In [6]:

```
p_b_a=float(input('Enter probability that person has a disease(after he took a test):')

p_a=float(input('Enter probabiltiy that the test person have a disease:'))

p_b=p_a*p_b_a+(1-p_a)*(1-p_b_a)

p_a_b1=(p_b_a*p_a)*p_b

print('Probability probabiltiy that the test gave a true result:',p_a_b1)
```

```
Enter probability that person has a disease(after he took a test):0.5
Enter probabiltiy that the test person have a disease:0.3
Probability probabiltiy that the test gave a true result: 0.075
```

In [15]:

```
print("Bayes theorem to find P[B\A]\n")
sample_space=[1,2,3,4,5,6]
#A is the number of even numbers
#B had  numbers of greater then 4
A=[2,4,6]
B=[5,6]
C=[6]
# C is the intersection of A and B
# We want to find the conditional probability of P[A\B]
P_A= len(A)/len(sample_space)
P_B= len(B)/len(sample_space)
P_C=len(C)/len(sample_space)
# P_M is P[A/B]
P_M=P_C/P_B
print ("p[A\B]: ", P_M)
#P_K is P[B\A]
# using bayes theorem
P_k=(P_M*P_B)/P_A
print("P[B\A]: ", P_k)
```

```
Zbayes theorem to find P[B\A]

p[A\B]:  0.5
P[B\A]:  0.3333333333333333
```

In [19]:

```
import pandas as pd
import numpy as np
df = pd.DataFrame({'gender': np.repeat(np.array(['Male', 'Female']), 150),
                   'sport': np.repeat(np.array(['Baseball', 'Basketball', 'Football',
                                                'Soccer', 'Baseball', 'Basketball',
                                                'Football', 'Soccer']),
                                      (34, 40, 58, 18, 34, 52, 20, 44))})

#produce contingency table to summarize raw data
survey_data = pd.crosstab(index=df['gender'], columns=df['sport'], margins=True)

#view contingency table
print(survey_data)
#calculate probability of being male,given that individual prefers baseball
a=survey_data.iloc[1,0]/survey_data.iloc[2,0]
print("Conditoinal Probability of being male,given that individual prefers baseball: ",
```

```
sport   Baseball  Basketball  Football  Soccer  All
```

```
gender
Female        34          52         20       44  150
Male          34          40         58       18  150
All           68          92         78       62  300
Conditoinal Probability of being male,given that individual prefers baseball:  0.5
```

In [20]:

```
print("finding conditional probability\n")
sample_space=[1,2,3,4,5,6]
#A is the number of even numbers
#B had  numbers of greater then 4
A=[2,4,6]
B=[5,6]
C=[6]
# C is the intersection of A and B
# We want to find the conditional probability of P[A\B]
P_A= len(A)/len(sample_space)
P_B= len(B)/len(sample_space)
P_C=len(C)/len(sample_space)
conditonal_probability=P_C/P_B
print ("conditonal probability: ", conditonal_probability)
```

```
finding conditional probability

conditonal probability:  0.5
```

In [25]:

```
# Binomial probability law
# In the old days,there was a probability of p of success in any attempt to make
# a telephone call.Calculate the prpbability of having n success in k attempts
import math
p=float(input("Enter prpbabiltity:"))
n=int(input("Enter attempts:"))
k=int(input("Enter succeses:"))
q=1-p
comb=math.comb(n,k)
base1=p
exponent1=k
p2=pow(base1,exponent1)
base2=q
exponent2=n-k
q2=pow(base2,exponent2)
result=comb*p2*q2
print("Probabiltiy of ",n,"successes in ",k,"attempts:",result)
```

```
Enter prpbabiltity:0.4
Enter attempts:10
Enter succeses:2
Probabiltiy of  10 successes in  2 attempts: 0.12093235199999998
```

In [28]:

```
def nCr(n, r):
    if (r > n / 2):
        r = n - r
    answer = 1
    for i in range(1, r + 1):
        answer *= (n - r + i)
        answer /= i
    return answer;

def binomialProbability(n, k, p):
```

