Kwong Yuet Michael Fadillah Wong

1a Generate a vector of integers from 1 to 100. Compute PX(x) (using mygeometricpmf(p, x)) for each element in the vector. Use p-values of 0.2, 0.5, and 0.9.

Ans.

```
x = [1:1:100];
p1 = 0.2;
p2 = 0.5;
p3 = 0.9;
%a
ai = mygeometricpmf(p1, x);
ai =
 Columns 1 through 10

    0.2000    0.1600    0.1280    0.1024    0.0819    0.0655    0.0524    0.0419    0.0336    0.0268

  Columns 11 through 20

    0.0215    0.0172    0.0137    0.0110    0.0088    0.0070    0.0056    0.0045    0.0036    0.0029

  Columns 21 through 30

    0.0023    0.0018    0.0015    0.0012    0.0009    0.0008    0.0006    0.0005    0.0004    0.0003

  Columns 31 through 40

    0.0002    0.0002    0.0002    0.0001    0.0001    0.0001    0.0001    0.0001    0.0000    0.0000

  Columns 41 through 50

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 51 through 60

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 61 through 70

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 71 through 80

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 81 through 90
```

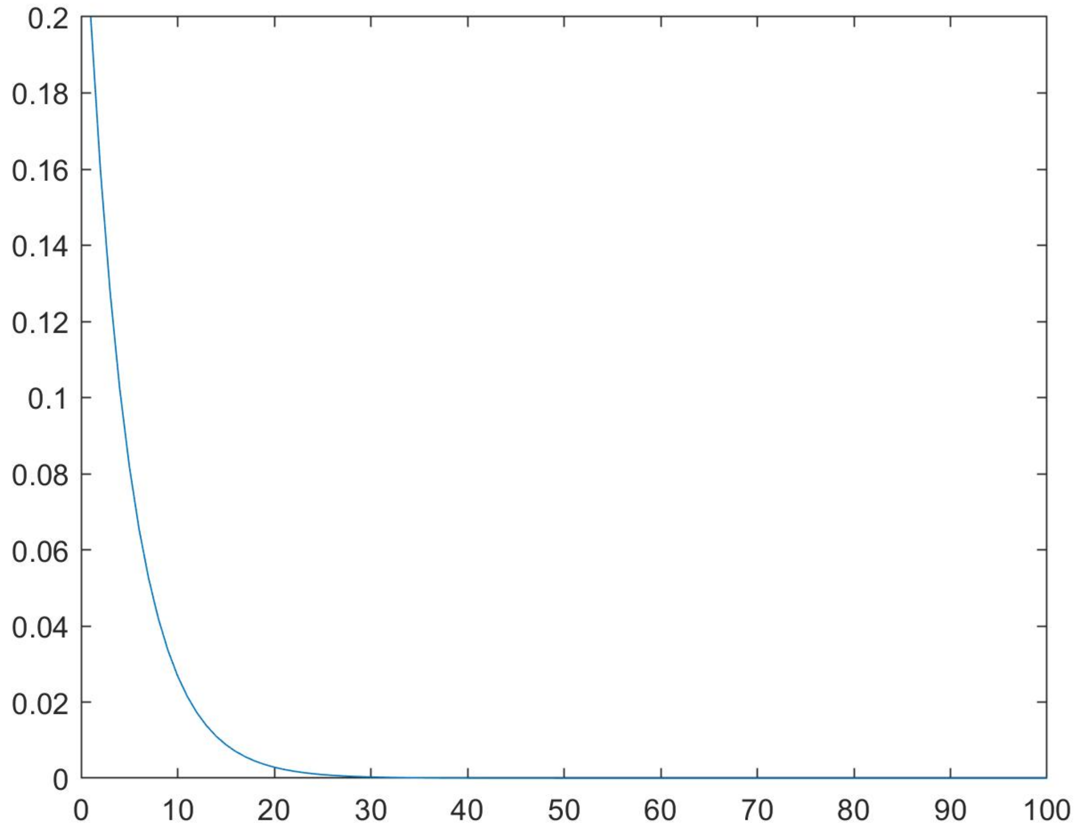
```
  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 91 through 100

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

aii = mygeometricpmf(p2, x);
aii =
Columns 1 through 10

  0.5000    0.2500    0.1250    0.0625    0.0313    0.0156    0.0078    0.0039    0.0020    0.0010

 Columns 11 through 20

  0.0005    0.0002    0.0001    0.0001    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 21 through 30

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 31 through 40

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 41 through 50

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 51 through 60

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 61 through 70

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 71 through 80

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 81 through 90

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

 Columns 91 through 100

  0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000
```

```
aiii = mygeometricpmf(p3, x);
aiii =

  Columns 1 through 10

    0.9000    0.0900    0.0090    0.0009    0.0001    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 11 through 20

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 21 through 30

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 31 through 40

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 41 through 50

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 51 through 60

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 61 through 70

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 71 through 80

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 81 through 90

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

  Columns 91 through 100

    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000    0.0000

function f = mygeometricpmf(p,x)
    f = p.*(1-p).^(x-1);
end
```

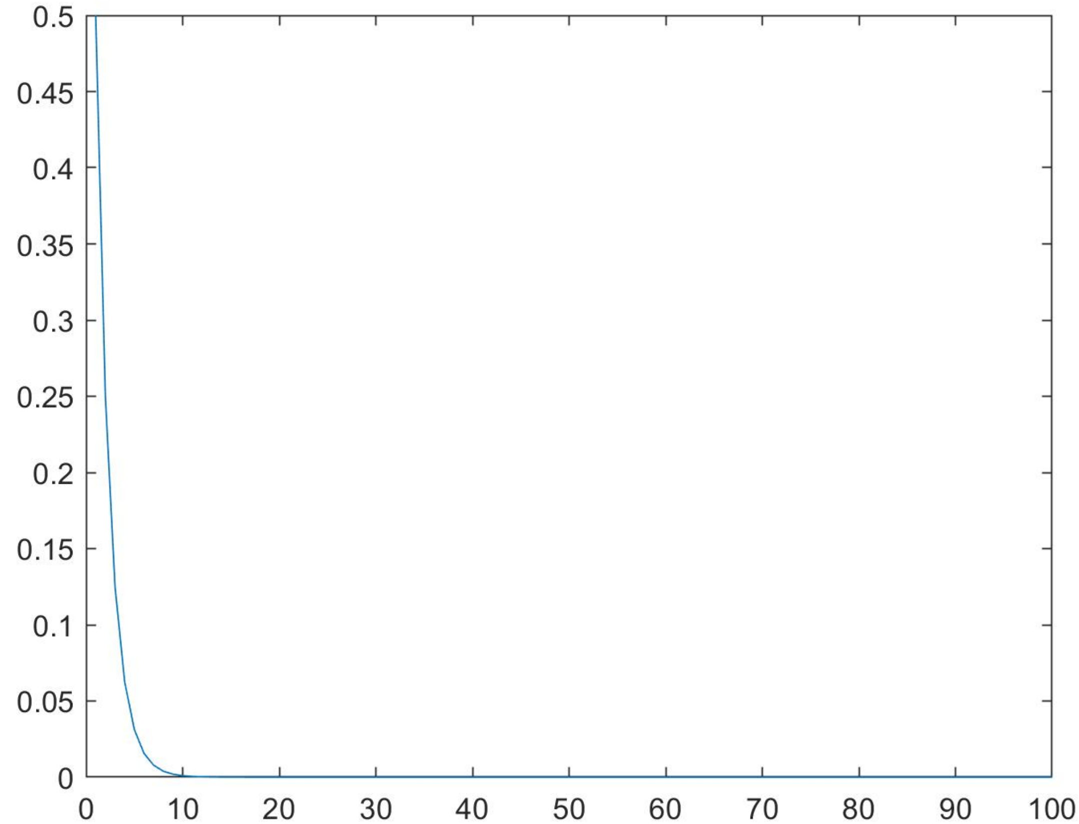
1b. Plot PX(x) for this vector of x. Generate plots for each value of p

Ans.

