Big Data Analytics – CS7070

Programming Project #2

Disha Rao, M13254448

PHASE-2

2. **Phase-2**: Write a program for Spark (in PySpark, Scala, or Spark-JAVA) that takes as input a graph in the list representation (generated in phase-1 above) and produces a list of records such that the key is a node number and the value is the 2-hop projection of the node in the graph.

Items to be submitted: (i) your program source code, (ii) Output of your program for the graph representation generated in Phase-2 for the TinyDataSet, (iii) Output of your program for the output generated by Phase-1 for the SmallDataSet.

Solution:

(i) Program Source code for tinydataset:

(Reading Phase1 output from HDFS)

```
phase1_tiny_rdd = sc.textFile("/home/maria_dev/pp2/phase2/Phase1_output_tinydataset.txt")

#phase1_tiny_transform = phase1_tiny_rdd.collect()
phase1_tiny_transform = phase1_tiny_rdd.map(lambda x:eval(x))
phase1_tiny_transform.collect()

[(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])]

phase1_test = phase1_tiny_transform.map(lambda x:{x[0]:x[1]})
phase1_test.collect()

def trans(x,y):
    x.update(y)
    return x

phase1_test_new = phase1_test.reduce(lambda x,y:trans(x,y))

phase1_test_new
{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]}

phase1_tiny_transformed = phase1_tiny_rdd.map(lambda x:eval(x)).map(lambda x:(x[0],[{i:phase1_test_new[i]} for i in x[1]])).map(lambda x:{x[0]:[{k: v for d in x[1] for k, v in d.items()}]})
phase1_tiny_transformed.collect()


[{1: [{3: [1, 2, 4, 6], 4: [1, 3, 5, 6, 7]}]},
{2: [{3: [1, 2, 4, 6], 6: [2, 3, 4, 5, 10]}]},
{3: [{1: [3, 4], 2: [3, 6], 4: [1, 3, 5, 6, 7], 6: [2, 3, 4, 5, 10]}]},
```

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

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

phase1_tiny_transformed.coalesce(1).saveAsTextFile("/home/maria_dev/pp2/phase2/phase2_output_tinydataset")
```

(ii) Output for tinydataset:

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

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

(iii) Source code for Small dataset:

```
%pyspark
phase1_small_rdd = sc.textFile("/home/maria_dev/pp2/phase2/Phase1_output_smalldataset.txt")

%pyspark
phase1_small_transform = phase1_small_rdd.map(lambda x:eval(x))
phase1_small_transform.collect()

[(1, [2, 3, 6, 10, 37]), (2, [1, 3, 6, 7, 11]), (3, [1, 2, 4, 5, 7, 8, 12]), (4, [3, 5, 8, 9, 13]), (5, [3, 4, 9, 33]), (6, [1, 2, 7, 10, 11]), (7, [2, 3, 6, 8, 11, 12]), (8, [3, 4, 7, 9, 12, 13]), (9, [4, 5, 8, 13, 14]), (10, [1, 6, 11, 15, 19, 38]), (11, [2, 6, 7, 10, 12, 15, 16]), (12, [3, 7, 8, 11, 13, 16, 17]), (13, [4, 8, 9, 12, 14, 17, 18]), (14, [9, 13, 18, 23, 34]), (15, [10, 11, 16, 19]), (16, [11, 12, 15, 17, 20, 21]), (17, [12, 13, 16, 21, 22]), (18, [13, 14, 22, 23]), (19, [10, 15, 20, 24, 28, 39]), (20, [16, 19, 21, 24, 25, 29]), (21, [16, 17, 20, 22, 25, 26, 30]), (22, [17, 18, 21, 23, 26, 27, 31]), (23, [14, 18, 22, 27, 32, 35]), (24, [19, 20, 28, 29]), (25, [20, 21, 29, 30]), (26, [21, 22, 27, 30, 31]), (27, [22, 23, 26, 31, 32]), (28, [19, 24, 29, 30, 31, 32, 40]), (29, [20, 24, 25, 28, 30, 31]), (30, [21, 25, 26, 28, 29, 31, 32]), (31, [22, 26, 27, 28, 29, 30, 32]), (32, [23, 27, 28, 30, 31, 36]), (33, [5, 34]), (34, [14, 33, 35]), (35, [23, 34, 36]), (36, [32, 35]), (37, [1, 38]), (38, [10, 37, 39]), (39, [19, 38, 40]), (40, [28, 39])]

