Integrative activity – test cases explanation

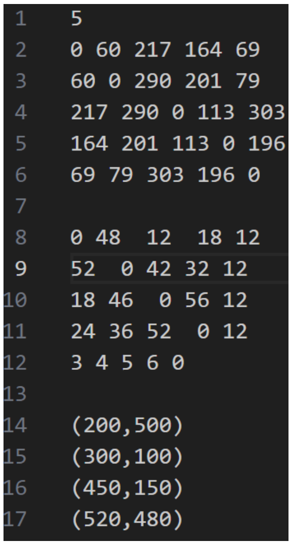
Analysis and Design of Advanced Algorithms

Group 601

Integrative activity – test cases explanation

Test case 1:

This first case test case is a standard input, it does not properly test for any edge cases, and theres nothing special about this test case, the purpose of applying this test case is to guarantee the proper functionalities of the code, and that we can guarantee the functinalities.



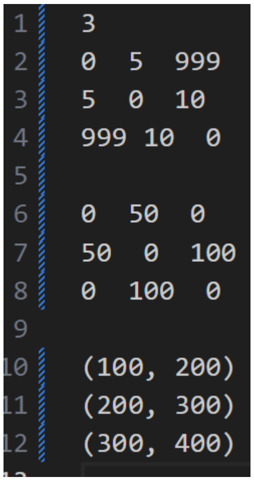
```
5
0 60 217 164 69
60 0 290 201 79
217 290 0 113 303
164 201 113 0 196
69 79 303 196 0

0 48  12  18 12
52  0 42 32 12
18 46  0 56 12
24 36 52  0 12
3 4 5 6 0

(200,500)
(300,100)
(450,150)
(520,480)
```

Test case 2:

This test case, while being a smaller input graph, it has edges with big numbers to simulate edges that are separated by a big margin number, and the flow input has some connections with 0 in order to test cases where no data flow is possible.



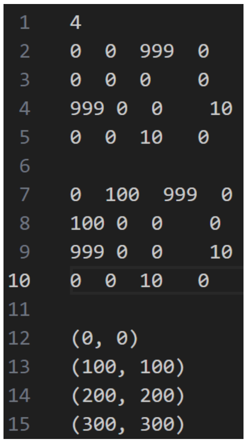
```
3
0  5  999
5  0  10
999 10  0

0  50  0
50  0  100
0  100  0

(100, 200)
(200, 300)
(300, 400)
```

Test case 3:

This test case includes zero values indicating no distance between some neighborhoods and a very large value to simulate almost disconnected neighborhoods, this case is similar to the previous input, however it is slightely bigger, in order to guarantee the best functionality. It is the same case with the input for the flow graph.



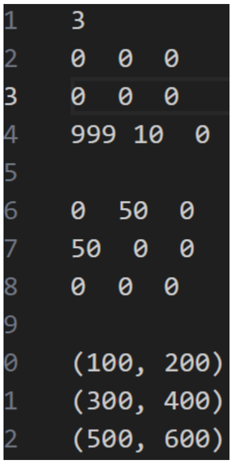
```
4
0  0  999  0
0  0  0    0
999 0  0     10
0  0  10   0

0  100  999  0
100 0  0     0
999 0  0     10
0  0  10   0

(0, 0)
(100, 100)
(200, 200)
(300, 300)
```

Test case 4:

This test case includes has very little connections between the nodes, in order to guarantee the cases where there exist almost no inputs, and the graph in the end is just a single connection simulating a linked list.



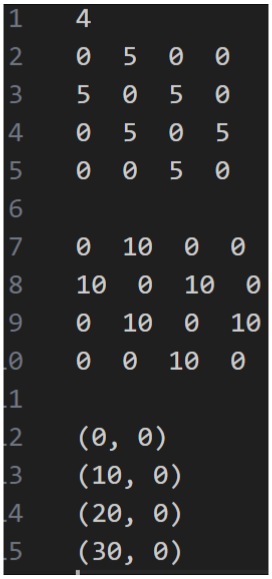
```
3
0  0  0
0  0  0
999 10  0

0  50  0
50  0  0
0  0  0

(100, 200)
(300, 400)
(500, 600)
```

Test case 5:

Similarly to the previous case, this test case has a slightly bigger input, but has a low amount of connections, in oirder to guarantee the functionality when the graph is a linked list.



```
4
0  5  0  0
5  0  5  0
0  5  0  5
0  0  5  0

0  10  0  0
10  0  10  0
0  10  0  10
0  0  10  0

(0, 0)
(10, 0)
(20, 0)
(30, 0)
```