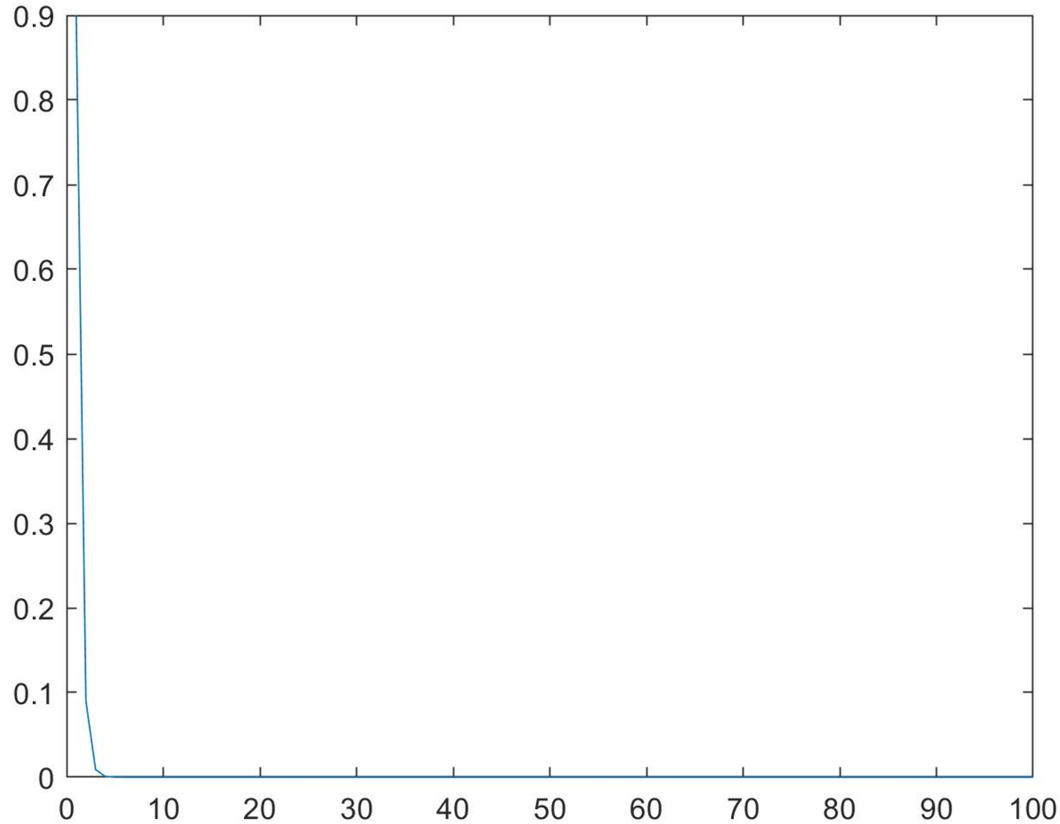
```
%b
plotAi = plot(ai)
```



plotAii = plot(aii)



plotAiii = plot(aiii)



1c. Compute E[X] for this vector of integers using the function that you implemented above. Compute E[X] for each value of p

Ans.

eAi = sum(x .* ai)
eAi = 5
eAii = sum(x .* aii)
eAii = 2
eAiii = sum(x .* aiii)
eAiii = 1.1111

1d. Compute V ar[X] for this vector of integers using the function that you implemented above. Compute V ar[X] for each value of p

Ans.

varianceAi = (1-p1)/(p1 ^ 2)
varianceAi = 20

varianceAii = (1 - p2)/ (p2 ^ 2)
varianceAii = 2
varianceAiii = (1 - p3)/ (p3 ^ 2)
varianceAiii = 0.1235
2a. Generate 1000 samples with the following values for (μ,σ): (70, 10), (150, 20), (-20, 70)