%pyspark
phase1_test_small = phase1_small_transform.map(lambda x:{x[0]:x[1]})
phase1_test_small.collect()

[{1: [2, 3, 6, 10, 37]}, {2: [1, 3, 6, 7, 11]}, {3: [1, 2, 4, 5, 7, 8, 12]}, {4: [3, 5, 8, 9, 13]}, {5: [3, 4, 9, 33]}, {6: [1, 2, 7, 10, 11]}, {7: [2, 3, 6, 8, 11, 12]}, {8: [3, 4, 7, 9, 12, 13]}, {9: [4, 5, 8, 13, 14]}, {10: [1, 6, 11, 15, 19, 38]}, {11: [2,
```

```
    6, 7, 10, 12, 15, 16]}, {12: [3, 7, 8, 11, 13, 16, 17]}, {13: [4, 8, 9, 12, 14, 17, 18]
    }, {14: [9, 13, 18, 23, 34]}, {15: [10, 11, 16, 19]}, {16: [11, 12, 15, 17, 20, 21]}, {
    17: [12, 13, 16, 21, 22]}, {18: [13, 14, 22, 23]}, {19: [10, 15, 20, 24, 28, 39]}, {20:
     [16, 19, 21, 24, 25, 29]}, {21: [16, 17, 20, 22, 25, 26, 30]}, {22: [17, 18, 21, 23, 2
    6, 27, 31]}, {23: [14, 18, 22, 27, 32, 35]}, {24: [19, 20, 28, 29]}, {25: [20, 21, 29,
    30]}, {26: [21, 22, 27, 30, 31]}, {27: [22, 23, 26, 31, 32]}, {28: [19, 24, 29, 30, 31,
     32, 40]}, {29: [20, 24, 25, 28, 30, 31]}, {30: [21, 25, 26, 28, 29, 31, 32]}, {31: [22
    , 26, 27, 28, 29, 30, 32]}, {32: [23, 27, 28, 30, 31, 36]}, {33: [5, 34]}, {34: [14, 33
    , 35]}, {35: [23, 34, 36]}, {36: [32, 35]}, {37: [1, 38]}, {38: [10, 37, 39]}, {39: [19
    , 38, 40]}, {40: [28, 39]}]
15.
16. %pyspark
17. def trans(x,y):
18.     x.update(y)
19.     return x
20.
21. phase1_test_new = phase1_test_small.reduce(lambda x,y:trans(x,y))
22.
23. phase1_test_new
24.
25. {1: [2, 3, 6, 10, 37], 2: [1, 3, 6, 7, 11], 3: [1, 2, 4, 5, 7, 8, 12], 4: [3, 5, 8, 9,
    13], 5: [3, 4, 9, 33], 6: [1, 2, 7, 10, 11], 7: [2, 3, 6, 8, 11, 12], 8: [3, 4, 7, 9, 1
    2, 13], 9: [4, 5, 8, 13, 14], 10: [1, 6, 11, 15, 19, 38], 11: [2, 6, 7, 10, 12, 15, 16]
    , 12: [3, 7, 8, 11, 13, 16, 17], 13: [4, 8, 9, 12, 14, 17, 18], 14: [9, 13, 18, 23, 34]
    , 15: [10, 11, 16, 19], 16: [11, 12, 15, 17, 20, 21], 17: [12, 13, 16, 21, 22], 18: [13
    , 14, 22, 23], 19: [10, 15, 20, 24, 28, 39], 20: [16, 19, 21, 24, 25, 29], 21: [16, 17,
     20, 22, 25, 26, 30], 22: [17, 18, 21, 23, 26, 27, 31], 23: [14, 18, 22, 27, 32, 35], 2
    4: [19, 20, 28, 29], 25: [20, 21, 29, 30], 26: [21, 22, 27, 30, 31], 27: [22, 23, 26, 3
    1, 32], 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26
    , 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 30, 31, 36], 33:
    [5, 34], 34: [14, 33, 35], 35: [23, 34, 36], 36: [32, 35], 37: [1, 38], 38: [10, 37, 39
