# Башни 2.0

В первой подзадаче можно было явно построить граф, где ребро $(u, v)$ проведено если из $u$-й башни можно прыгнуть на $v$-ю. После этого достаточно из каждой вершины запустить DFS и посчитать количество посещённых вершин. Пусть $m$ – количество рёбер в графе, тогда $n$ DFS-ов работют за $O(nm)$. Поскольку $m = \Theta(n^2)$ (нижняя оценка когда высоты башен возрастают), алгоритм работает за $O(n^3)$ и проходит при ограничениях $n \leqslant 100$.

Для решения второй подзадачи мы можем для каждой вершины хранить битсет достижимых вершин. Более формально, битсет вершины $v$ имеет размер $n$, и $i$-й бит равен 1 если вершина $i$ достижима из вершины $v$, 0 иначе. Сначала научимся эффективно строить граф. Заметим, что ребро $(u, v)$ существует тогда и только тогда, когда $v$-я башня является одним из максимумов на отрезке, в котором $u$ — середина, а $v$ является одним из концов. Мы можем предпосчитать максимумы на каждом отрезке за $O(n^2)$, после этого можно проверить существование ребра за $O(1)$. Поскольку все башни имеют разную высоту, если мы будем рассматривать их в порядке убывания высоты, все рёбра из очередной башни будут вести в предыдущие. Это значит, мы можем поддерживать инвариант что множество достижимых из всех рассмотренных вершин посчитано корректно, после чего для вычисления множества достижимых из очередной вершины $v$ достаточно взять OR битсетов вершин, куда ведут рёбра из $v$. Асимптотика такого решения — $O\left(\frac{n^3}{w}\right)$, где $w$ — длина машинного слова.

В третьей подзадаче все высоты равны либо 1, либо 2. Из любой вершины существует рёбра во все вершины со значением 2, поэтому их колчиество можно добавить к ответу для всех вершин. Кроме этого, нам нужно посчитать для каждой вершины $v$ со значением 1 количество других вершин со значением 1, которые из неё достижимы. Можно заметить, что если рядом с $v$ в массиве есть значение 2, мы не сможем попасть из $v$ в другую вершину со значением 1. В противном случае, мы можем, используя рёбра длины 1, добраться до всех вершин в максимальном по включению отрезке из 1, содержащем $v$, и ни одна другая вершина со значением 1 не достижима. Используя эти наблюдения, можно получить линейное решение.

Четвёртая подзадача была идентична третьей, но у нас дополнительно могли быть башни высотой 3. Сначала заметим, что все такие башни достижимы из всех остальных. Для каждой башни высотой 2 предпосчитаем следующую башню высоты 2 слева и справа, а также ближайшую башню высоты 3. Пусть башни $u$ и $v$ — башни высоты 2, и существует ребро из $u$ в $v$. Пусть $w_1, w_2, \ldots, w_k$ — башни высоты 2 на отрезке от $u$ до $v$. Заметим, что вместо ребра от $u$ до $v$, мы могли бы пойти по пути $u \to w_1 \to w_2 \to \ldots \to w_k \to v$. Это значит, что из всех рёбер между башнями высоты 2 можно оставить только те, которые ведут в ближайшую слева и справа башню, которые мы предпосчитали. Это позволяет легко посчитать для каждой вершины $v$ высоты 2 значения $l_v$ — количество достижимых слева, и $r_v$ — справа, а также итоговый ответ. Теперь рассмотрим башни $v$ высоты 1. Используя критерий из предыдущей подзадачи, мы можем понять какое множество башен высоты 1 достижимо — это может быть либо только исходная башня, либо максимальный по включению отрезок башен высоты 1, обозначим его за $[a, b]$. Нам осталось посчитать количество достижимых башен высоты 2. Заметим, что в любом пути из $v$ первая башня высоты больше 1 должна быть либо $a - 1$-й, либо $b + 1$-й. Тогда достаточно добавить к ответу $l_{a-1}$ если вершина $a - 1$ достижма и $r_{b+1}$ если вершина $b + 1$ достижима. Решение работает за $O(n)$.