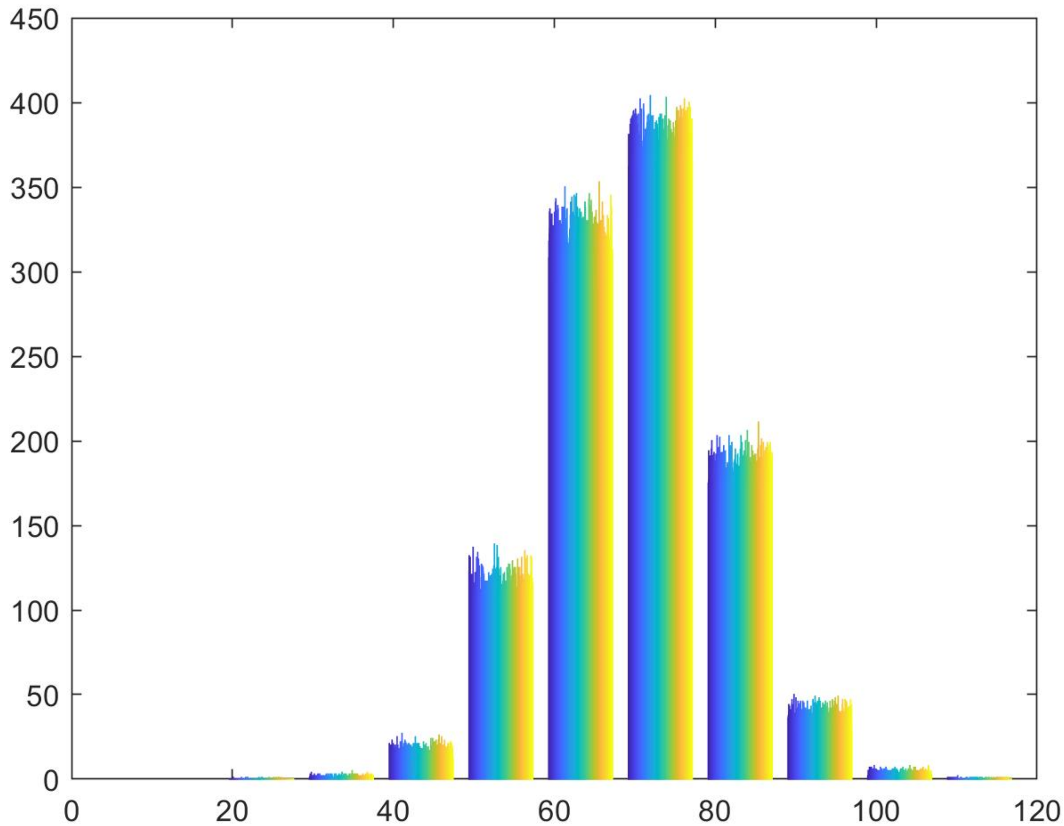
Ans.

```
m = 1000;
ui = 70;
oi = 10;
uii = 150;
oii = 20;
uiii = -20;
oiii = 70;
bi = mygaussiansamples(ui, oi, m)
bii = mygaussiansamples(uii, oii, m)
biii = mygaussiansamples(uiii, oiii, m
```