    ], 39: [19, 38, 40], 40: [28, 39]}
26.
27. %pyspark
28. phase1_small_transformed = phase1_small_rdd.map(lambda x:eval(x)).map(lambda x:(x[0],[{
    i:phase1_test_new[i]} for i in x[1]])).map(lambda x:{x[0]:[{k: v for d in x[1] for k, v
     in d.items()}]})
29. phase1_small_transformed.collect()
30.
31. [{1: [{2: [1, 3, 6, 7, 11], 3: [1, 2, 4, 5, 7, 8, 12], 6: [1, 2, 7, 10, 11], 10: [1, 6,
     11, 15, 19, 38], 37: [1, 38]}]}, {2: [{1: [2, 3, 6, 10, 37], 3: [1, 2, 4, 5, 7, 8, 12]
    , 6: [1, 2, 7, 10, 11], 7: [2, 3, 6, 8, 11, 12], 11: [2, 6, 7, 10, 12, 15, 16]}]}, {3:
    [{1: [2, 3, 6, 10, 37], 2: [1, 3, 6, 7, 11], 4: [3, 5, 8, 9, 13], 5: [3, 4, 9, 33], 7:
    [2, 3, 6, 8, 11, 12], 8: [3, 4, 7, 9, 12, 13], 12: [3, 7, 8, 11, 13, 16, 17]}]}, {4: [{
    3: [1, 2, 4, 5, 7, 8, 12], 5: [3, 4, 9, 33], 8: [3, 4, 7, 9, 12, 13], 9: [4, 5, 8, 13,
    14], 13: [4, 8, 9, 12, 14, 17, 18]}]}, {5: [{3: [1, 2, 4, 5, 7, 8, 12], 4: [3, 5, 8, 9,
     13], 9: [4, 5, 8, 13, 14], 33: [5, 34]}]}, {6: [{1: [2, 3, 6, 10, 37], 2: [1, 3, 6, 7,
     11], 7: [2, 3, 6, 8, 11, 12], 10: [1, 6, 11, 15, 19, 38], 11: [2, 6, 7, 10, 12, 15, 16
    ]}]}, {7: [{2: [1, 3, 6, 7, 11], 3: [1, 2, 4, 5, 7, 8, 12], 6: [1, 2, 7, 10, 11], 8: [3
    , 4, 7, 9, 12, 13], 11: [2, 6, 7, 10, 12, 15, 16], 12: [3, 7, 8, 11, 13, 16, 17]}]}, {8
    : [{3: [1, 2, 4, 5, 7, 8, 12], 4: [3, 5, 8, 9, 13], 7: [2, 3, 6, 8, 11, 12], 9: [4, 5,
    8, 13, 14], 12: [3, 7, 8, 11, 13, 16, 17], 13: [4, 8, 9, 12, 14, 17, 18]}]}, {9: [{4: [
    3, 5, 8, 9, 13], 5: [3, 4, 9, 33], 8: [3, 4, 7, 9, 12, 13], 13: [4, 8, 9, 12, 14, 17, 1
    8], 14: [9, 13, 18, 23, 34]}]}, {10: [{1: [2, 3, 6, 10, 37], 6: [1, 2, 7, 10, 11], 11:
    [2, 6, 7, 10, 12, 15, 16], 15: [10, 11, 16, 19], 19: [10, 15, 20, 24, 28, 39], 38: [10,
     37, 39]}]}, {11: [{2: [1, 3, 6, 7, 11], 6: [1, 2, 7, 10, 11], 7: [2, 3, 6, 8, 11, 12],
     10: [1, 6, 11, 15, 19, 38], 12: [3, 7, 8, 11, 13, 16, 17], 15: [10, 11, 16, 19], 16: [
    11, 12, 15, 17, 20, 21]}]}, {12: [{3: [1, 2, 4, 5, 7, 8, 12], 7: [2, 3, 6, 8, 11, 12],
    8: [3, 4, 7, 9, 12, 13], 11: [2, 6, 7, 10, 12, 15, 16], 13: [4, 8, 9, 12, 14, 17, 18],
    16: [11, 12, 15, 17, 20, 21], 17: [12, 13, 16, 21, 22]}]}, {13: [{4: [3, 5, 8, 9, 13],
    8: [3, 4, 7, 9, 12, 13], 9: [4, 5, 8, 13, 14], 12: [3, 7, 8, 11, 13, 16, 17], 14: [9, 1
```

