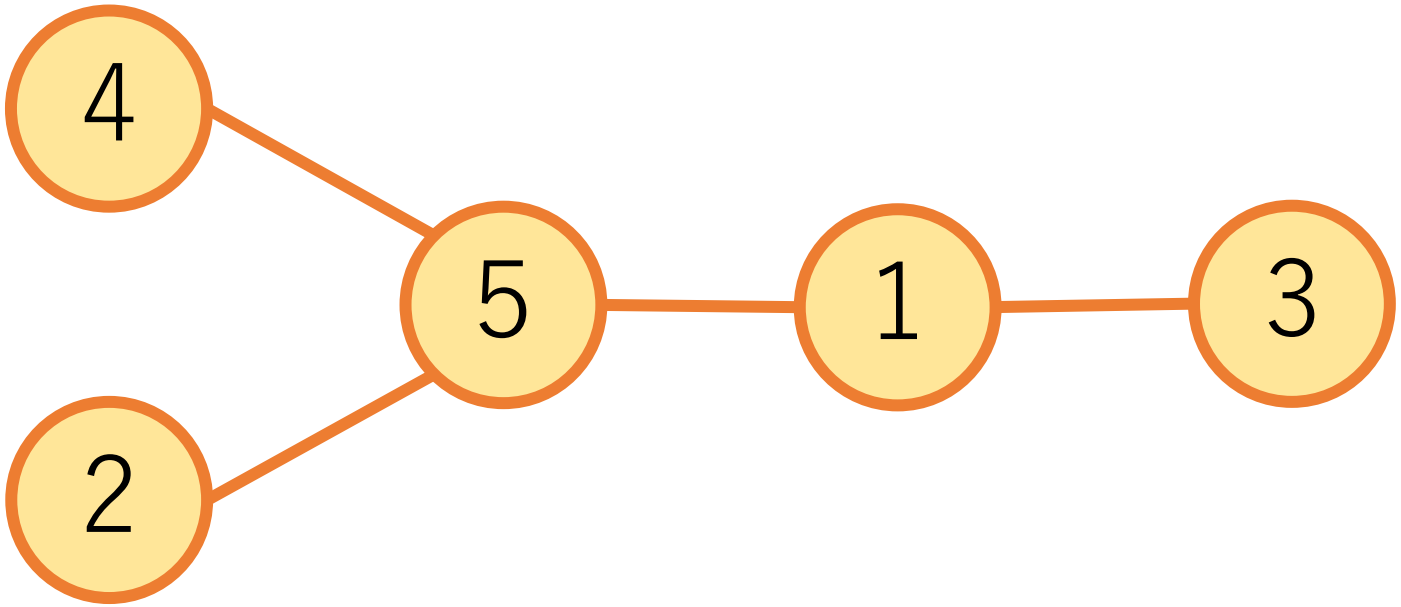
本選4 Cat Exercise

米山　瑛士 (ynymxiaolongbao)

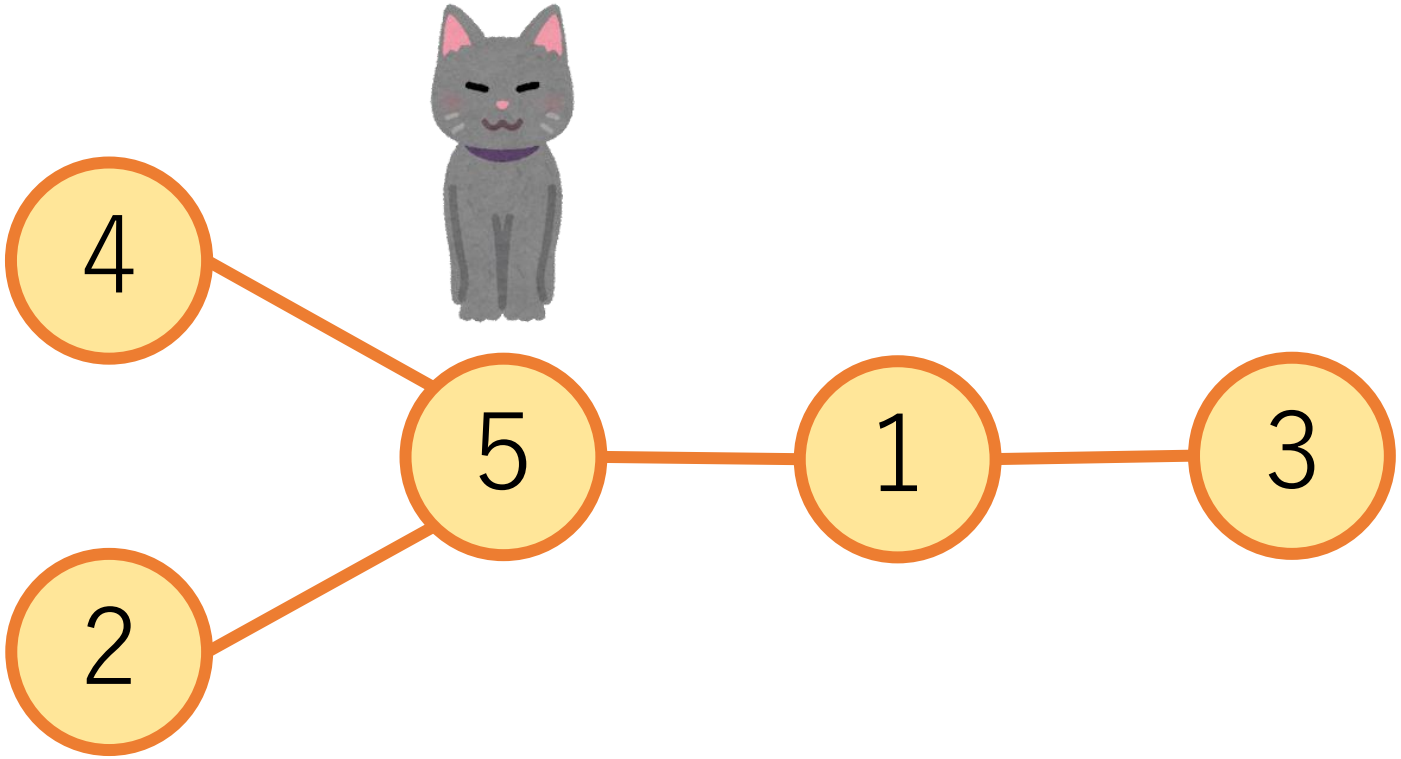
問題概要

- 木がある
- 各頂点は高さP[i]



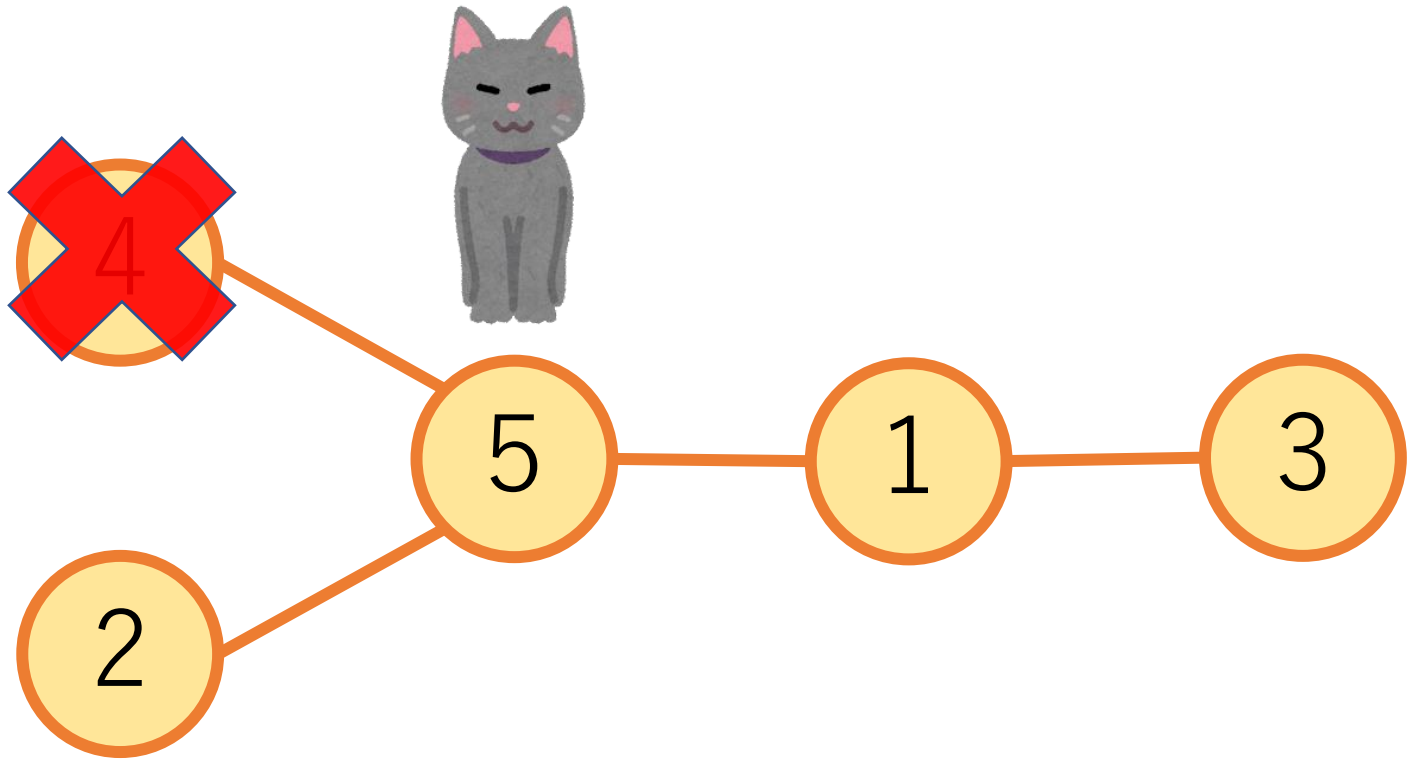
問題概要

- 猫がいる
- はじめ、最も高い頂点にいる



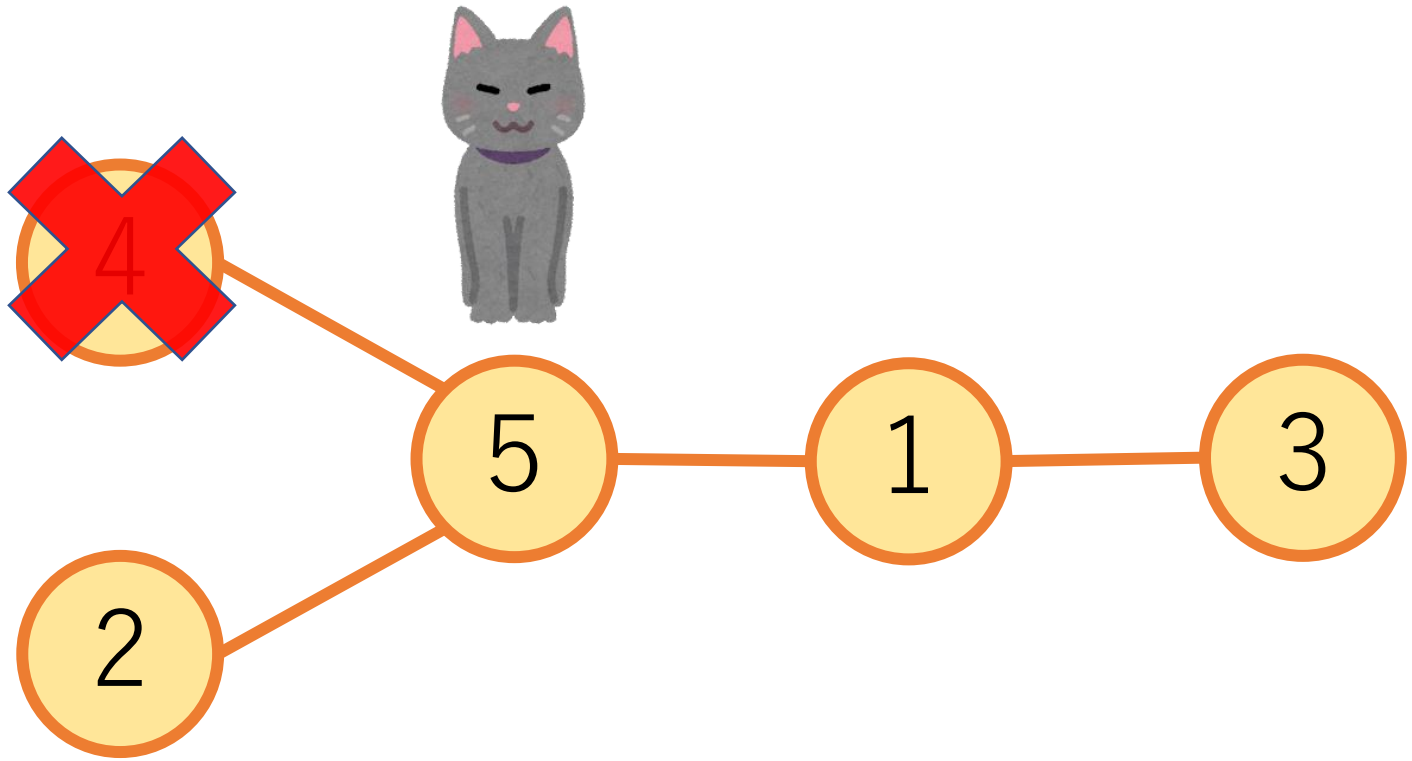
問題概要

- 一つの頂点を選んで障害物を置く