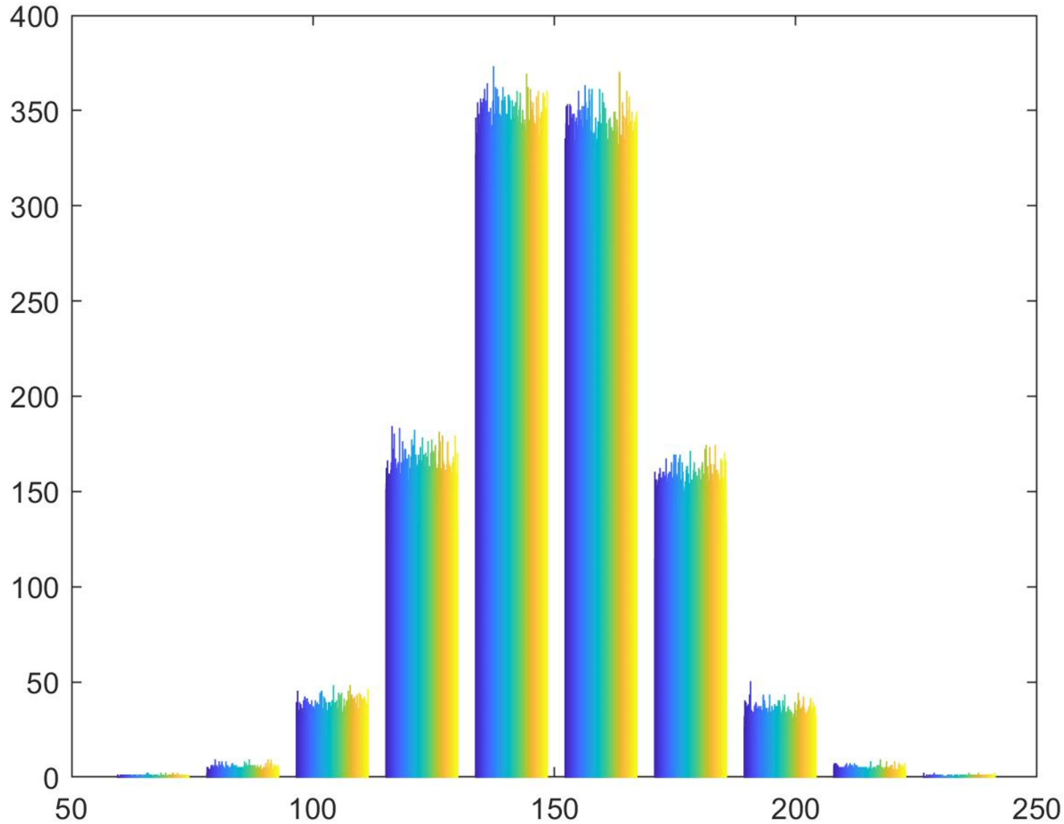
2b. Use the histogram function to plot the samples for each (μ,σ) pair. This should result in 3 plots. Compare and contrast the three plots, based on what you expect to see

Ans.

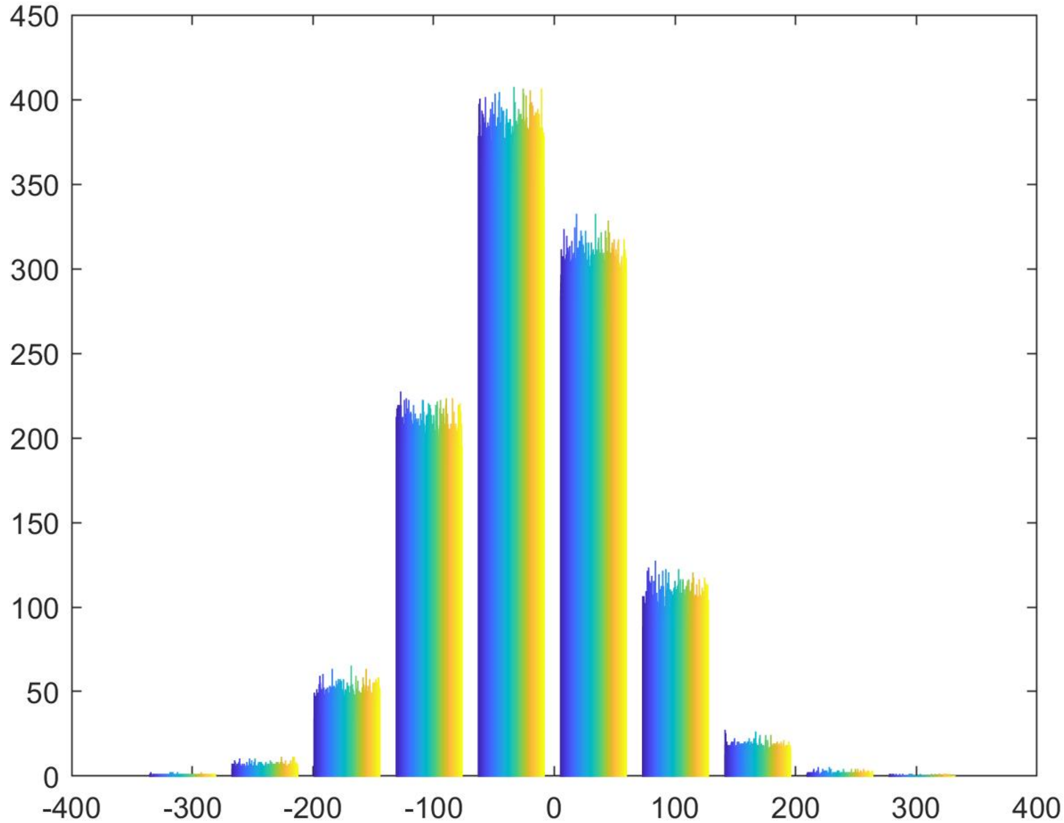
```
hist(bi) %most data should be around the mean (70), and the data should be focused, but not spread out
```