Сделаем наблюдение, полезное для полного решения. Мы хотим понять из каких вершин есть ребро в вершину $v$. Пусть $gl_v$ — ближайший элемент в массиве слева от $v$, больший $v$, $gr_v$ — аналогичный элемент справа. Если к некоторой вершине $u$ есть более близкий чем $v$ элемент, больший $v$-го, его можно выбрать из множества $\{gl_v, gr_v\}$. Это значит, что множество вершин, из которых проведено ребро в $v$ — отрезок $\left[\left\lfloor\frac{gl_v+v}{2}\right\rfloor + 1, \left\lceil\frac{gr_v+v}{2}\right\rceil - 1\right]$.

Используя этот факт, можно получить решение пятой подзадачи. Пусть в вершину $v$ ведут рёбра с отрезка $[c_v, d_v]$. Мы переберём $v$, посчитаем отрезок вершин, из которых достижима $v$, и добавим 1 к ответу на этом отрезке. Чтобы посчитать отрезок вершин, откуда достижима $v$, будем хранить текущий ответ (изначально это $[c_v, d_v]$), и отрезок рассмотренных вершин (изначально это $[v, v]$). После этого, пока отрезок рассмотренных вершин и ответ не совпадут, будем расширять отрезок рассмотренных вершин на 1 (слева или справа). Пусть $u$ — вершина, которую мы только что добавили в отрезок рассмотренных, она должна принадлежать отрезку ответа, и мы можем объединить

отрезок ответа с отрезком $[c_u, d_u]$. Когда отрезок ответа и рассмотренных вершин совпадут, не будет существовать вершин вне отрезка, из которой есть ребро в отрезок, следовательно, это все вершины, из которых достижима $v$. Решение работает за $O(n^2)$.

Для полного решения понадобится ещё несколько наблюдений. Заметим, что поскольку ребро $(i, j)$ проведено тогда, когда $j$-й элемент является максимумом на отрезке, где $i$ является серединой, а $j$ – одним из концов, из наличия ребра $(i, j)$ следует наличие рёбер $(k, j)$ для $i < k < j$. Аналогичное утвреждение можно сделать для ребра $(i, j)$ когда $j < i$. Пусть в некотором пути есть два соседних ребра $(u, v)$ и $(v, w)$, такие, что $w < u < v$. Тогда, используя свойство выше, мы получаем, что есть ребро $(u, w)$. Делая такие замены для произвольного пути, можно добиться того, чтобы все рёбра пути вели либо только налево, либо только направо. Посчитаем кол-во достижимых по рёбрам налево из каждой вершины, аналогично по рёбрам направо, и, сложив результаты, получим ответ. Пусть $L_i$ – самая правая вершина левее $i$, в которую есть ребро из $i$. Докажем, что можно оставить только рёбра $i \to L_i$, и количество достижимых слева не изменится. Действительно, пусть в каком-то пути мы пошли по ребру $i \to j$ и $j < L_i$. Мы знаем что тогда должно быть ребро $L_i \to j$, и $i \to j$ можно заменить на $i \to L_i$, $L_i \to j$. Делая такие замены, мы можем удалить все «плохие» рёбра. Тогда ответ для вершины $i$ — это ответ для $L_i$ плюс 1. Это значит, что для решения задачи достаточно уметь эффективно находить $L_i$ для каждого $i$.

Мы умеем за линейное время посчитать $gl_i$ и $gr_i$ для всех $i$, используя стандратный алгоритм со стеком. Поскольку входящие рёбра в каждую вершину ведут из отрезка, который мы можем вычислить, задача о подсчёте $L_i$ сводится к следующей: даны $n$ отрезков $[l_i, r_i]$. Для каждой позиции нужно вычислить минимальный индекс отрезка, который её покрывает. Существует множество решений этой задачи. Мы можем сделать сканлайн, который поддерживает set отрезков, которые покрывают данную позицию, тогда для каждого отрезка нужно создать событие «добавить $i$» в позции $l_i$ и «удалить $i$» в позиции $r_i + 1$. Такое решение работает $O(n \log n)$. Также можно решать задачу с помощью дерева отрезков (тоже за $O(n \log n)$) или с помощью системы непересекающихся множеств (за $O(n \cdot \alpha(n))$).

**BONUS:** попробуйте решить задачу «Башни 3.0» из 9-го класса. У вас есть $q$ запросов $(u, v, l, r)$, необходимо отвечать количество вершин на отрезке $[l, r]$, которые достижимы как из $u$, так и из $v$.