```
3, 18, 23, 34], 17: [12, 13, 16, 21, 22], 18: [13, 14, 22, 23]}]}, {14: [{9: [4, 5, 8,
13, 14], 13: [4, 8, 9, 12, 14, 17, 18], 18: [13, 14, 22, 23], 23: [14, 18, 22, 27, 32,
35], 34: [14, 33, 35]}]}, {15: [{10: [1, 6, 11, 15, 19, 38], 11: [2, 6, 7, 10, 12, 15,
16], 16: [11, 12, 15, 17, 20, 21], 19: [10, 15, 20, 24, 28, 39]}]}, {16: [{11: [2, 6, 7
, 10, 12, 15, 16], 12: [3, 7, 8, 11, 13, 16, 17], 15: [10, 11, 16, 19], 17: [12, 13, 16
, 21, 22], 20: [16, 19, 21, 24, 25, 29], 21: [16, 17, 20, 22, 25, 26, 30]}]}, {17: [{12
: [3, 7, 8, 11, 13, 16, 17], 13: [4, 8, 9, 12, 14, 17, 18], 16: [11, 12, 15, 17, 20, 21
], 21: [16, 17, 20, 22, 25, 26, 30], 22: [17, 18, 21, 23, 26, 27, 31]}]}, {18: [{13: [4
, 8, 9, 12, 14, 17, 18], 14: [9, 13, 18, 23, 34], 22: [17, 18, 21, 23, 26, 27, 31], 23:
 [14, 18, 22, 27, 32, 35]}]}, {19: [{10: [1, 6, 11, 15, 19, 38], 15: [10, 11, 16, 19],
20: [16, 19, 21, 24, 25, 29], 24: [19, 20, 28, 29], 28: [19, 24, 29, 30, 31, 32, 40], 3
9: [19, 38, 40]}]}, {20: [{16: [11, 12, 15, 17, 20, 21], 19: [10, 15, 20, 24, 28, 39],
21: [16, 17, 20, 22, 25, 26, 30], 24: [19, 20, 28, 29], 25: [20, 21, 29, 30], 29: [20,
24, 25, 28, 30, 31]}]}, {21: [{16: [11, 12, 15, 17, 20, 21], 17: [12, 13, 16, 21, 22],
20: [16, 19, 21, 24, 25, 29], 22: [17, 18, 21, 23, 26, 27, 31], 25: [20, 21, 29, 30], 2
6: [21, 22, 27, 30, 31], 30: [21, 25, 26, 28, 29, 31, 32]}]}, {22: [{17: [12, 13, 16, 2
1, 22], 18: [13, 14, 22, 23], 21: [16, 17, 20, 22, 25, 26, 30], 23: [14, 18, 22, 27, 32
, 35], 26: [21, 22, 27, 30, 31], 27: [22, 23, 26, 31, 32], 31: [22, 26, 27, 28, 29, 30,
 32]}]}, {23: [{14: [9, 13, 18, 23, 34], 18: [13, 14, 22, 23], 22: [17, 18, 21, 23, 26,
 27, 31], 27: [22, 23, 26, 31, 32], 32: [23, 27, 28, 30, 31, 36], 35: [23, 34, 36]}]},
{24: [{19: [10, 15, 20, 24, 28, 39], 20: [16, 19, 21, 24, 25, 29], 28: [19, 24, 29, 30,
 31, 32, 40], 29: [20, 24, 25, 28, 30, 31]}]}, {25: [{20: [16, 19, 21, 24, 25, 29], 21:
 [16, 17, 20, 22, 25, 26, 30], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26, 28, 29, 3
1, 32]}]}, {26: [{21: [16, 17, 20, 22, 25, 26, 30], 22: [17, 18, 21, 23, 26, 27, 31], 2
7: [22, 23, 26, 31, 32], 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30,
 32]}]}, {27: [{22: [17, 18, 21, 23, 26, 27, 31], 23: [14, 18, 22, 27, 32, 35], 26: [21
, 22, 27, 30, 31], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 30, 31, 36]}]}, {
28: [{19: [10, 15, 20, 24, 28, 39], 24: [19, 20, 28, 29], 29: [20, 24, 25, 28, 30, 31],
 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 3
0, 31, 36], 40: [28, 39]}]}, {29: [{20: [16, 19, 21, 24, 25, 29], 24: [19, 20, 28, 29],
 25: [20, 21, 29, 30], 28: [19, 24, 29, 30, 31, 32, 40], 30: [21, 25, 26, 28, 29, 31, 3
2], 31: [22, 26, 27, 28, 29, 30, 32]}]}, {30: [{21: [16, 17, 20, 22, 25, 26, 30], 25: [
20, 21, 29, 30], 26: [21, 22, 27, 30, 31], 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 2
4, 25, 28, 30, 31], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 30, 31, 36]}]},
{31: [{22: [17, 18, 21, 23, 26, 27, 31], 26: [21, 22, 27, 30, 31], 27: [22, 23, 26, 31,
 32], 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26,
28, 29, 31, 32], 32: [23, 27, 28, 30, 31, 36]}]}, {32: [{23: [14, 18, 22, 27, 32, 35],
27: [22, 23, 26, 31, 32], 28: [19, 24, 29, 30, 31, 32, 40], 30: [21, 25, 26, 28, 29, 31
, 32], 31: [22, 26, 27, 28, 29, 30, 32], 36: [32, 35]}]}, {33: [{5: [3, 4, 9, 33], 34:
[14, 33, 35]}]}, {34: [{14: [9, 13, 18, 23, 34], 33: [5, 34], 35: [23, 34, 36]}]}, {35:
 [{23: [14, 18, 22, 27, 32, 35], 34: [14, 33, 35], 36: [32, 35]}]}, {36: [{32: [23, 27,
 28, 30, 31, 36], 35: [23, 34, 36]}]}, {37: [{1: [2, 3, 6, 10, 37], 38: [10, 37, 39]}]}
, {38: [{10: [1, 6, 11, 15, 19, 38], 37: [1, 38], 39: [19, 38, 40]}]}, {39: [{19: [10,
15, 20, 24, 28, 39], 38: [10, 37, 39], 40: [28, 39]}]}, {40: [{28: [19, 24, 29, 30, 31,
 32, 40], 39: [19, 38, 40]}]}]
```

```
%pyspark
phase1_small_transformed.coalesce(1).saveAsTextFile("/home/maria_dev/pp2/phase2_output_
smalldataset")
```

(iii) Output for small dataset:

```
{1: [{2: [1, 3, 6, 7, 11], 3: [1, 2, 4, 5, 7, 8, 12], 6: [1, 2, 7, 10, 11], 10: [1, 6,
11, 15, 19, 38], 37: [1, 38]}]}
{2: [{1: [2, 3, 6, 10, 37], 3: [1, 2, 4, 5, 7, 8, 12], 6: [1, 2, 7, 10, 11], 7: [2, 3,
6, 8, 11, 12], 11: [2, 6, 7, 10, 12, 15, 16]}]}
{3: [{1: [2, 3, 6, 10, 37], 2: [1, 3, 6, 7, 11], 4: [3, 5, 8, 9, 13], 5: [3, 4, 9, 33],
 7: [2, 3, 6, 8, 11, 12], 8: [3, 4, 7, 9, 12, 13], 12: [3, 7, 8, 11, 13, 16, 17]}]}
{4: [{3: [1, 2, 4, 5, 7, 8, 12], 5: [3, 4, 9, 33], 8: [3, 4, 7, 9, 12, 13], 9: [4, 5, 8
, 13, 14], 13: [4, 8, 9, 12, 14, 17, 18]}]}
```

