()
```



In [7]:
```
# Строим график для значений k от 4 до 9.
plt.figure(figsize=(20, 10))
for k in range(4, 10, 1) :
    showEstimation(k)
plt.ylim(0, 0.5)
plt.legend()
plt.show()
```



In [8]:
```
# Строим график для значений k от 10 до 14.
plt.figure(figsize=(20, 10))
for k in range(10, 15, 1) :
    showEstimation(k)
plt.ylim(0, 1)
plt.legend()
plt.show()
```



Вывод:

При k > 5 параметр приближается оценкой значительно хуже, чем при меньших k.
Наилучшие результаты получились при k = 3.