Program & Output (Grouped Data) ¶

In [127]:

```
import pandas as pd
import numpy as np
import math
```

In [128]:

```
marks_ai = ["30 - 40", "40 - 50", "50 - 60", "60 - 70"]
marks_ads = ["30 - 40", "40 - 50", "50 - 60", "60 - 70"]
frequency_matrix = [
    [3, 1, 1, 0],
    [2, 6, 1, 2],
    [1, 2, 2, 1],
    [0, 1, 1, 1]
]

pd.DataFrame({
    'marks': marks_ai,
    marks_ads[0]: frequency_matrix[0],
    marks_ads[1]: frequency_matrix[1],
    marks_ads[2]: frequency_matrix[2],
})
```

Out[128]:

	marks	30 - 40	40 - 50	50 - 60
0	30 - 40	3	2	1
1	40 - 50	1	6	2
2	50 - 60	1	1	2
3	60 - 70	0	2	1

In [129]:

```
def get_median(arr):
    median = []
    for item in arr:
        avg = 0
        interval = None
        for elem in item.split('-'):
            if interval is None:
                interval = float(elem)
            else:
                interval = float(elem) - interval
            e = float(elem)
            avg += e
        avg = avg/2
        median.append(avg)
    return median, interval

x, range_x = get_median(marks_ai)
y, range_y = get_median(marks_ads)
```

In [130]:

```
Ax = x[int(len(x)/2)]
Ay = x[int(len(y)/2)]

dx = [(i - Ax) / range_x for i in x]
dy = [(i - Ay) / range_x for i in x]

pd.DataFrame({ 'dx': dx, 'dy': dy })
```

Out[130]:

	dx	dy
0	-2.0	-2.0
1	-1.0	-1.0
2	0.0	0.0
3	1.0	1.0

In [131]:

```
def update_frequency_matrix():
    matrix = []
    for i in range(len(frequency_matrix)):
        arr = []
        for j in range(len(frequency_matrix[i])):
            arr.append(dx[i]*dx[j]*frequency_matrix[i][j])
        matrix.append(arr)
    return matrix

updated_matrix = update_frequency_matrix()
pd.DataFrame(updated_matrix)
```

Out[131]:

	0	1	2	3
0	12.0	2.0	-0.0	-0.0
1	4.0	6.0	-0.0	-2.0
2	-0.0	-0.0	0.0	0.0
3	-0.0	-1.0	0.0	1.0

In [132]:

```
freqx = []
freqy = []
freqx_dx = []
freqy_dy = []
freqx_dx_dx = []
freqy_dy_dy = []
for i in range(len(marks_ai)):
    sum1 = 0
    sum2 = 0
    for j in range(len(frequency_matrix[i])):
        sum1 = sum1 + frequency_matrix[i][j]
        sum2 = sum2 + frequency_matrix[j][i]
    freqx.append(sum1)
    freqy.append(sum2)
    freqx_dx.append(sum1*dx[i])
    freqy_dy.append(sum2*dy[i])
    freqx_dx_dx.append(sum1*dx[i]*dx[i])
    freqy_dy_dy.append(sum2*dy[i]*dy[i])

freqx_dx_dy = []
freqy_dx_dy = []
for i in range(len(updated_matrix)):
    sum1 = 0
    sum2 = 0
    for j in range(len(updated_matrix[i])):
        sum1 = sum1 + updated_matrix[i][j]
        sum2 = sum2 + updated_matrix[j][i]
    freqx_dx_dy.append(sum1)
    freqy_dx_dy.append(sum2)
```

In [133]:

```
pd.DataFrame({
    'freqx': freqx,
    'freqx_dx': freqx_dx,
    'freqx_dx_dx': freqx_dx_dx,
    'freqy': freqy,
    'freqy_dy': freqy_dy,
    'freqy_dy_dy': freqy_dy_dy,
    'freqx_dx_dy': freqx_dx_dy,
    'freqx_dx_dy': freqy_dx_dy
})
```

Out[133]:

	freqx	freqx_dx	freqx_dx_dx	freqy	freqy_dy	freqy_dy_dy	freqx_dx_dy
0	5	-10.0	20.0	6	-12.0	24.0	16.0
1	11	-11.0	11.0	10	-10.0	10.0	7.0
2	6	0.0	0.0	5	0.0	0.0	0.0
3	3	3.0	3.0	4	4.0	4.0	-1.0

In [134]:

```
def E(arr):
    sum1 = 0
    for item in arr:
        sum1 = sum1 + item
    return sum1
```

In [135]:

```
# Applying the correlation formula
N = E(freqx)
Efdx = E(freqx_dx)
Efdx_squared = E(freqx_dx_dx)
Efdy = E(freqy_dy)
Efdy_squared = E(freqy_dy_dy)
Efdxdy = E(freqx_dx_dy)

numerator = Efdxdy - ( (Efdx*Efdy) / N)
denominator = math.sqrt(Efdx_squared - ( (Efdx*Efdx) / N )) * math.sqrt(Efdy_squared - ( (Efdy*Efd

r = numerator / denominator

print("Correlation: ", end='')
print(r)
if 0.3 < r < 0.75:
    print('It is Moderately positively Correlated!')
elif 0.75 <= r < 1:
    print('It is Highly positively Correlated!')
elif r >= 1:
    print('It is perfect positively Correlated!')
elif -0.3 < r < -0.75:
    print('It is Moderately negatively Correlated!')
elif -0.75 <= r < -0.1:
    print('It is Highly negatively Correlated!')
elif r <= -1:
    print('It is perfect negatively Correlated!')
else:
    print('It is not that correlated')
```

```
Correlation: 0.393847778765559274
It is Moderately positively Correlated!
```

In [ ]: