Вариант 1.

Входные данные: Неориентированный граф G=(V, E)

Выходные данные: True если граф G является связным и False иначе

Алгоритм (полином. дет. МТ)

Алгоритм поиска в глубину DFS(G) -- найти для своего варианта

Для всех u\in V

Visited[u]=False // все вершины не просмотрены

Для всех u\in V

Если Visited[u]=False то

Visited[u]=True

count=+1

Explore(u)

Для всех (u, w)\in E

Если Visited[w]=False

Visited[w]=True

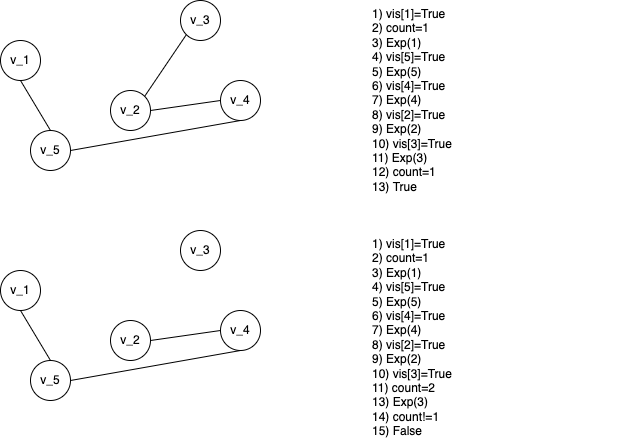
Explore(w)

Если count!=1 то

Вернуть False

Иначе

Вернуть True



Сложность алгоритма

1) Цикл по всем вершинам (установить значение False) -- O(n)

2) Цикл по всем вершинам \* цикл по всем ребрам -- O(n\*n\*(n-1)/2)

3) Общее время (по числу итераций) -- O(n+n\*n\*(n-1)/2)=O(n^3)

Как должно выглядеть сравнение теоретической и практической временной сложности?

Вход 1 -- граф из 10 вершин: теоретическая сложность O(1000), практическая сложность 10 сек

Вход 2 -- граф из 50 вершин: теоретическая сложность O(125000), практическая сложность 20 сек

Вывод -- при увеличении размера входа в 5 раз, практическая временная сложность возрастает в 2 раза