Flipped Coin assessments

this notebook is to assess the maximum log-likelihood of a simple flliped coin game

given 2 possible results, H, T p is the probability to get H The coin was flipped N times k times out of N, it turned out to be H (hense, N-k times T) so the probability to get drill this specific series of fllips is: (p^k)*((1-p)^(N-k))

In [15]:
```
import matplotlib.pyplot as plt
import numpy as np
from numpy import *
```

In [16]:
```
def calculate_probability(p, n, k):
    return ((p**k)*((1-p)**(n-k)))
```

thie function below will calculate the maximum likelihood of p based on k n using this theory: pML = max(ln(p)) = max(ln((p**k)*((1-p)**(n-k))) = max(kln(p)+(N-k)*ln(1-p)) In order to find the max, we will have to derive the expression above: d/dp(pML) and compare it to 0 d/dp{kln(p)+(N-k)ln(1-p)} = k/p - (N-k)/(1-p) =0 k=pN ==> p=k/Ndef calculate_probability(p, n, k): return ((p*k)*((1-p)**(n-k)))

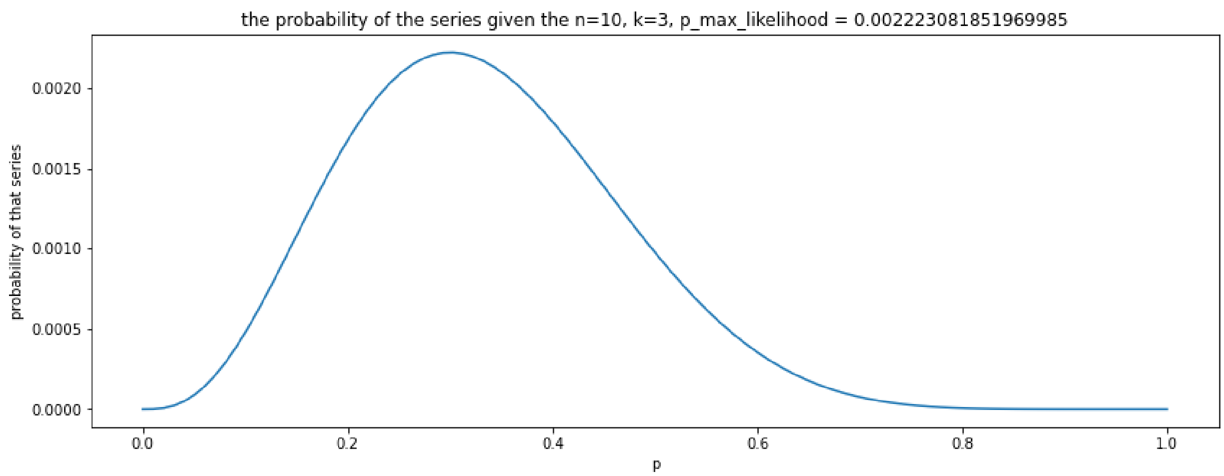
In [17]:
```
def calc_max_of_p(n,k):
    return k/n
```