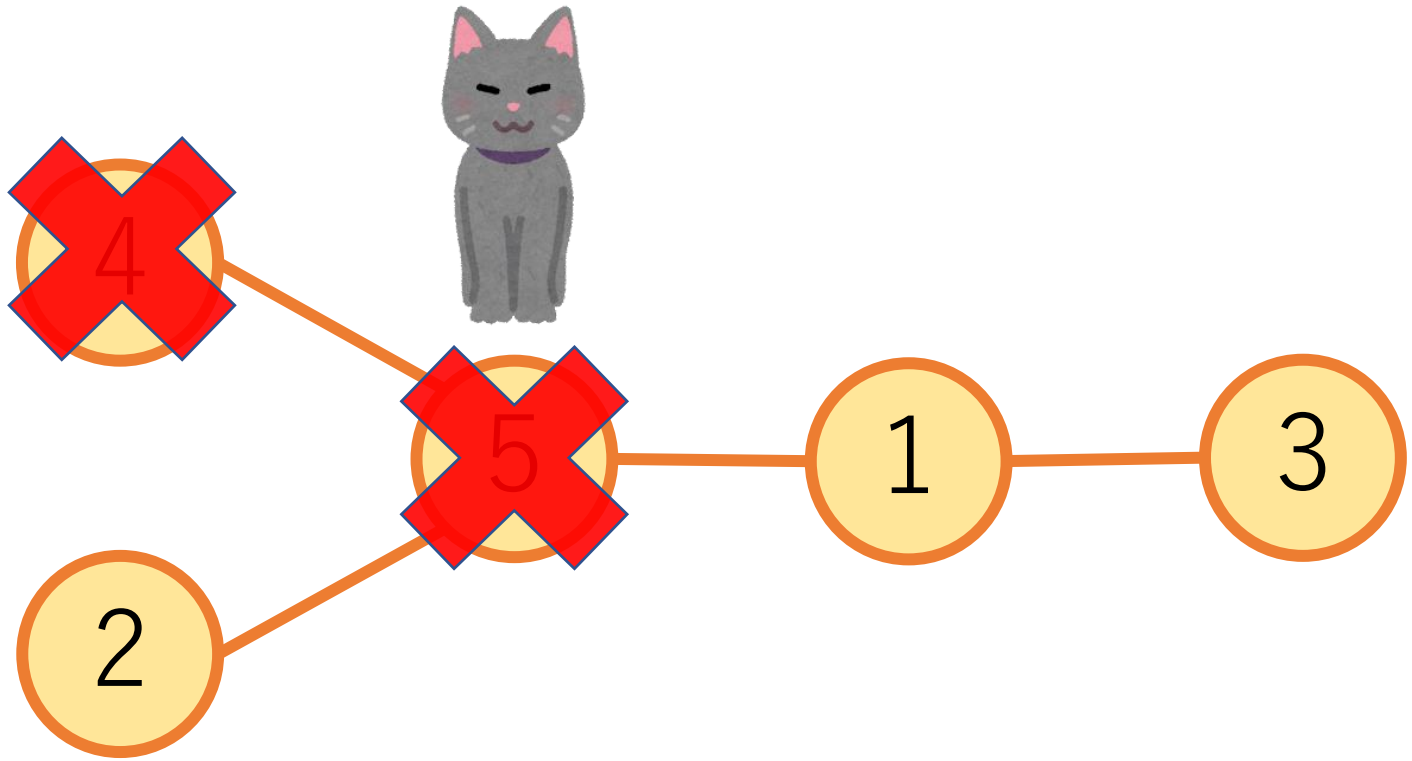
問題概要

- 選んだ頂点に猫がいなければ、何も起こらない



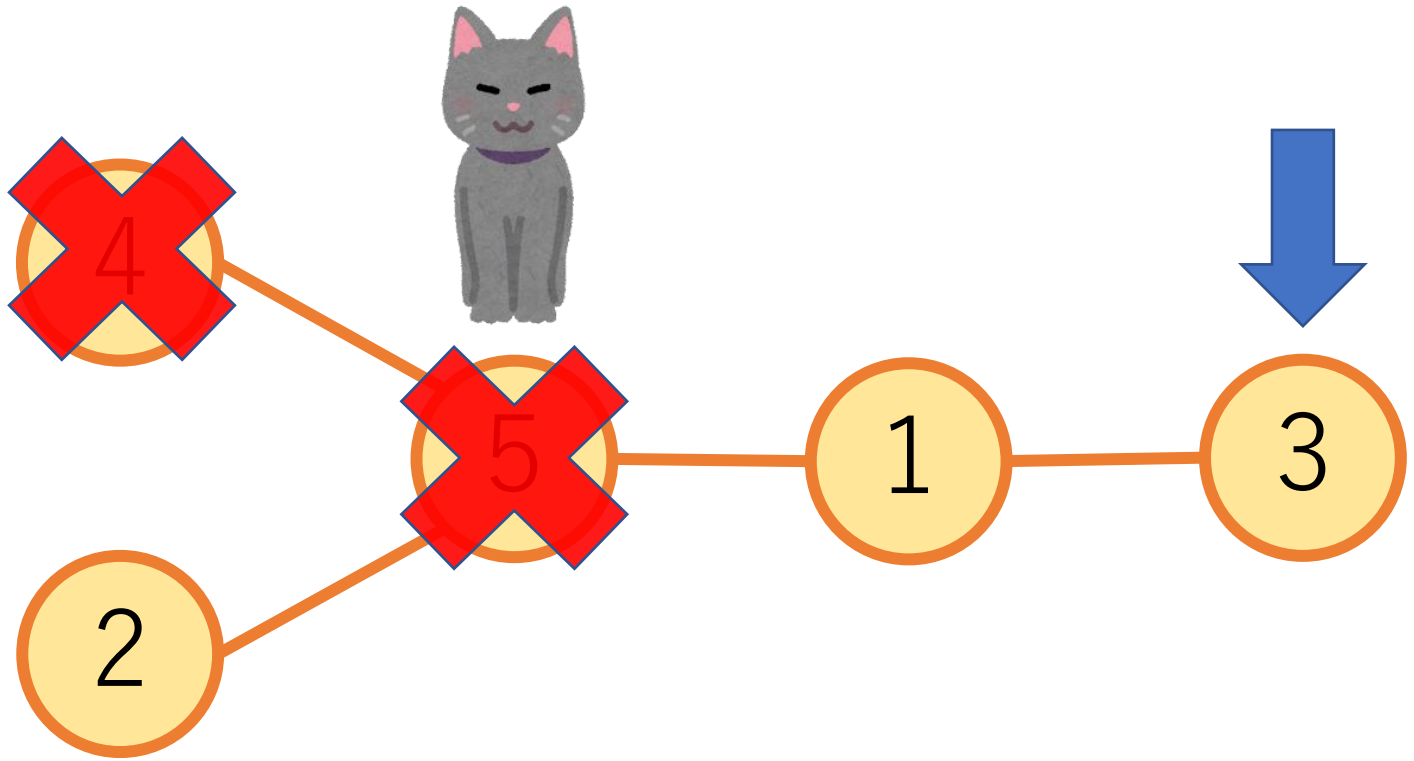
問題概要

- （再）一つの頂点を選んで障害物を置く



