In [5]:

```
#Write a python program to determine the following for given |A| and |B|
# (a) Number of functions from A to B and Number of functions from B to A
# (b) Number of one to one functions from A to B and B to A
# (c) Number of onto functions from A to B and B to A
# (d) Number of bijective functions from A to B and B to A
from math import perm, comb, factorial
def fun(m,n):
    return n**m
def bij(m,n):
    if m==n:
        return factorial(n)
    else:
        return 0
def one2one(m,n):
    if m<=n:
        return perm(n,m)
    else:
        return 0
def on2(m,n):
    sum=0
    if m>=n:
        for k in range(n):
            sum+=(-1)**k*comb(n,n-k)*(n-k)**m
    return sum

m=int(input("Enter |A|:"))
n=int(input("Enter |B|:"))
print("Number of functions from A to B is ", fun(m,n))
print("Number of functions from B to A is  ", fun(n,m))
print("Number of one to one functions from A to B is ",one2one(m,n))
print("Number of one to one functions from B to A is ",one2one(n,m))
print("Number of onto functions from A to B is ",on2(m,n))
print("Number of onto functions from B to A is ",on2(n,m))
print("Number of bijective functions from A to B is ",bij(m,n))
print("Number of bijective functions from B to A is ",bij(n,m))
```

```
Enter |A|:4
Enter |B|:5
Number of functions from A to B is  625
Number of functions from B to A is   1024
Number of one to one functions from A to B is  120
Number of one to one functions from B to A is  0
Number of onto functions from A to B is  0
Number of onto functions from B to A is  240
Number of bijective functions from A to B is  0
Number of bijective functions from B to A is  0
```

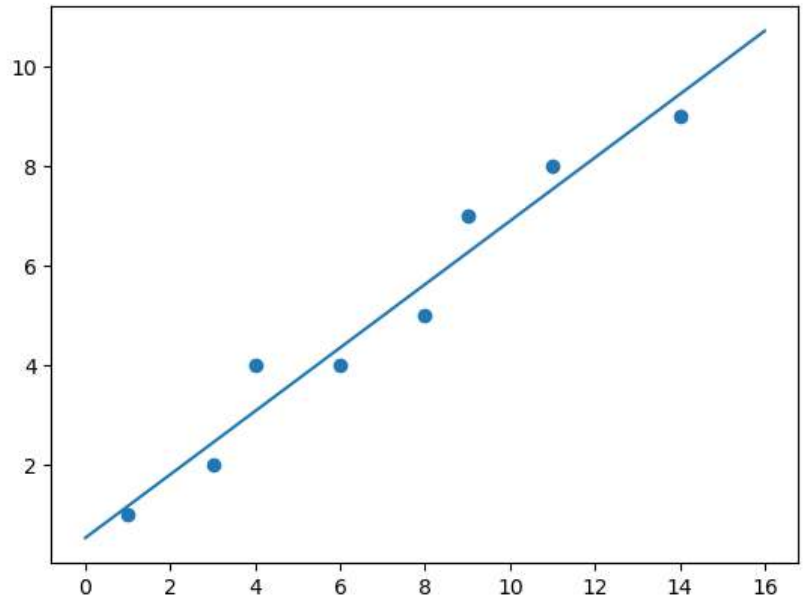
In [6]:

```
#write a python program to fit the following data to the curve y = a x + b
# x: 1  3  4  6  8  9  11  14
# y: 1  2  4  4  5  7  8   9
from numpy import sum
x=[1, 3, 4, 6, 8, 9, 11, 14]
y=[1, 2, 4, 4, 5, 7, 8, 9]
n=len(x)
sy=sum(y)
sx=sum(x)
x2=[i**2 for i in x]
sx2=sum(x2)
xy=[i*j for i,j in zip(x,y)]
sxy=sum(xy)
print("Σx=",sx,"Σy=",sy,"Σx^2=",sx2,"Σxy=",sxy)

from sympy import *
a,b,X = symbols('a,b,X')
Neq1 = Eq(a*sx+n*b,sy)
Neq2 = Eq(a*sx2+b*sx,sxy)
#result = solve([Neq1,Neq2],(a,b))
sol = solve([Neq1,Neq2],(a,b))
a1=sol[a]
b1=sol[b]

from pylab import *
Y=a1*X+b1
Y=lambdify(X,Y)
X=linspace(0,16,20)
plot(X,Y(X))
scatter(x,y)
show()
```

```
Σx= 56 Σy= 40 Σx^2= 524 Σxy= 364
```