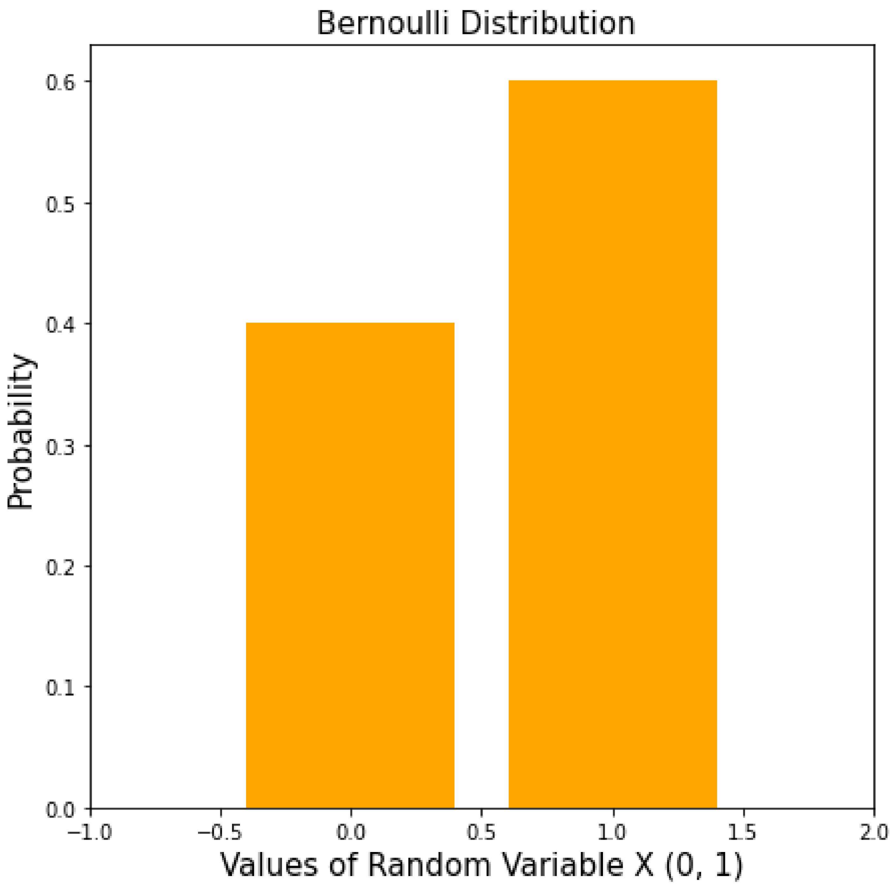
```
    return (nCr(n, k) * pow(p, k) *
                        pow(1 - p, n - k))

p=float(input("Enter prpbabiltity:"))
n=int(input("Enter attempts:"))
k=int(input("Enter succeses:"))

probability = binomialProbability(n, k, p)

print("Probability of", k,
    "heads when a coin is tossed", end = " ")
print(n, "times where probability of each head is",
                                    round(p, 6))
print("is = ", round(probability, 6))
```

```
Enter prpbabiltity:0.4
Enter attempts:10
Enter succeses:2
Probability of 2 heads when a coin is tossed 10 times where probability of each head is
0.4
is =  0.120932
```

In [38]:

```
import matplotlib.pyplot as plt
from scipy.stats import bernoulli
#
# Instance of Bernoulli distribution with parameter p = 0.7
#
p=input("Enter probability:")
p=float(p)
bd = bernoulli(p)
#
# Outcome of experiment can take value as 0, 1
#
X = [0, 1]
#
# Create a bar plot; Note the usage of "pmf" function
# to determine the probability of different values of
# random variable
#
plt.figure(figsize=(7,7))
plt.xlim(-1, 2)
plt.bar(X, bd.pmf(X), color='orange')
plt.title('Bernoulli Distribution ', fontsize='15')
plt.xlabel('Values of Random Variable X (0, 1)', fontsize='15')
plt.ylabel('Probability', fontsize='15')
plt.show()
```

```
Enter probability:0.6
```




In [31]:

```
import numpy as np
#created a bernoulli class

class bernoulli():
    def pmf(x,p):
        """
        probability mass function
        """
        f = p**x*(1-p)**(1-x)
        return f

    def mean(p):
        """
        expected value of bernoulli random variable
        """
        return p

    def var(p):
        """
        variance of bernoulli random variable
        """
        return p*(1-p)

    def std(p):
        """
        standart deviation of bernoulli random variable
        """
        return bernoulli.var(p)**(1/2)

    def rvs(p,size=1):
        """
```