```
5. {5: [{3: [1, 2, 4, 5, 7, 8, 12], 4: [3, 5, 8, 9, 13], 9: [4, 5, 8, 13, 14], 33: [5, 34]}]}
6. {6: [{1: [2, 3, 6, 10, 37], 2: [1, 3, 6, 7, 11], 7: [2, 3, 6, 8, 11, 12], 10: [1, 6, 11, 15, 19, 38], 11: [2, 6, 7, 10, 12, 15, 16]}]}
7. {7: [{2: [1, 3, 6, 7, 11], 3: [1, 2, 4, 5, 7, 8, 12], 6: [1, 2, 7, 10, 11], 8: [3, 4, 7, 9, 12, 13], 11: [2, 6, 7, 10, 12, 15, 16], 12: [3, 7, 8, 11, 13, 16, 17]}]}
8. {8: [{3: [1, 2, 4, 5, 7, 8, 12], 4: [3, 5, 8, 9, 13], 7: [2, 3, 6, 8, 11, 12], 9: [4, 5, 8, 13, 14], 12: [3, 7, 8, 11, 13, 16, 17], 13: [4, 8, 9, 12, 14, 17, 18]}]}
9. {9: [{4: [3, 5, 8, 9, 13], 5: [3, 4, 9, 33], 8: [3, 4, 7, 9, 12, 13], 13: [4, 8, 9, 12, 14, 17, 18], 14: [9, 13, 18, 23, 34]}]}
10. {10: [{1: [2, 3, 6, 10, 37], 6: [1, 2, 7, 10, 11], 11: [2, 6, 7, 10, 12, 15, 16], 15: [10, 11, 16, 19], 19: [10, 15, 20, 24, 28, 39], 38: [10, 37, 39]}]}
11. {11: [{2: [1, 3, 6, 7, 11], 6: [1, 2, 7, 10, 11], 7: [2, 3, 6, 8, 11, 12], 10: [1, 6, 11, 15, 19, 38], 12: [3, 7, 8, 11, 13, 16, 17], 15: [10, 11, 16, 19], 16: [11, 12, 15, 17, 20, 21]}]}
12. {12: [{3: [1, 2, 4, 5, 7, 8, 12], 7: [2, 3, 6, 8, 11, 12], 8: [3, 4, 7, 9, 12, 13], 11: [2, 6, 7, 10, 12, 15, 16], 13: [4, 8, 9, 12, 14, 17, 18], 16: [11, 12, 15, 17, 20, 21], 17: [12, 13, 16, 21, 22]}]}
13. {13: [{4: [3, 5, 8, 9, 13], 8: [3, 4, 7, 9, 12, 13], 9: [4, 5, 8, 13, 14], 12: [3, 7, 8, 11, 13, 16, 17], 14: [9, 13, 18, 23, 34], 17: [12, 13, 16, 21, 22], 18: [13, 14, 22, 23]}]}
14. {14: [{9: [4, 5, 8, 13, 14], 13: [4, 8, 9, 12, 14, 17, 18], 18: [13, 14, 22, 23], 23: [14, 18, 22, 27, 32, 35], 34: [14, 33, 35]}]}
15. {15: [{10: [1, 6, 11, 15, 19, 38], 11: [2, 6, 7, 10, 12, 15, 16], 16: [11, 12, 15, 17, 20, 21], 19: [10, 15, 20, 24, 28, 39]}]}
16. {16: [{11: [2, 6, 7, 10, 12, 15, 16], 12: [3, 7, 8, 11, 13, 16, 17], 15: [10, 11, 16, 19], 17: [12, 13, 16, 21, 22], 20: [16, 19, 21, 24, 25, 29], 21: [16, 17, 20, 22, 25, 26, 30]}]}
17. {17: [{12: [3, 7, 8, 11, 13, 16, 17], 13: [4, 8, 9, 12, 14, 17, 18], 16: [11, 12, 15, 17, 20, 21], 21: [16, 17, 20, 22, 25, 26, 30], 22: [17, 18, 21, 23, 26, 27, 31]}]}
18. {18: [{13: [4, 8, 9, 12, 14, 17, 18], 14: [9, 13, 18, 23, 34], 22: [17, 18, 21, 23, 26, 27, 31], 23: [14, 18, 22, 27, 32, 35]}]}
19. {19: [{10: [1, 6, 11, 15, 19, 38], 15: [10, 11, 16, 19], 20: [16, 19, 21, 24, 25, 29], 24: [19, 20, 28, 29], 28: [19, 24, 29, 30, 31, 32, 40], 39: [19, 38, 40]}]}
20. {20: [{16: [11, 12, 15, 17, 20, 21], 19: [10, 15, 20, 24, 28, 39], 21: [16, 17, 20, 22, 25, 26, 30], 24: [19, 20, 28, 29], 25: [20, 21, 29, 30], 29: [20, 24, 25, 28, 30, 31]}]}
21. {21: [{16: [11, 12, 15, 17, 20, 21], 17: [12, 13, 16, 21, 22], 20: [16, 19, 21, 24, 25, 29], 22: [17, 18, 21, 23, 26, 27, 31], 25: [20, 21, 29, 30], 26: [21, 22, 27, 30, 31], 30: [21, 25, 26, 28, 29, 31, 32]}]}
22. {22: [{17: [12, 13, 16, 21, 22], 18: [13, 14, 22, 23], 21: [16, 17, 20, 22, 25, 26, 30], 23: [14, 18, 22, 27, 32, 35], 26: [21, 22, 27, 30, 31], 27: [22, 23, 26, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32]}]}
23. {23: [{14: [9, 13, 18, 23, 34], 18: [13, 14, 22, 23], 22: [17, 18, 21, 23, 26, 27, 31], 27: [22, 23, 26, 31, 32], 32: [23, 27, 28, 30, 31, 36], 35: [23, 34, 36]}]}
24. {24: [{19: [10, 15, 20, 24, 28, 39], 20: [16, 19, 21, 24, 25, 29], 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 24, 25, 28, 30, 31]}]}
25. {25: [{20: [16, 19, 21, 24, 25, 29], 21: [16, 17, 20, 22, 25, 26, 30], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26, 28, 29, 31, 32]}]}
26. {26: [{21: [16, 17, 20, 22, 25, 26, 30], 22: [17, 18, 21, 23, 26, 27, 31], 27: [22, 23, 26, 31, 32], 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32]}]}
27. {27: [{22: [17, 18, 21, 23, 26, 27, 31], 23: [14, 18, 22, 27, 32, 35], 26: [21, 22, 27, 30, 31], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 30, 31, 36]}]}
28. {28: [{19: [10, 15, 20, 24, 28, 39], 24: [19, 20, 28, 29], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32], 32: [23, 27, 28, 30, 31, 36], 40: [28, 39]}]}
29. {29: [{20: [16, 19, 21, 24, 25, 29], 24: [19, 20, 28, 29], 25: [20, 21, 29, 30], 28: [19, 24, 29, 30, 31, 32, 40], 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32]}]}
```

