In [14]:

```python
import numpy as np
array1=np.array([[1,2,3],[4,5,6]])
array2=np.array([[7,8,9],[10,11,12]])
print("Array 1:")
print(array1)
print("\n Array 2:")
print(array2)
```

```
Array 1:
[[1 2 3]
 [4 5 6]]

 Array 2:
[[ 7  8  9]
 [10 11 12]]
```

In [15]:

```python
array_addition=array1+array2
print("\n Addition of Array 1 and Array 2:")
print(array_addition)
```

```

 Addition of Array 1 and Array 2:
[[ 8 10 12]
 [14 16 18]]
```

In [16]:

```python
array_subtraction=array1-array2
print("\n Subtaction of array 1 and Array 2:")
print(array_subtraction)
```

```

 Subtaction of array 1 and Array 2:
[[-6 -6 -6]
 [-6 -6 -6]]
```

In [17]:

```python
array_multiplication=array1*array2
print("\n Element-wise multiplication of aray 1 and array 2 :")
print(array_multiplication)
```

```

 Element-wise multiplication of aray 1 and array 2 :
[[ 7 16 27]
 [40 55 72]]
```

In [18]:

```python
array_transpose=np.transpose(array1)
print("\n Transpose of Array 1:")
print(array_transpose)
```

```

 Transpose of Array 1:
[[1 4]
 [2 5]
 [3 6]]
```

In [27]:

```
array_reshaped=array1.reshape(3,2)
print("\nReshaped array 1(from 2x3 to 3x2):")
print(array_reshaped)
```

```

Reshaped array 1(from 2x3 to 3x2):
[[1 2]
 [3 4]
 [5 6]]
```

In [21]:

```
mean_array1=np.mean(array1)
print("\n Mean of array1:")
print(mean_array1)
```

```

 Mean of array1:
3.5
```

In [23]:

```
median_array1=np.median(array1)
print("\n Median of array 1")
print(median_array1)
```

```

 Median of array 1
3.5
```

In [24]:

```
min_value=np.min(array1)
max_value=np.max(array1)
print("\n Min and Max of Array1:")
print("Min:", min_value)
print("Max:",max_value)
```

```

 Min and Max of Array1:
Min: 1
Max: 6
```

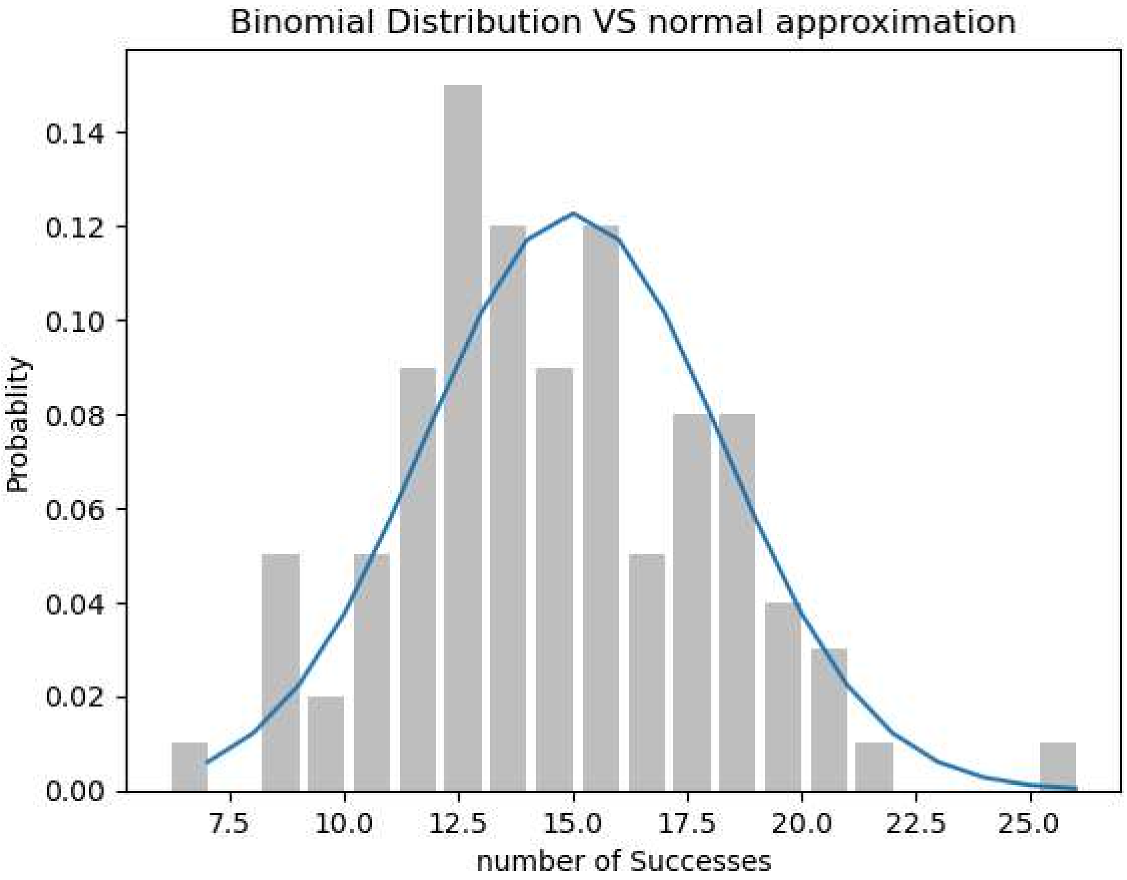
In [26]:

```
std_deviation=np.std(array1)
print("\n Standard Deviation of Array1:")
print(std_deviation)
```

```

 Standard Deviation of Array1:
1.707825127659933
```

In [28]:

```
import random
import math
from collections import Counter
import matplotlib.pyplot as plt

def normal_cdf(x,mu,sigma):
    return(1+math.erf((x-mu)/(sigma*math.sqrt(2))))/2

def bernoulli_trial(p):
    return 1 if random.random()<p else 0

def binomial(n,p):
    return sum(bernoulli_trial(p) for _ in range(n))

def make_hist(p,n,num_points):
    data=[binomial(n,p) for _ in range(num_points)]
    histogram=Counter(data)
    plt.bar([x-0.4 for x in histogram.keys()],[v/num_points for v in histogram.
    mu=p*n
    sigma=math.sqrt(n*p*(1-p))
    xs=range(min(data),max(data)+1)
    ys=[normal_cdf(i+0.5,mu,sigma)-normal_cdf(i-0.5,mu,sigma)for i in xs]
    plt.plot(xs,ys)

    plt.title("Binomial Distribution VS normal approximation")
    plt.xlabel("number of Successes")
    plt.ylabel("Probablity")
    plt.show()

make_hist(0.3,50,100)
```


Binomial Distribution VS normal approximation
Probablity
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
number of Successes

```
In [ ]:
```