In [2]:

```
#Question 1: Binomial Distribution 1

from math import factorial
def bino(n, x, p):
    return factorial(n) // factorial(n - x) // factorial(x) * (p ** x) * (1 - p) ** (n
- x)


b = map(float, input().split())
g = map(float, input().split())

p = b / (b + g)

r = bino(6, 3, p) + bino(6, 4, p) + bino(6, 5, p) + bino(6, 6, p)

print("%.3f" % r)
```

```
1.09 1
0.696
```

In [3]:

```
#Question 2: Binomial Distribution 2

from math import factorial


def bino(n, x, p):
    return factorial(n) // factorial(n - x) // factorial(x) * (p ** x) * (1 - p) ** (n
- x)

p = map(int, input().split())
n = map(int, input().split())

p = p / 100

r = bino(n, 0, p) + bino(n, 1, p) + bino(n, 2, p)
print("%.3f" % r)

r = sum(bino(n, i, p) for i in range(2, n + 1))
print("%.3f" % r)
```

```
12 10
0.891
0.342
```

In [4]:

```
#Question 3: Normal Distribution I

import math


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


mu = 20
std = 2


print(round(cp(19.5, mu, std), 3))

print(round(cp(22, mu, std) - cp(20, mu, std), 3))
```

```
0.401
0.341
```

In [5]:

```
#Question 4: Normal Distribution II

import math

miu, std = 70, 10
cp = lambda x: 0.5 * (1 + math.erf((x - miu) / (std * math.sqrt(2))))

# percentage of students having grade > q1
print(round((1-cp(80))*100,2))

# percentage of students having grade ≥ q2
print(round((1-cp(60))*100,2))

# percentage of students having grade < q2
print(round((cp(60))*100,2))
```

```
15.87
84.13
15.87
```

In [6]:

```
#Question 5: The Central Limit Theorem I

import math


def cp(x, mean, std):
    return 1/2*(1+math.erf((x-mean) / std / math.sqrt(2)))


mean = 205
std = 15
n = 49
tgt = 9800

msum = n * mean
ssum = std * n**(1/2)

print(round(cp(tgt, msum, ssum), 4))
```

```
0.0098
```

In [7]:

```
#Question 6: The Central Limit Theorem II

import math

def cp(x, mu, si):

    return 1 / 2 * (1 + math.erf((x - mu) / si / math.sqrt(2)))

tckts = int(input())
stdts = int(input())
mu = float(input())
si = float(input())

P = cp(tckts, stdts * mu, math.sqrt(stdts) * si)

print("{:.4f}".format(P))
```

```
250
100
2.4
2.0
0.6915
```

In [8]:

```
#Question 7: The Central Limit Theorem III

import math

samples = float(input())
mean = float(input())
sd = float(input())
interval = float(input())
z = float (input())

e = z * sd / math.sqrt(samples)
print('{:2f}'.format(mean - e))
print('{:2f}'.format(mean + e))
```

```
100
500
80
0.95
1.96
484.320000
515.680000
```

In [9]:

```
#Question 8: Pearson Correlation Coefficient I

n = int(input())
x = list(map(float,input().strip().split()))
y = list(map(float,input().strip().split()))

meanx = sum(x) / n
meany = sum(y) / n

stdx = (sum([(i - meanx)**2 for i in x]) / n)**0.5
stdy = (sum([(i - meany)**2 for i in y]) / n)**0.5


covariance = sum([(x[i] - meanx) * (y[i] -meany) for i in range(n)])

coefficient = covariance / (n * stdx * stdy)

print(round(coefficient,3))
```

```
10
10 9.8 8 7.8 7.7 7 6 5 4 2
200 44 32 24 22 17 15 12 8 4
0.612
```

In [11]:

```
#Question 9: Least Square Regression Line

maths, stats = [],[]

for i in range(5):
    m, s= map(int,input().split())
    maths.append(m)
    stats.append(s)

def b_value(x,y):
    n = len(x)
    xy =[x[i]*y[i] for i in range(n) ]
    x_square = [i**2 for i in x]
    # y_square = [i**2 for i in y]

    b = (n*(sum(xy))-((sum(x)*sum(y))))/float(((n*sum(x_square))-sum(x)**2))
    return b

def ab_values(x,y):
    x_mean = sum(x)/float(len(x))
    y_mean = sum(y)/float(len(y))
    b = b_value(x,y)
    a = y_mean - b*x_mean
    return a,b

a,b = ab_values(maths,stats)
print (a + b*80)
```

```
2 7
0.18 0.89 109.85

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-11-41d979ae0eb3> in <module>
      4 
      5 for i in range(5):
----> 6     m, s= map(int,input().split())
      7     maths.append(m)
      8     stats.append(s)

ValueError: invalid literal for int() with base 10: '0.18'
```

In [13]:

```
#Question 10: Multiple Linear Regression

features, featuresets = map(int, input().split())
a = list()
b = list()
for _ in range(featuresets):
    *tempx, tempy = [1] + list(map(float, input().split()))
    a.append((tempx))
    b.append(tempy)

Xa = np.array(a)
Ya = np.array(b)

B = np.matmul(np.matmul(np.linalg.inv(np.matmul(np.transpose(Xa),Xa)), np.transpose(Xa))
, Ya)

ntests = int(input())
for _ in range(ntests):
    sample = [1] + list(map(float, input().split()))
    samplea = np.array(sample)
    print(np.matmul(samplea,B))
```

```
2 7
0.18 0.89 109.85
1.0 0.26 155.72
0.92 0.11 137.66
0.07 0.37 76.17
0.85 0.16 139.75
0.99 0.41 162.6
0.87 0.47 151.77
4
0.49 0.18
105.21455835106931
0.57 0.83
142.6709513072991
0.56 0.64
132.93605469124682
0.76 0.18
129.70175404502444
```

In [ ]: