29. Топологическая сортировка ориентированного ациклического графа: определение и алгоритм поиска (с доказательством корректности).

Топологическая сортировка ориентированного ациклического графа $G(V, E)$ представляет собой упорядочивание вершин таким образом, что для любого ребра $(u, v) \in E$ номер вершины u меньше номера вершины v.

Предположим, что граф ацикличен, т.е. решение существует. Что делает обход в глубину? При запуске из какой-то вершины v он пытается запуститься вдоль всех рёбер, исходящих из v. Вдоль тех рёбер, концы которых уже были посещены ранее, он не проходит, а вдоль всех остальных — проходит и вызывает себя от их концов.

Таким образом, к моменту выхода из вызова $DFS(v)$ все вершины, достижимые из v как непосредственно (по одному ребру), так и косвенно (по пути) — все такие вершины уже посещены обходом. Следовательно, если мы будем в момент выхода из $DFS(v)$ добавлять нашу вершину в начало некоего списка, то в конце концов в этом списке получится топологическая сортировка.

```
for (int i = 0; i < n; ++i)
    dfs(i);
// вывести вершины в порядке убывания tout
```

Корректность: ▲ Покажем, что если $(u, v) \in E$, то u выведется раньше v. Рассмотрим 2 случая

1. v была замечена DFSом раньше чем u. Тогда к моменту выхода из v u все еще не посещена, иначе нашелся бы цикл $\Rightarrow tout[v] < tout[u]$

2. u была замечена раньше чем v. Тогда мы перейдем по ребру (u, v) в v, а до выхода из u необходимо выйти из всех вершин, в которые мы попали из нее (по лемме о белых путях) $\Rightarrow tout[v] < tout[u]$ ■

30. Отношение сильной связности между вершинами. Компоненты сильной связности. Сильно связный граф.

Определение: В ориентированном графе вершины u, v - *сильно связаны*, если из u есть путь в v и из v есть путь в u.

Утверждение: сильная связность является отношением эквивалентности

▲ Рефлексивность и симметричность очевидны из определения. Транзитивность получается из того что просто склеиваем пути ■

Определение: Классы эквивалентности относительно отношения сильной связности, на которые разбивается граф, называются *компонентами сильной связности*.

Определение: Ориентированный граф *сильно связен*, если для произвольной вершины все остальные вершины достижимы из нее.

31. Алгоритм Косарайю. Корректность и время работы.

1. Строим граф H с инвертированными ребрами

2. Выполним DFS на H, вычисляющий для каждой вершины время выхода DFS из нее (этот массив обозначим за f).

3. Выполняем DFS на исходном графе, перебирая вершины в порядке убывания $f(u)$.

▲ Докажем, что вершины s, t взаимно достижимы тогда и только тогда, когда после выполнения алгоритма они принадлежат одному дереву обхода.

$\Rightarrow$

Если вершины s, t были взаимно достижимы в графе G, то на третьем этапе будет найден путь из одной вершины в другую, это означает, что по окончанию алгоритма обе вершины лежат в одном поддереве.

$\Leftarrow$

1. Вершины s, t лежат в одном и том же дереве поиска в глубину на третьем этапе алгоритма. Значит, что они обе достижимы из корня r этого дерева.

2. Вершина r была рассмотрена вторым обходом в глубину раньше, чем s и t, значит время выхода из нее при первом обходе в глубину (по обратным ребрам) больше, чем время выхода из вершин s и t. Из этого мы получаем 2 случая:

   - Обе эти вершины были достижимы из r в H. А это означает взаимную достижимость вершин s, r и взаимную достижимость вершин r, t (так как они достижимы из r и по прямым и по обратным ребрам). А складывая пути мы получаем взаимную достижимость вершин s, t.
   - Хотя бы одна не достижима из r в H, например t. Значит и r была не достижима из t в H, так как время выхода r больше (если была бы достижима, было бы меньше). Значит между этими вершинами нет пути, но последнего быть не может, потому что t была достижима из r по пункту 1.

Значит s, t достижимы в обоих графах. ■