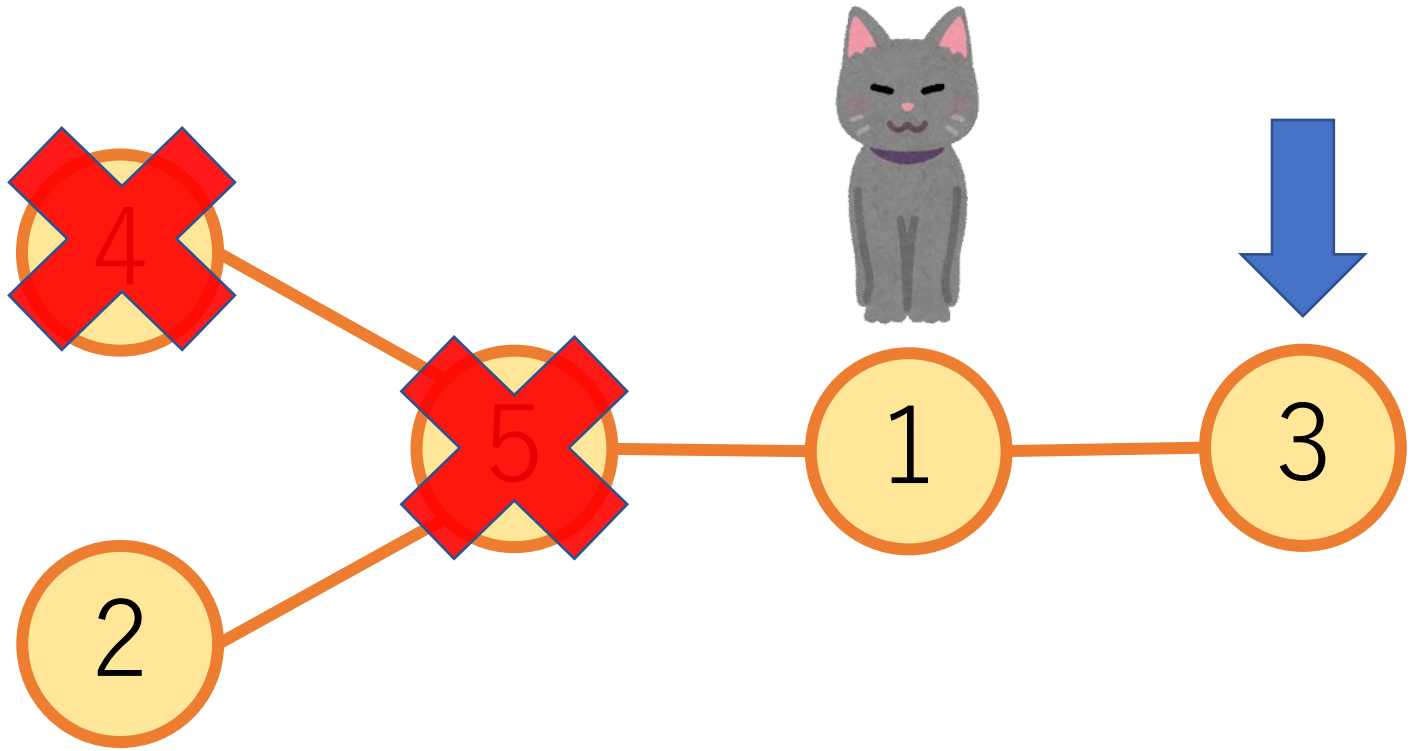
問題概要

- 選んだ頂点に猫がいた場合、その頂点から到達できる最も高い頂点に猫が移動する



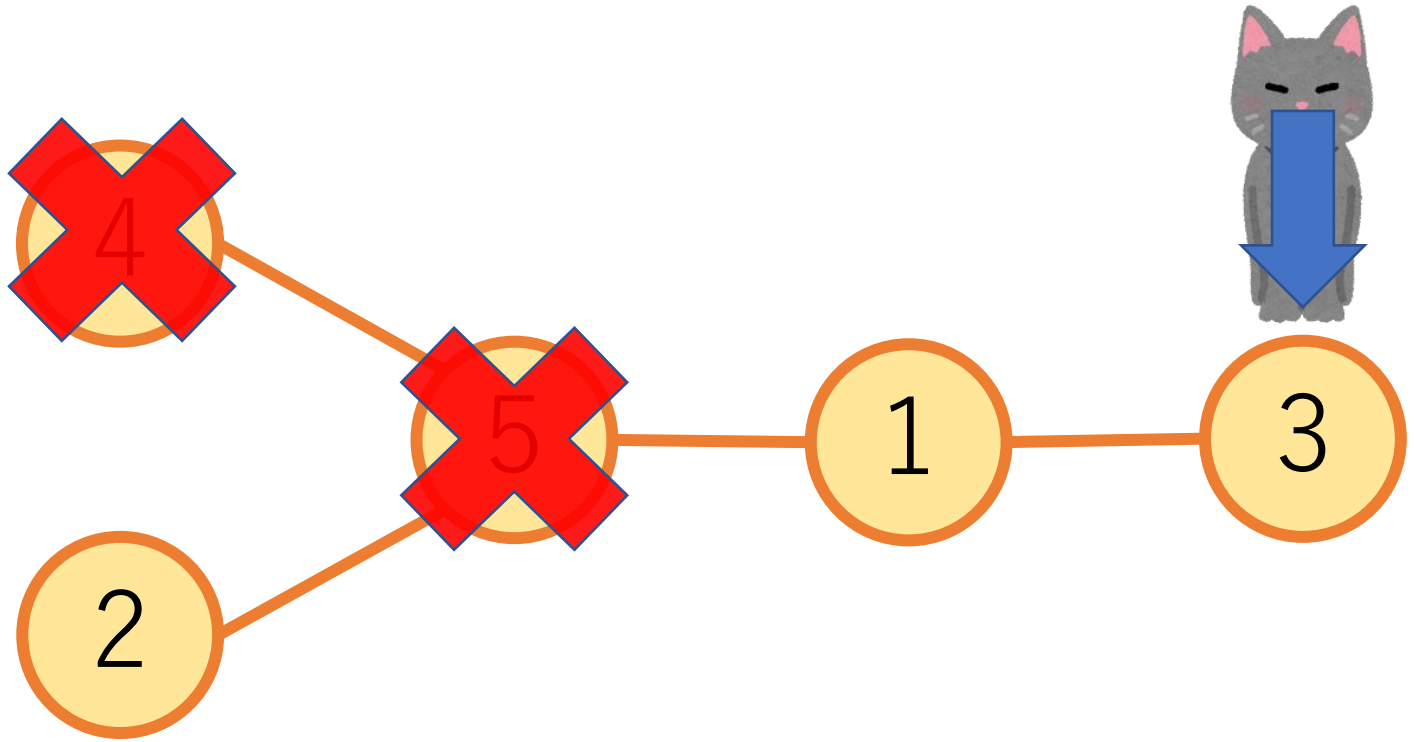
問題概要

- 選んだ頂点に猫がいた場合、その頂点から到達できる最も高い頂点に猫が移動する