hist(bii) %most data should be around the mean (150), and the data is more spread out than bi.



hist(biii))%most data should be around the mean (-20), but the data is spread out, because the standard deviation is high



3a. Create two random variables X and Y , where X = [800, 1200, 1600] and Y = [400, 800, 1200].

Ans.
X = [800,1200,1600];
Y = [400,800,1200];

3b. Using the table shown below, create variable P xy, which stores the joint PMF between X and Y .

Ans.
jPmf = [0.2,0.05,0.1;0.05,0.2,0.1;0,0.1,0.2]

3c. Compute values for the marginal PMF of X (e.g. PX(x)) and the marginal PMF of Y (e.g. PY (y)).

Ans.
%PX(800) = 0.2 + 0.05 + 0.1 = 0.35
%PX(1200) = 0.05 + 0.2 + 0.1 = 0.35
%PX(1600) = 0 + 0.1 + 0.2 = 0.3
Px = [0.35, 0.35, 0.3]

```
%PY(400) = 0.2 + 0.05 + 0 = 0.25
%PY(800) = 0.05 + 0.2 + 0.1 = 0.35
%PY(1200) = 0.1 + 0.1 + 0.2 = 0.4
Py = [0.25,0.35,0.4]
```

3d. Write a MATLAB function called myexpvalue(x, P x) that computes the expected value of a discrete random variable x, using its PMF, P x. Hint: use x(:) and P x(:) in the calculation

Ans.

```
function f = myexpvalue(x, Px)
   f = sum(x .* Px)
end
```

3e. Compute the expected values for X and Y , using myexpvalue(x, P x).

Ans.

```
xExp = myexpvalue(X, Px)
xExp = 1180

yExp = myexpvalue(Y, Py)
yExp = 860
```

3f. Use the ndgrid function to generate a grid of the two random variables. Store the grids as Sx and Sy.

Ans.

```
[Sx, Sy] = ndgrid(X, Y)
Sx =

     800      800      800
    1200     1200     1200
    1600     1600     1600


Sy =

     400      800     1200
     400      800     1200
     400      800     1200
```

3g. Write a MATLAB function called mycovariance(Sx, Sy, P xy, Ex, Ey) that computes the covariance between a grid of random variables (Sx and Sy) using the joint PMF (e.g. P xy) and marginal expected values (e.g. Ex, Ey).

Ans.

```
function g = mycovariance(Sx, Sy, Pxy, Ex, Ey)
  g = sum(sum(Sx .* Sy .* Pxy)) - Ex * Ey
end
```

3h. Use mycovariance(Sx, Sy, P xy, Ex, Ey) to compute the covariance between X and Y

Ans.

```
covar = mycovariance(Sx, Sy, jPmf, xExp, yExp)
covar = 49200
```