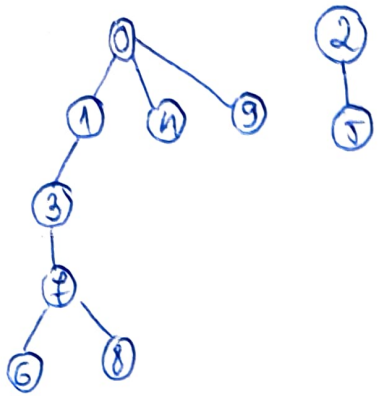


Bound dict

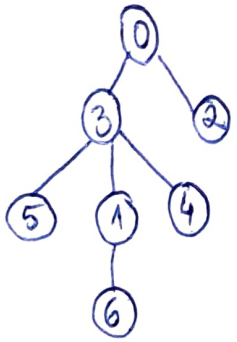
0	[1,4,9]
1	[0,3]
2	[5]
3	[1,7]
4	[0]
5	[2]
6	[7]
7	[3,6,8]
8	[7]
9	[0]
10	–

11 vertices and 8 edges

x	init queue	connect. comp	visited[x]	queue after run for x
call breath_first_traversal(0)				
0	[0]	{0}	true	[1,4,9]
1	[1,4,9]	{0,1}	true	[4,9,3] - 0 is visited
4	[4,9,3]	{0,1,4}	true	[9,3] - 0 is visited
9	[9,3]	{0,1,4,9}	true	[3] - 0 is visited
3	[3]	{0,1,4,9,3}	true	[7] - 1 is visited
7	[7]	{0,1,4,9,3,7}	true	[6,8] - 3 is visited
6	[6,8]	{0,1,4,9,3,7,6}	true	[8] - 7 is visited
8	[8]	{0,1,4,9,3,7,6,8}	true	[] - 7 is visited - queue is empty
call breath_first_traversal(2)				
2	[2]	{2}	true	[5]
5	[5]	{2,5}	true	[] - 2 is visited - queue is empty
call breath_first_traversal(10)				
10	[10]	{10}	true	[] - 10 is an isolated vertex, queue is empty

The connected components are: {0,1,4,9,3,7,6,8}
{2,5}
{10}

Find the connected components of an undirected graph using BFS



Bound dict

key	value
0	[3,2]
1	[2,5,3,4,6]
2	[1,4,0]
3	[0,5,1,4]
4	[5,6,2,1,3]
5	[4,3,1]
6	[4,1]

7 vertices and 12 edges

call breath_first_traversal(0)

x	queue init	connected comp	visited[x]	queue after run for x
0	[0]	{0}	true	[3,2]
3	[3,2]	{0,3}	true	[2, 5,1,4] - 0 is visited
2	[2, 5,1,4]	{0,3,2}	true	[5,1,4] - 1,4,0 are visited
5	[5,1,4]	{0,3,2,5}	true	[1,4] - 4,3,1 are is visited
1	[1,4]	{0,3,2,5,1}	true	[6] ,1,2,3,5 are visited
4	[4,6]	{0,3,2,5,1,4}	true	[6] - 5,2,1,3 are visited

~~because 1 and 4 are visited, for these values the queue we don't add for the next 2 values of the queue, we will add only 6 because the other vertices are visited, so the queue will be [6,6,6,6,6].~~

6	[6]	{0,3,2,5,1,4,6}	true	[ ] - all vertices are visited

~~For the remaining values, we will have the same result, so at the end the queue will be empty, the code is finished~~ → connected comp {0,3,2,5,1,4,6}

~~For the ot~~ All the vertices are visited, the algorithm is finished