Phase 4 :

data =

1	4
1	3
2	3
2	6
3	1
3	2
3	4
3	6
4	7
4	5
4	1
4	3
4	6
5	8
5	4
5	9
5	6
6	4
6	2
6	3
6	5
6	10
7	10
7	8
7	4
8	7
8	9
8	5
9	5
9	8
9	10
10	7
10	9
10	6

Phase 3 output :

{1 : [(1, 3, 4)]}
{2 : [(2, 3, 6)]}
{3 : [(1, 3, 4) (2, 3, 6) (3, 4, 6)]}
{4 : [(1, 3, 4) (3, 4, 6) (4, 5, 6)]}
{5 : [(4, 5, 6) (5, 8, 9)]}
{6 : [(2, 3, 6) (3, 4, 6) (4, 5, 6)]}
{7 : [ ]}
{8 : [(5, 8, 9)]}
{9 : [(5, 8, 9)]}
{10 : [ ]}

No of possible triangles for each node:

Node 1: 1, 3, 4

Node 2: 2, 3, 6

Node 3: 1, 2, 3 1, 4, 3 2, 4, 3 1, 3, 6 3, 2, 6
3, 4, 6

Node 4: 1, 3, 4 1, 4, 5 1, 4, 6 1, 4, 7 3, 4, 5
3, 4, 6 3, 4, 7 4, 5, 6 4, 5, 7 4, 6, 7

Node 5: 5, 4, 6 5, 4, 8 5, 4, 9 5, 6, 8 5, 6, 9
5, 8, 9

Node 6: 2, 3, 6 3, 5, 6 2, 5, 6 2, 6, 10 3, 4, 6
3, 6, 5 3, 6, 10 4, 5, 6 4, 6, 10 5, 6, 10.

clustering coefficient

$$\text{Node 1 CC} = \frac{\#\text{ of triangles incident on Node 1}}{\#\text{ of triangles possible } ({}^{n}C_2)}$$ where

n: no of vertices the node is connected to.

$$\frac{1}{{}^{2}C_2} = \frac{1}{1} = 1$$

$$\text{Node 2 CC} = \frac{1}{1} = \frac{1}{{}^{2}C_2} = \frac{1}{1}$$

$$\text{Node 3 CC} = \frac{3}{6} = \frac{1}{2}$$

$$\text{Node 4 (CC)} = \frac{3}{5C_2} = \frac{3}{\frac{5!}{3!\,2!}} = \frac{3}{10}.$$

$$\text{Node 5 (CC)} = \frac{2}{4C_2} = \frac{2}{\frac{4!}{2!\,2!}} = \frac{1}{3}$$

$$\text{Node 6 (CC)} = \frac{3}{5C_2} = \frac{3}{\frac{5!}{3!\,2!}} = \frac{3}{10}.$$

Algorithm.

1) Phase 1 output is as follows:

df_1 = (1, [3, 4])
(2, [3, 6])
(3, [1, 2, 4, 6])
(4, [1, 3, 5, 6, 7])
(5, [4, 6, 8, 9])
(6, [2, 3, 4, 5, 10])
(7, [4, 8, 10])
(8, [5, 7, 9])
(9, [5, 8, 10])
(10, [6, 7, 9])

Against each node is the list of nodes connected directly to it.

2) Phase 3 output is as follows

```
df_2 = {1 : [(1, 3, 4)]
        {2 : [(2, 3, 6)]
        {3 : [(1, 3, 4), (2, 3, 6), (3, 4, 6)]
        {4 : [(1, 3, 4), (3, 4, 6), (4, 5, 6)]
        {5 : [(4, 5, 6), (5, 8, 9)]
        {6 : [(2, 3, 6), (3, 4, 6), (4, 5, 6)]
        {7 : [ ]
        {8 : [(5, 8, 9)]
        {9 : [(5, 8, 9)]
        {10 : [ ]
```

Against each node is the list of ~~tri~~ nodes that form a triangle. This would be the number of possible triangles for each node.

3) clustering_coefficient = { }

```
for i ← 0 to len(df_2):
begin
    clustering_coefficient[i] = len(df_2[str(i)]) / calculate_comb(df_1[str(i)])
endfor
```

```
fn Calculate_comb(n)
begin
    x = len(n[1])
    nC2 = fact(x) / fact(2) . fact(x-2)
    return nC2
end
```

Node 7 (CC) = 0

Node 8 (CC) = $\frac{1}{{}^{3}C_{2}} = \frac{1}{\frac{3!}{2!\,1!}} = \frac{1}{3}$

Node 9 (CC) = $\frac{1}{{}^{3}C_{2}} = \frac{1}{3}$

Node 10 (CC) = 0.

clustering_coefficient =

{
"1" : 1,
"2" : 1,
"3" : 0.5,
"4" : 0.3,
"5" : 0.33,
"6" : 0.3,
"7" : 0,
"8" : 0.33,
"9" : 0.33,
"10" : 0
}