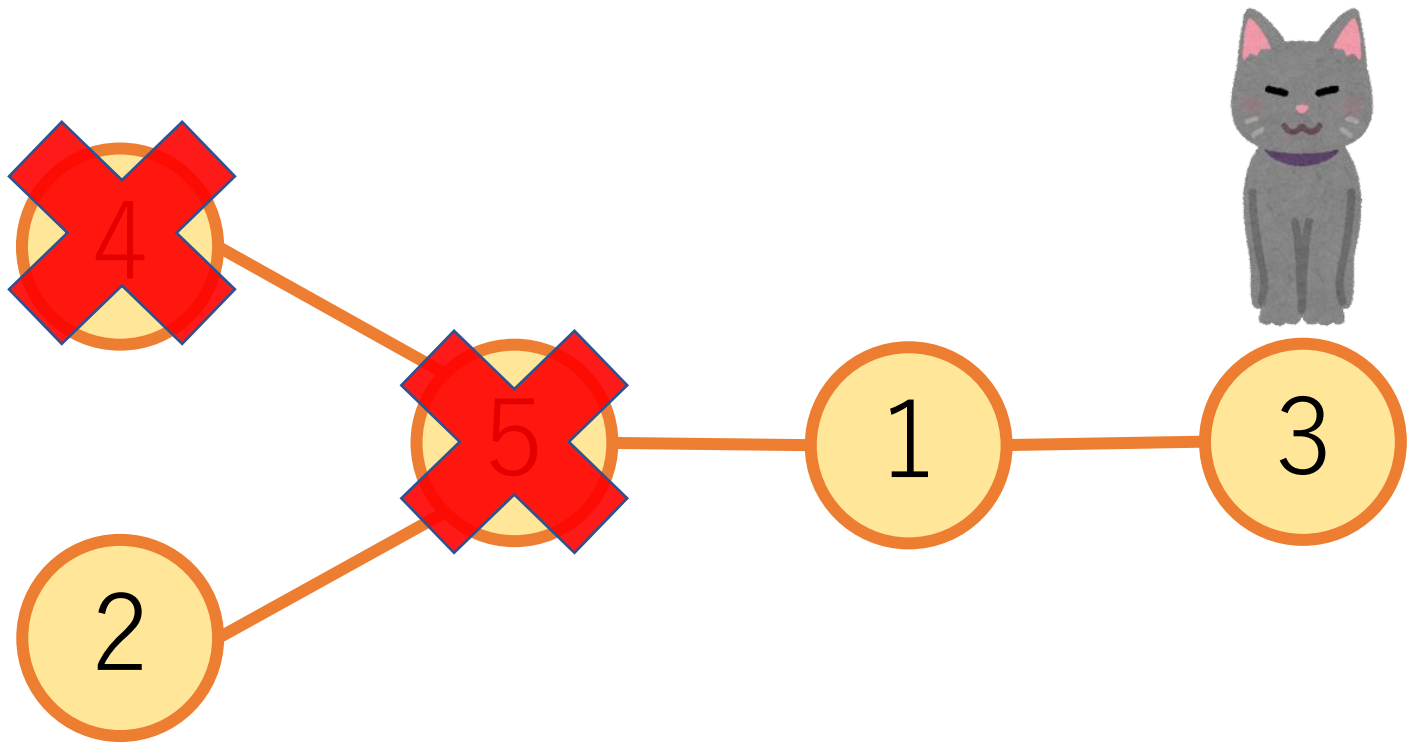
問題概要

- 選んだ頂点に猫がいた場合、その頂点から到達できる最も高い頂点に猫が移動する



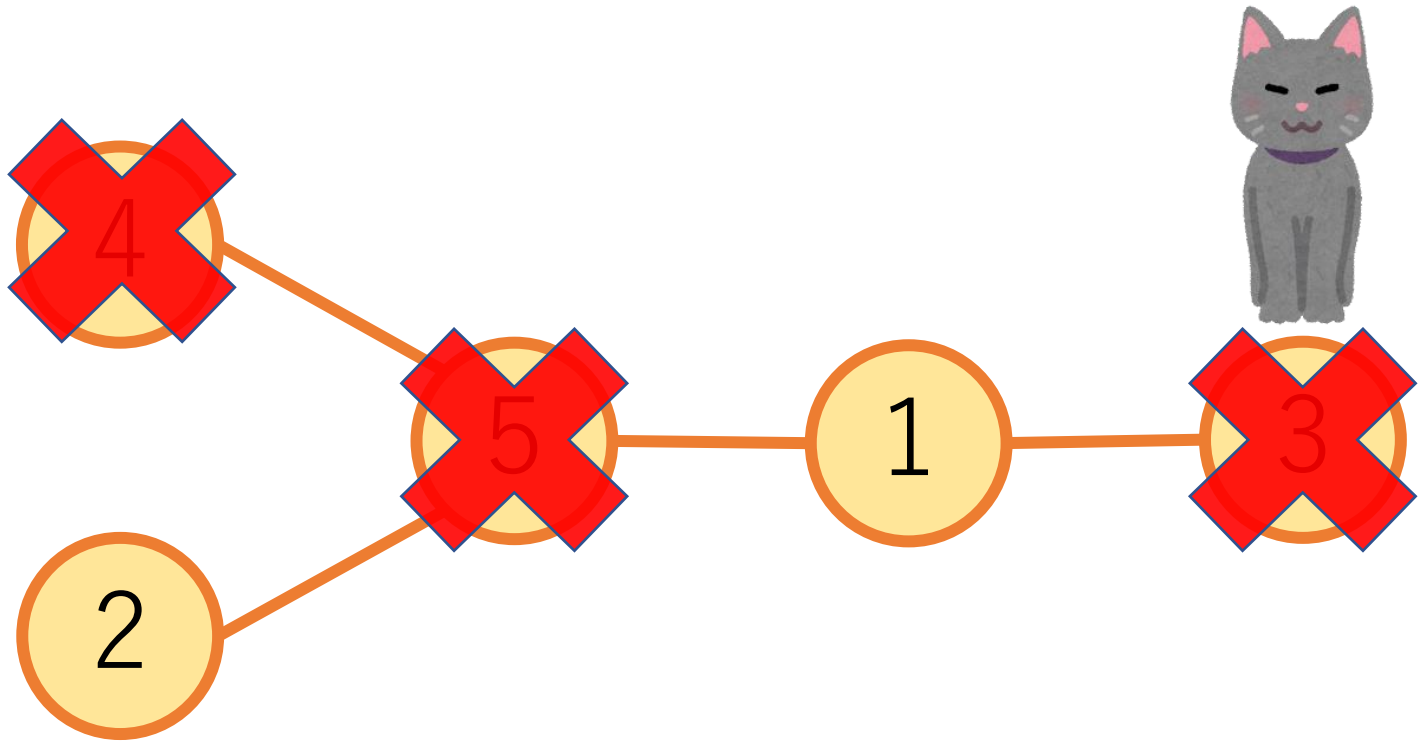
問題概要

- 猫は距離 2 だけ移動した



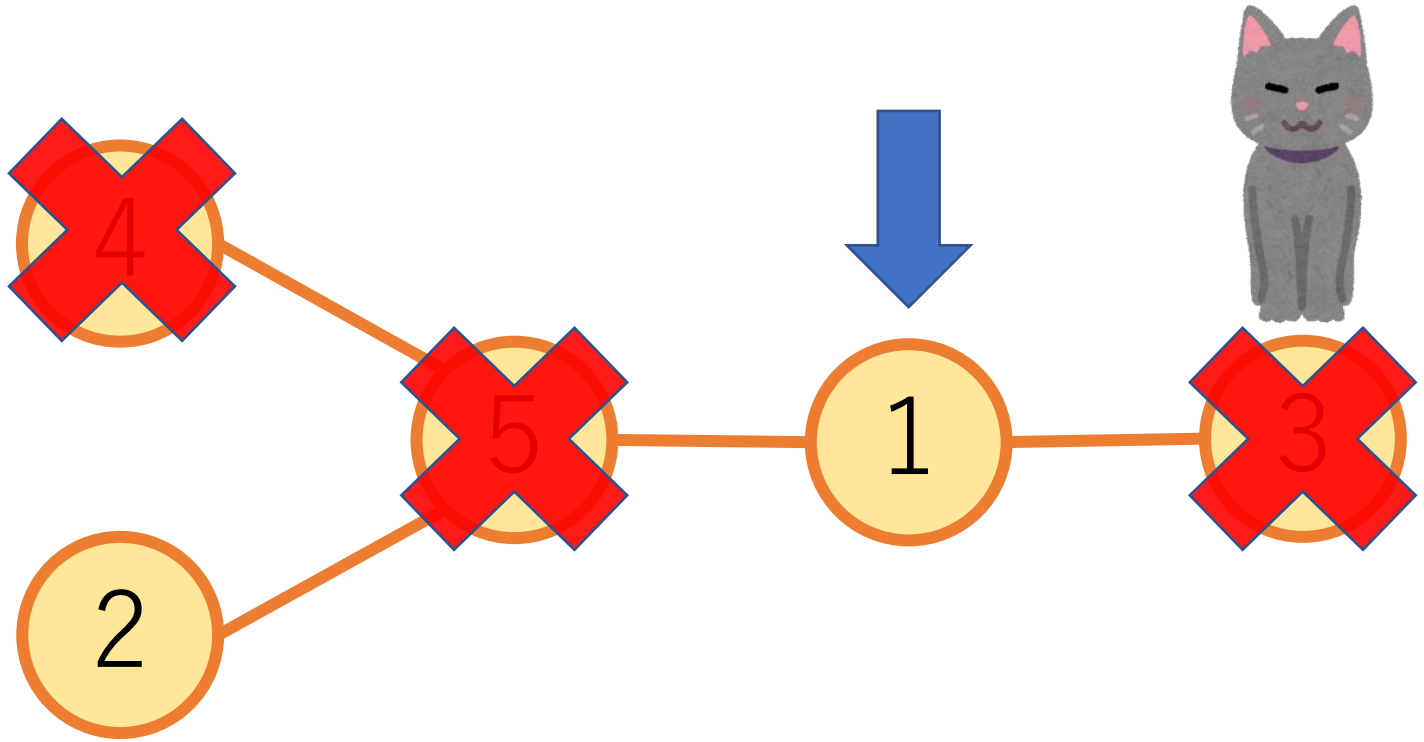
問題概要

- （再）一つの頂点を選んで障害物を置く



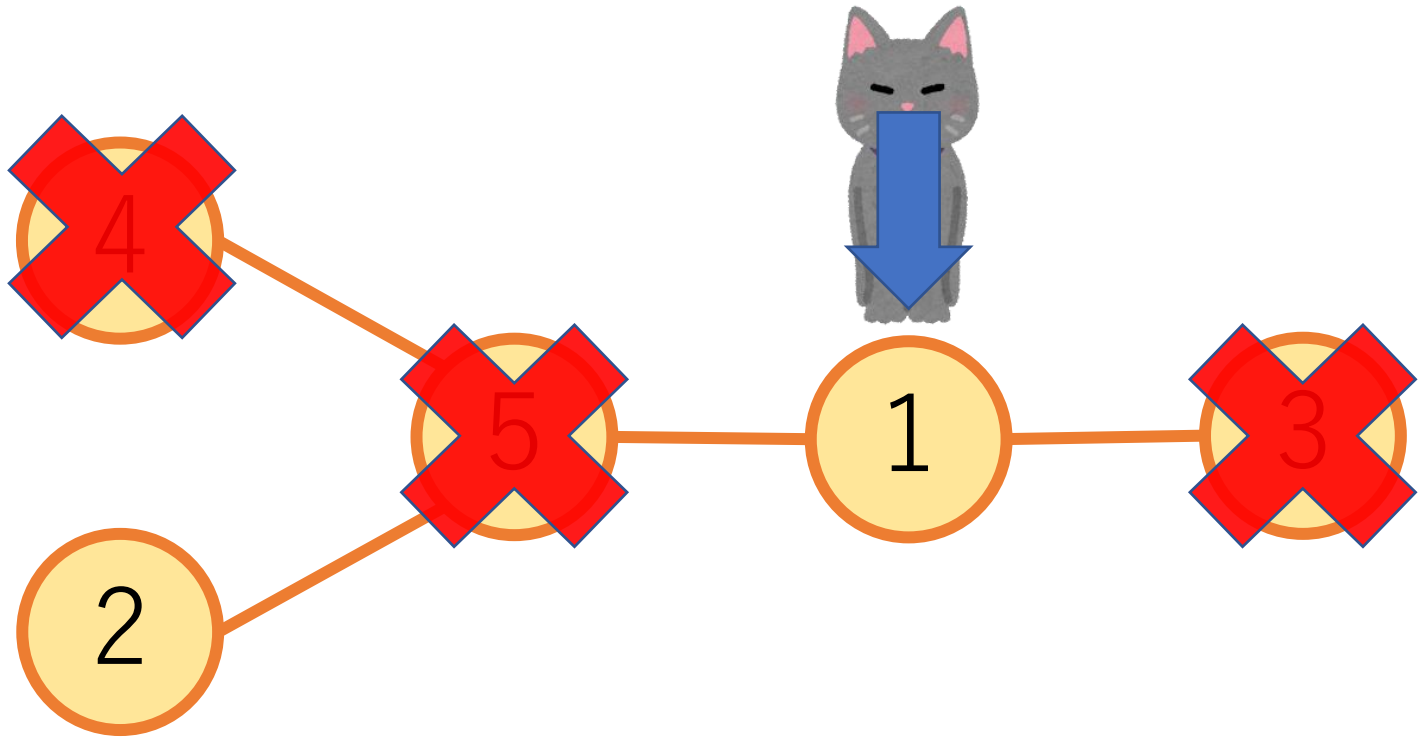
問題概要