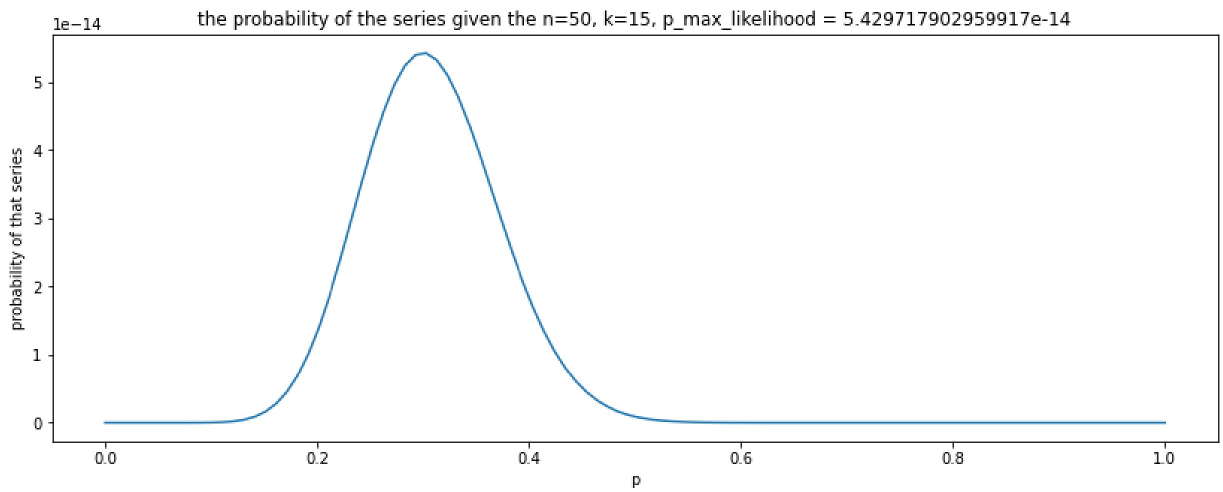
In [18]:
```
def plot_array(arr, n, k):
    plt.figure(figsize=(14,5))
    plt.plot(np.linspace(0,1,100), arr)
    plt.xlabel('p')
    plt.ylabel('probability of that series')
    plt.title(f'the probability of the series given the n={n}, k={k}, p_max_likel
    return
```

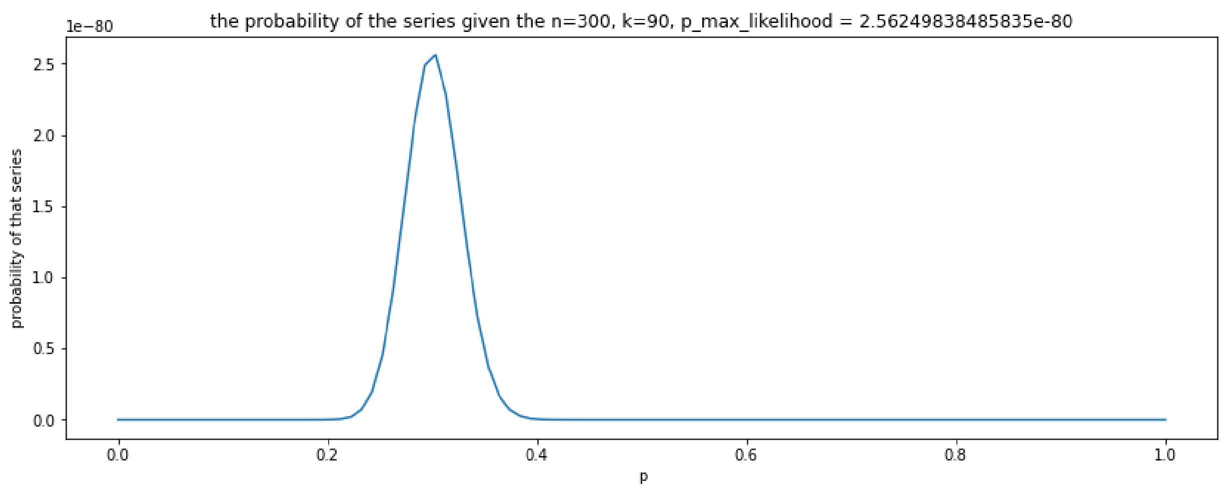
I have ran this game 3 times, each with different inputs. this is to examine the change of the maximum log-likelihood probability for each case.

In [19]:
```
required_examples = [(10,3), (50,15), (300,90)] ## (n,k)

for n,k in required_examples:
    arr = []
    for i in np.linspace(0,1,100):
        arr.append(calculate_probability(i, n, k))
    arr = np.array(arr)
    plot_array(arr, n, k)
```







we can see that the there's a connection between the size of the data (more values) and the inprobability (hence, more likelihood) ==> the bigger the data is - the smaller inprobability