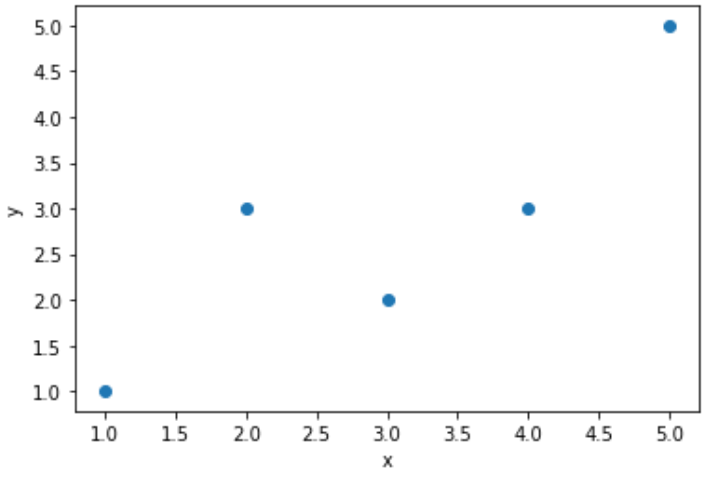
Simple Linear Regression

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

x = np.array([1,2,4,3,5])
y = np.array([1,3,3,2,5])
n = np.size(x)
print(x)
print(y)
plt.scatter(x,y)
plt.xlabel('x')
plt.ylabel('y')
plt.show()

```

```
[1 2 4 3 5]
[1 3 3 2 5]
```



In [8]:
```
#calculate mean of x and mean of y
mean_x= np.mean(x)
mean_y =np.mean(y)
print ("mean_x = ",mean_x," mean_y = ",mean_y)

#find the error for x and y  and multiplication of deviation of x and y i.e. res
sum_resi = np.sum((x-mean_x)*(y-mean_y))

#squared error for x
sq_xi = np.sum((x-mean_x)*(x-mean_x))

print ("residual and squared error")
print ("sum of residual error = ",sum_resi, "sum of squared x error = ",sq_xi)
```

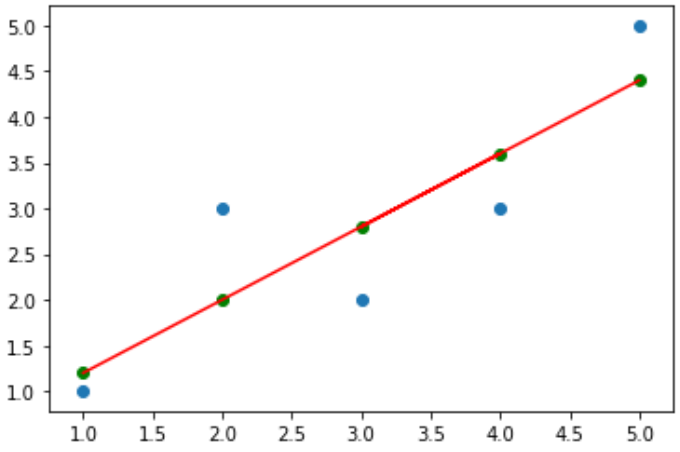
```
mean_x =  3.0  mean_y =  2.8
residual and squared error
sum of residual error =  8.0 sum of squared x error =  10.0
```

In [9]:
```
#calculate interceprt B1 and B0
B1 = sum_resi/sq_xi
B0 = mean_y - B1*mean_x
print ("B0 =",B0, "\nB1 =",B1)
```

```
B0 = 0.39999999999999947
B1 = 0.8
```

In [10]:
```
y_pred = B0 + B1*x
print ("predicted y values ",y_pred)
print ("actual y values ",y)
#calculate squared error for predicted values
sq_er = np.sum((y_pred-y)*(y_pred-y))
print (sq_er)
RMSE  = np.sqrt(sq_er/n)
print ("RMSE = ",RMSE)
plt.scatter(x,y)
plt.scatter(x,y_pred, color = "g")
plt.plot(x, y_pred,color = "r")
plt.show()
```

```
predicted y values  [1.2 2.  3.6 2.8 4.4]
actual y values  [1 3 3 2 5]
2.400000000000001
RMSE =  0.692820323027551
```



In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```