- （再）選んだ頂点に猫がいた場合、その頂点から到達できる最も高い頂点に猫が移動する



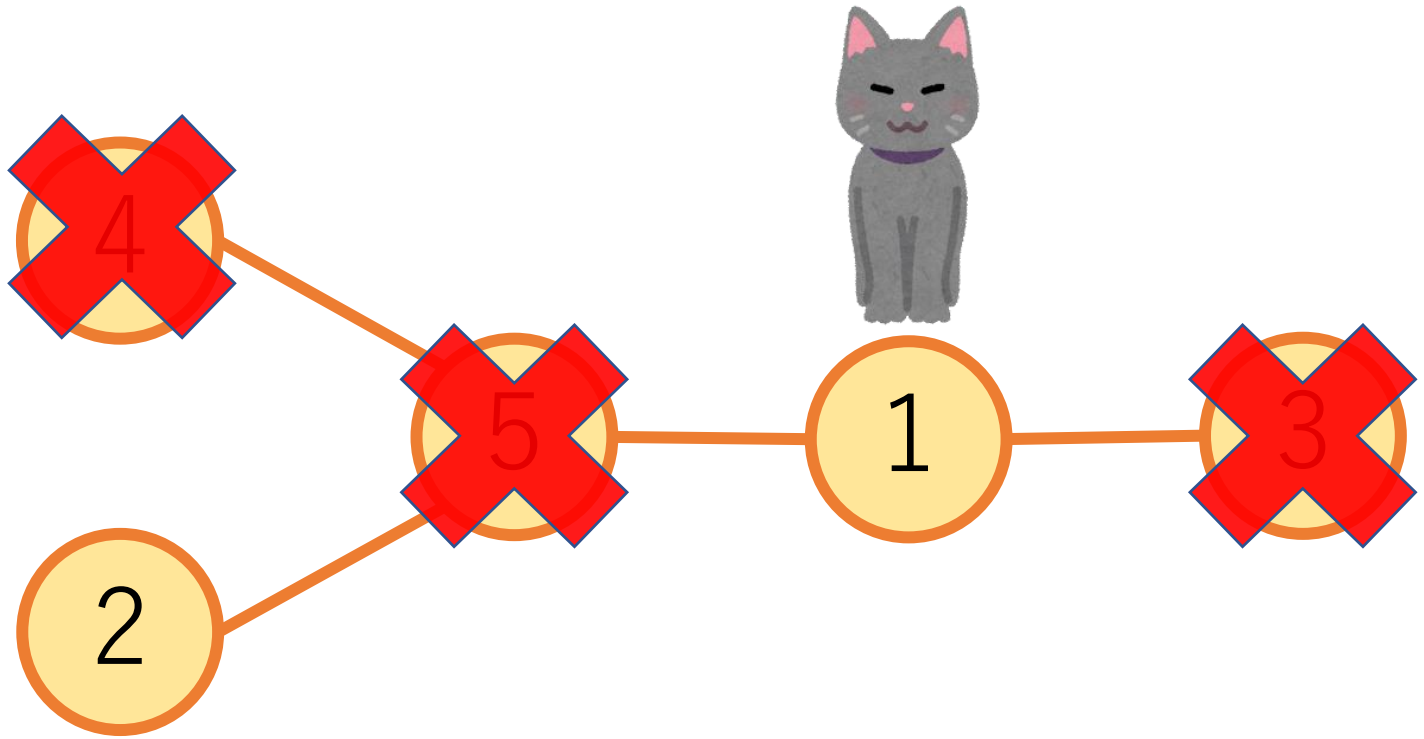
問題概要

- （再）選んだ頂点に猫がいた場合、その頂点から到達できる最も高い頂点に猫が移動する