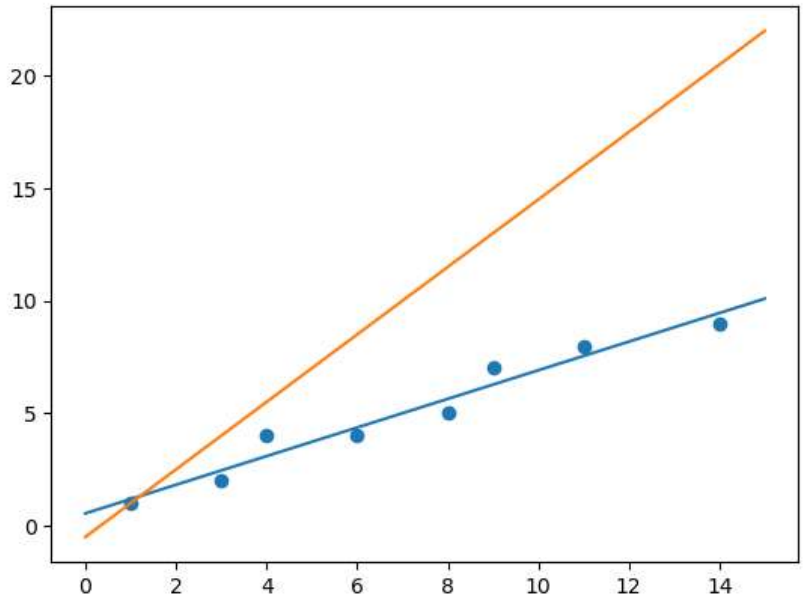
In [7]:

```
#write a python program to find the correlation coefficient and draw regression lines
#of y on x and x on y for the following data
# x: 1  3  4  6  8  9  11  14
# y: 1  2  4  4  5  7  8   9
x=[1, 3, 4, 6, 8, 9, 11, 14]
y=[1, 2, 4, 4, 5, 7, 8, 9]
n=len(x)
meanx=sum(x)/n
meany=sum(y)/n
devx=x-meanx
devy=y-meany
sdx=sqrt(sum([i**2 for i in devx])/n)
sdy=sqrt(sum([i**2 for i in devy])/n)
covxy=sum(devx*devy)/n
r=covxy/(sdx*sdy)

myx=r*sdy/sdx
mxy=r*sdx/sdy

from sympy import *
X1,Y2=symbols("X1,Y2")
from pylab import *
Y1=myx*X1-myx*meanx+meany #regression line of y on x
Y1=lambdify(X1,Y1)
X2=mxy*Y2-mxy*meany+meanx #regression line of x on y
X2=lambdify(Y2,X2)
X1=linspace(0,15,100)
plot(X1,Y1(X1))
Y2=linspace(0,15,100)
plot(Y2,X2(Y2))

scatter(x,y)
show()
```



In [8]:

```python
# Determine whether the following expression is Tautology or contradiction
# not ((p or q) and (not p or r) and (not q or r)) or r

def truthTable(expression,inputs):
    if inputs>3 or inputs<2:
        return(print("Numbe of primitive statements should be either 2 or 3"))

    expression = expression.replace("and","&")
    expression = expression.replace("or","|")
    expression = expression.replace("not","~")
    print("Logical Expression:")
    print("  X = ")
    display(expression)

    X=[]
    print("\nTruth Table:")
    if inputs==2:
        print("  -------------")
        print("  | p | q | X |")
        print("  -------------")
        for p in range(0,2):
            for q in range(0,2):
                x = abs(eval(expression))
                X.append(x)
                print("  | " + str(p) + " | " + str(q) + " | " + str(x) + " |" )
                print("  -------------")
    elif inputs==3:
        print("  -----------------")
        print("  | p | q | r | X |")
        print("  -----------------")
        for p in range(0,2):
            for q in range(0,2):
                for r in range(0,2):
                    x = abs(eval(expression))
                    X.append(x)
                    print("  | " + str(p) + " | " + str(q) + " | " + str(r) + " | " + str(x) + " |" )
                    print("  -----------------")
    check=all( i == X[0] for i in X)
    if check:
        if X[0]==0:
            print("Given logical expression is contradiction")
        else:
            print("Given logical expression is Tautology")
    else:
        print("Given logical expression is neither Tautology nor contradiction")
################################################
expression = "not ((p or q) and (not p or r) and (not q or r)) or r"
truthTable(expression,3)
```

```
Logical Expression:
  X =

'~ ((p | q) & (~ p | r) & (~ q | r)) | r'


Truth Table:
  -----------------
  | p | q | r | X |
  -----------------
  | 0 | 0 | 0 | 1 |
  -----------------
  | 0 | 0 | 1 | 1 |
  -----------------
  | 0 | 1 | 0 | 1 |
  -----------------
  | 0 | 1 | 1 | 1 |
  -----------------
  | 1 | 0 | 0 | 1 |
  -----------------
  | 1 | 0 | 1 | 1 |
  -----------------
  | 1 | 1 | 0 | 1 |
  -----------------
  | 1 | 1 | 1 | 1 |
  -----------------
Given logical expression is Tautology
```

In [10]:
```
# Four coins are tossed 100 times and the following results are obtained. Fit a binomial distribution
#for the data and test the goodness of fit at 5% level of significance by using python program
# x:  0   1   2   3   4
# f:  5  29  36  25   5

from math import comb
print("H0=Binomial distribution is a good fit for the given frequency distribution")
print("H1=Binomial distribution is not a good fit for the given frequency distribution")
x=[0,1,2,3,4]
f=[5,29,36,25,5]  #Oi
ΣOi=sum(f)
n=len(x)-1
p=float(input("\nenter the probability of success: "))
q=1-p  #probability of failure
px=[comb(n,i)*p**i*q**(n-i) for i in x]
Ei=[ΣOi*i for i in px]
ΣEi=sum(Ei)
print("\nEi: ",end=" ")
for i in Ei:
    print(i,end="    ")
print("\n")
print("ΣOi=",ΣOi,"   ΣEi=",ΣEi)
flag=[(f[i]-Ei[i])**2/Ei[i] for i in range(len(f))]

χ2=sum(flag)
print("\nχ2=",χ2)
df=n
χ2tab=float(input("\nEnter the tabulated value of χ2: "))
if χ2<=χ2tab:
    print("\nAccept the null hypothesis")
else:
    print("\nReject the null hypothesis and accept the alternate hypothesis")
```

```
H0=Binomial distribution is a good fit for the given frequency distribution
H1=Binomial distribution is not a good fit for the given frequency distribution

enter the probability of success: 0.5

Ei:  6.25    25.0    37.5    25.0    6.25

ΣOi= 100    ΣEi= 100.0

χ2= 1.2

Enter the tabulated value of χ2: 9.49

Accept the null hypothesis
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```