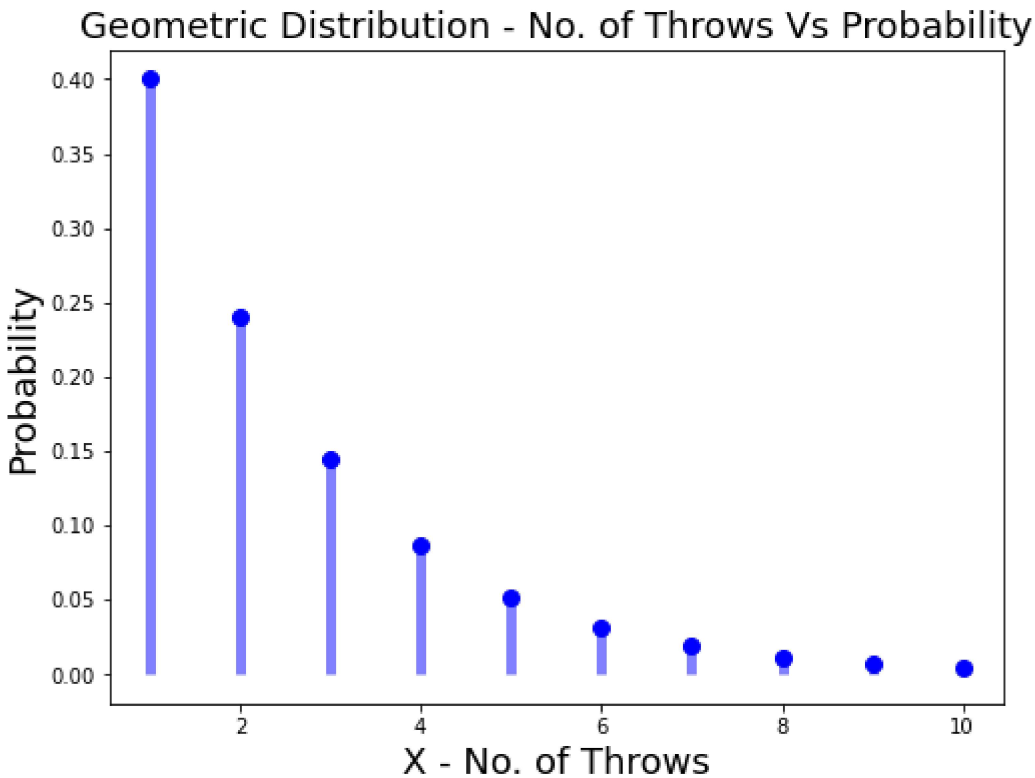
```
        random variates
        """
        rvs = np.array([])
        for i in range(0,size):
            if np.random.rand() <= p:
                a=1
                rvs = np.append(rvs,a)
            else:
                a=0
                rvs = np.append(rvs,a)
        return rvs

p=0.2 # probability of having an accident
print(bernoulli.mean(p)) # return -> 0.2
print(bernoulli.var(p) )# return -> 0.16
print(bernoulli.std(p) )# return -> 0.4
#each execution generates random numbers, so array may be change
print(bernoulli.rvs(p,size=10))
#return-> array([0., 0., 0., 0., 1., 0., 1., 0., 0., 1.])
```

```
0.2
0.16000000000000003
0.4
[0. 1. 1. 0. 0. 0. 0. 0. 1. 0.]
```

In [45]:

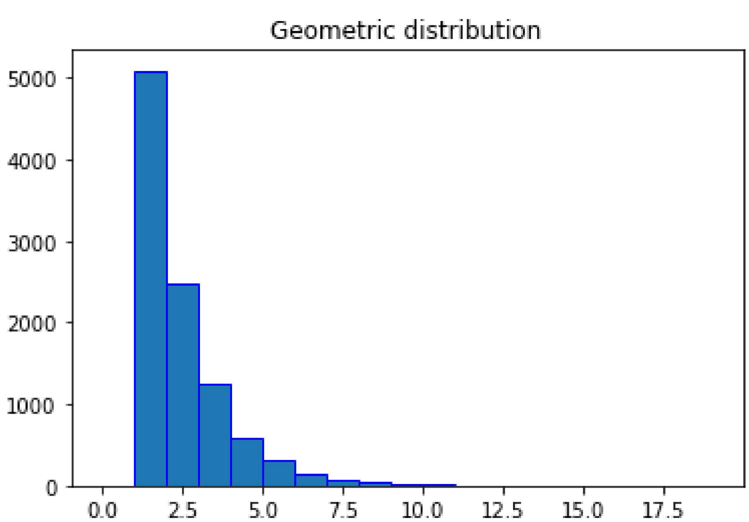
```
from scipy.stats import geom
import matplotlib.pyplot as plt
#
# X = Discrete random variable representing number of throws
# p = Probability of the perfect throw
#
X = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
p = float(input("Enter Probability: "))
#
# Calculate geometric probability distribution
#
geom_pd = geom.pmf(X, p)
#
# Plot the probability distribution
#
fig, ax = plt.subplots(1, 1, figsize=(8, 6))
ax.plot(X, geom_pd, 'bo', ms=8, label='geom pmf')
plt.ylabel("Probability", fontsize="18")
plt.xlabel("X - No. of Throws", fontsize="18")
plt.title("Geometric Distribution - No. of Throws Vs Probability", fontsize="18")
ax.vlines(X, 0, geom_pd, colors='b', lw=5, alpha=0.5)
plt.show()
```

```
Enter Probability: 0.4
```




In [46]:

```
import matplotlib.pyplot as plt
import numpy as np

#fixing the seed for reproducibility
#of the result
np.random.seed(10)

size = 10000
#drawing 10000 sample from
#geometric distribution
sample = np.random.geometric(0.5, size)
bin = np.arange(0,20,1)

plt.hist(sample, bins=bin, edgecolor='blue')
plt.title("Geometric distribution")
plt.show()
```



In [ ]:

```
```