```
30. {30: [{21: [16, 17, 20, 22, 25, 26, 30], 25: [20, 21, 29, 30], 26: [21, 22, 27, 30, 31]
    , 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 24, 25, 28, 30, 31], 31: [22, 26, 27, 28,
    29, 30, 32], 32: [23, 27, 28, 30, 31, 36]}]}
31. {31: [{22: [17, 18, 21, 23, 26, 27, 31], 26: [21, 22, 27, 30, 31], 27: [22, 23, 26, 31,
     32], 28: [19, 24, 29, 30, 31, 32, 40], 29: [20, 24, 25, 28, 30, 31], 30: [21, 25, 26,
    28, 29, 31, 32], 32: [23, 27, 28, 30, 31, 36]}]}
32. {32: [{23: [14, 18, 22, 27, 32, 35], 27: [22, 23, 26, 31, 32], 28: [19, 24, 29, 30, 31,
     32, 40], 30: [21, 25, 26, 28, 29, 31, 32], 31: [22, 26, 27, 28, 29, 30, 32], 36: [32,
    35]}]}
33. {33: [{5: [3, 4, 9, 33], 34: [14, 33, 35]}]}
34. {34: [{14: [9, 13, 18, 23, 34], 33: [5, 34], 35: [23, 34, 36]}]}
35. {35: [{23: [14, 18, 22, 27, 32, 35], 34: [14, 33, 35], 36: [32, 35]}]}
36. {36: [{32: [23, 27, 28, 30, 31, 36], 35: [23, 34, 36]}]}
37. {37: [{1: [2, 3, 6, 10, 37], 38: [10, 37, 39]}]}
38. {38: [{10: [1, 6, 11, 15, 19, 38], 37: [1, 38], 39: [19, 38, 40]}]}
39. {39: [{19: [10, 15, 20, 24, 28, 39], 38: [10, 37, 39], 40: [28, 39]}]}
40. {40: [{28: [19, 24, 29, 30, 31, 32, 40], 39: [19, 38, 40]}]}
```