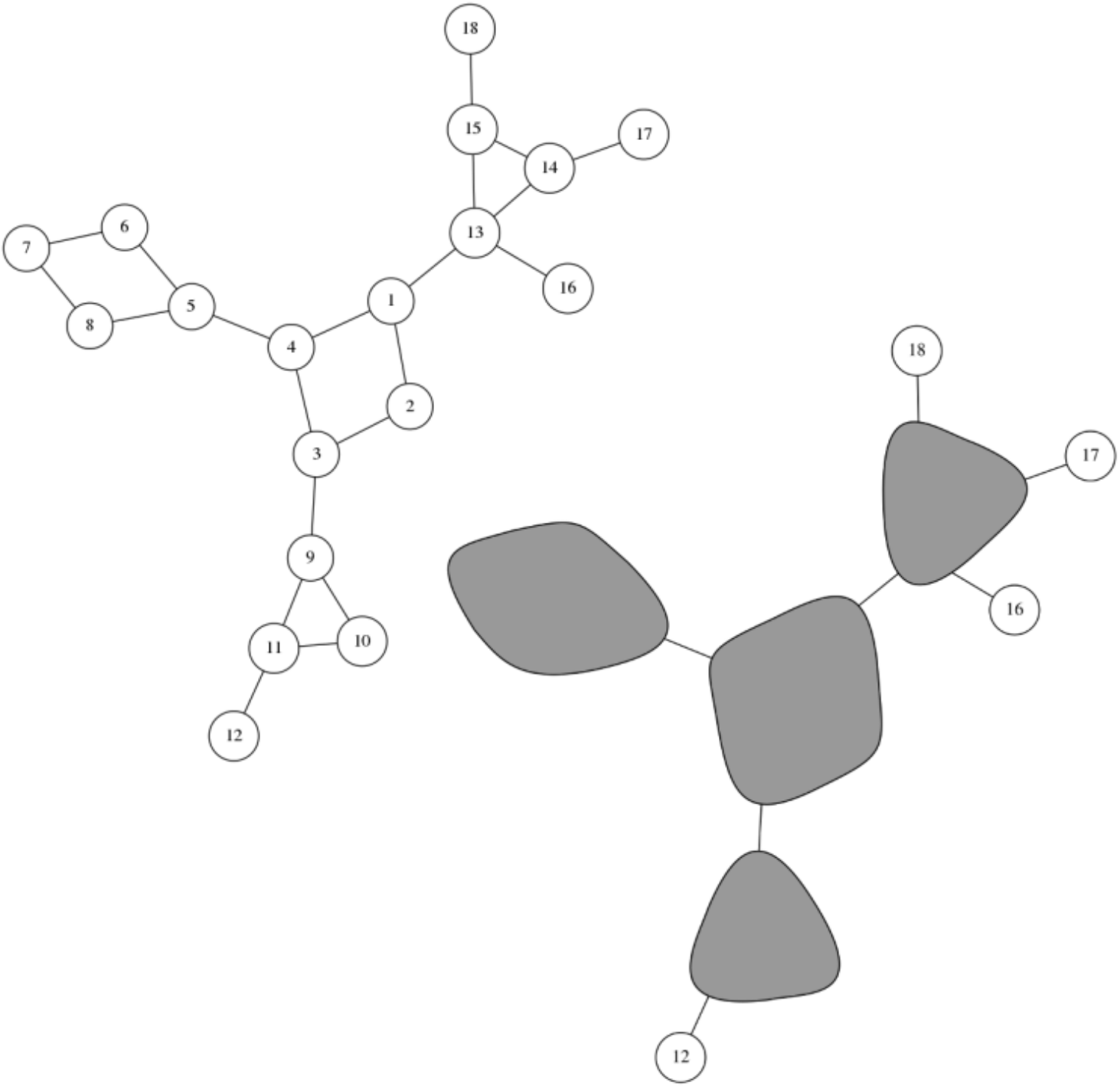
E. Cactus

Аналитично решение на задачата

https://codeforces.com/contest/231/problem/E

В тази задача трябва да намерим броя на простите пътища между два върха в граф, който е *върхов кактус*. Върхов кактус е такъв грав, в който всеки квърх участва в най-много един цикъл. След като проучим структурата на върховия кактус, ще установим, че ако *обединим/смачкаме* всеки цикъл в един връх, ще получим дърво. Следователно, нека обединим всеки цикъл в един връх от първоначално подадения граф и получим това дърво. Също така, всеки връх в това дърво ще отбележим с 1, ако е смачкан цикъл или 0, ако не е участвал в нито един цикъл.



Как ще смачкаме циклите? Ще използваме алгоритъма на Tarjan, за намиране на обратни ребра. Всеки път когато процесираме върху непосетен връх, ще го добавяме в стек. Когато достигнем връх, на който всички деца са посетени и започнем да развиваме рекурсията ще проверяваме дали от този връх е излизало ребро което е сочело към някой негов предшественик в dfs дървото. Ако това не е изпълнено то tin[u] == low[u], т.е. момента на влизането в него ще е равен на дълбочината му. Ако пък това е изпълнено, ще запаметяваме върха на стека и ще изкарваме от него върхове, който ще са от един цикъл, докато не стигнем върха който сме запаметили. (Има и по интуитивен начин като използваме Disjoint Set Union структурата).

Сега, за да намерим броя на пътищата от връх a до връх b в първоначалния граф, ще направим следното: Допускаме, че c е връх, който съответства на a в полученото дърво (той може да е връх, който съответства на единичен връх или на смачкан цикъл съдържащ върха a) и d е връх, който съответсвта на върха b. Нека с deg отбележим броя на върховете отбелязани с 1 в дървото в пътя между c и d. Лесно е да се види, че отговора на заявката за броя на пътищата между тези два върха c и d ще е $2^{deg} \mod 10^9 + 7$, тъй като всеки цикъл (връх маркиран с 1) увеличава броя на възможностите за пътища - двойно (може да преминем от един връх до друг връх по два начина в цикъла).

Това автоматично означава, че трябва да преброим върховете маркирани с 1 по пътя от един връх до друг в дървото и то бързо, за да може да отговаряме на заявки. Може да го направим по следния начин: закачаме дървото в някакъв произволен връх, който ще наречем корен. За всеки връх v ще дефинираме променлива cnt_v броя на върховете маркирани с 1 в пътя от него до корена на дървото (включително върха и корена). Да допуснем например, че искаме да намерим броя на върховете маркирани с 1 по пътя между a и b. Нека c е най-ниския общ прародител на a и b. Тогава този търсен брой на върховете маркирани с 1 е равен на $cnt_a + cnt_b - 2 \cdot cnt_c$, ако върха c е маркиран с 0 или $cnt_a + cnt_b - 2 \cdot cnt_c + 1$, ако c е връх маркиран с 1. Най-ниския общ прародител (least common ancestor LCA) може да бъде намерен чрез няколко похвата.

LCA: https://cp-algorithms.com/graph/lca.html

В нашето решение ще използваме възможно най-бързия алгоритъм за намиране на LCA за заявки - ще си построим разредена таблица (Sparse Table), върху масива от върхове, който се образува като пуснем dfs (ойлеров път) от даден връх в редуцирания граф (който е дърво).