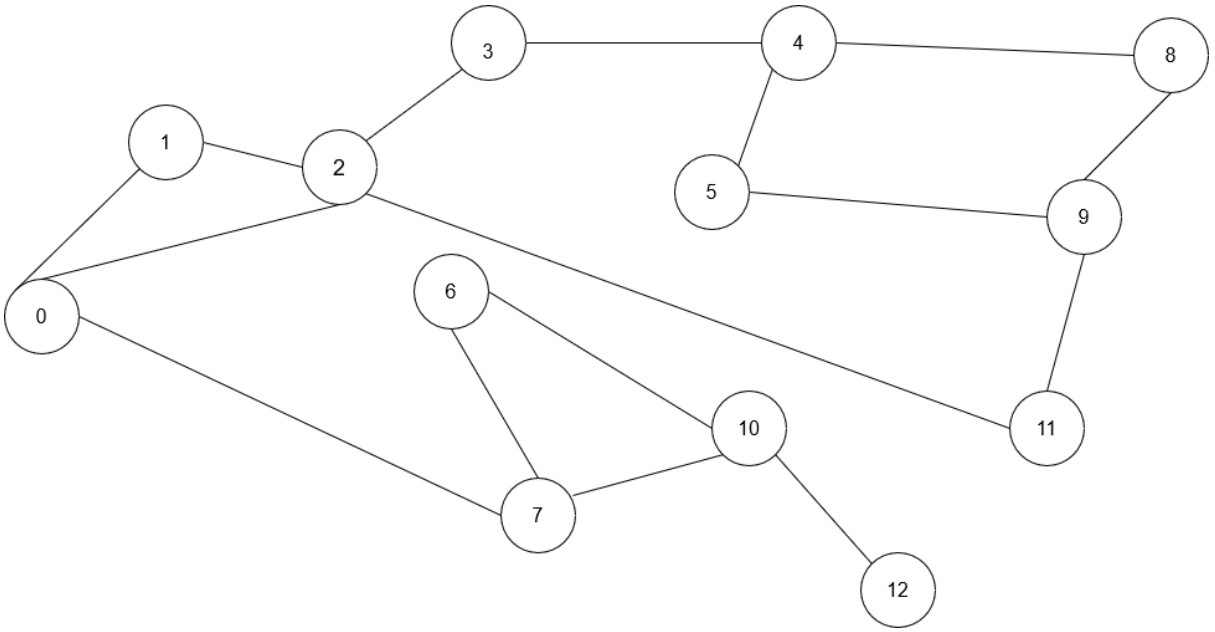
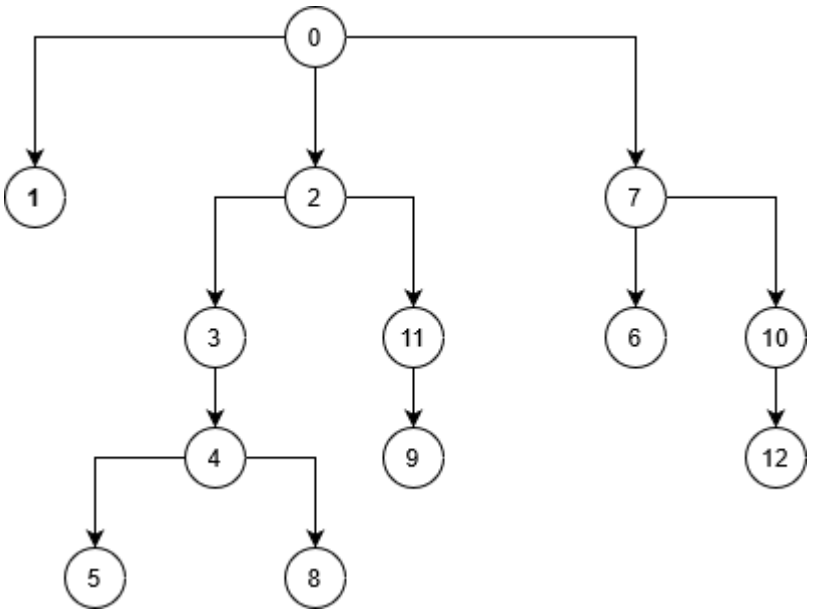
Задача 1.

Рассмотрим следующий граф.



Перед нами граф, на котором изображены 13 агентов и связи между ними.
При запуске программы каждый агент запоминает рандомное число от 20 до 40.

Произведем обход графа в ширину, начиная с вершины 0, чтобы составить алгоритм вычисления среднего арифметического.



Будем считать среднее арифметическое, поднимаясь по дереву обхода снизу вверх. Итоговое значение будет вычислено на агенте под номером “0” и напечатано на экран (== передано в “центр”).

Данное дерево можно представить с помощью следующей матрицей инцидентности:

-	0	1	2	3	4	5	6	7	8	9	10	11	12
0		1	1					1					
1													
2				1								1	
3					1								
4						1			1				
5													
6													
7							1				1		
8													
9													
10													1
11										1			
12													

A[i][j] = 1 ⇔ Агент i может получить сообщения от агента j.

Детали реализации:
При создании агента в качестве аргументов ему передается его рандомно сгенерированное число (от 20 до 40), количество сообщений, которое он должен принять от потомков перед передачей сообщения дальше, а также столбец из матрицы A, соответствующий ему.
Агент 0 также знает, сколько всего агентов находится в сети.
Каждый агент реализует два поведения: **SenderBehaviour** и **ListenerBehaviour**.

ListenerBehaviour: агент слушает входящие сообщения и при получении нового инкрементирует счетчик принятых сообщений и прибавляет к своему числу то, которое было получено в сообщении. Когда агент принял все сообщения от своих потомков, он передает сообщение со своим числом своему родителю. Если у агента нет потомков, то число передается сразу (без ожидания).

У **ZeroHelloAgent** также есть поведение **ZeroSenderBehaviour**, в котором происходит печать среднего арифметического на экран, когда до агента доходят все сообщения.

Эффективность:

Количество агентов	Невязка	Количество сообщений		Число тактов*	Память
		между агентами	в центр		
13	0	12	1	4	13 * (13 + 1 + 1 + 1) + 1 = 209 (ячейка памяти**)

**Считаем, что принятие/отправка сообщения и подсчет суммы происходит на один такт*

*** Считаем память для массивов связей, чисел, количества полученных сообщений, количества сообщений, которые нужно получить перед следующей отправкой.*