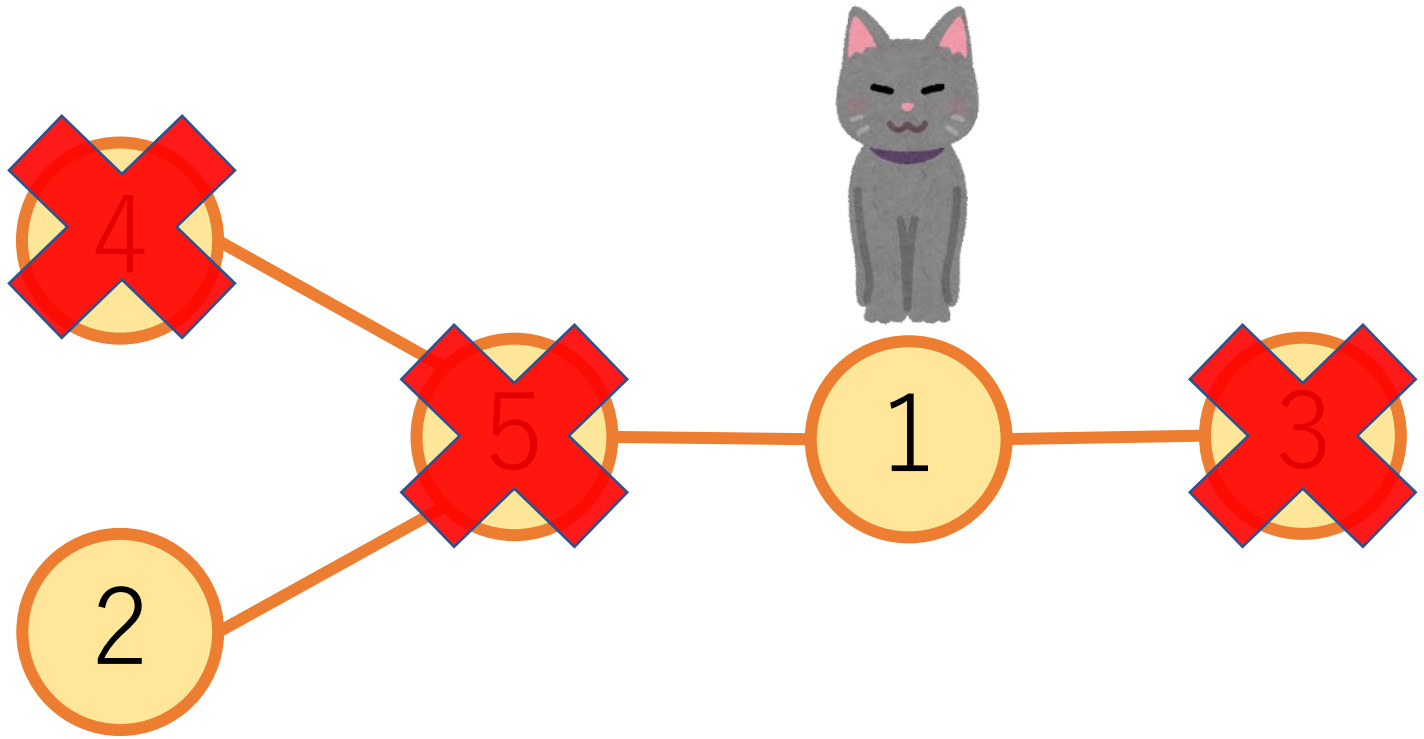
問題概要

- 猫は合計距離 3 だけ移動した



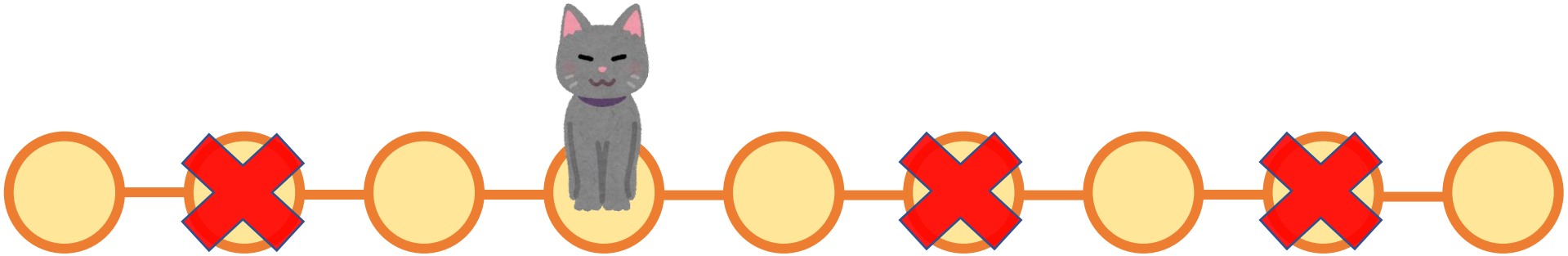
問題概要

- 頂点を選ぶ順番を工夫して、猫の移動距離の総和を最大化する



小課題１(N<=16, パス)

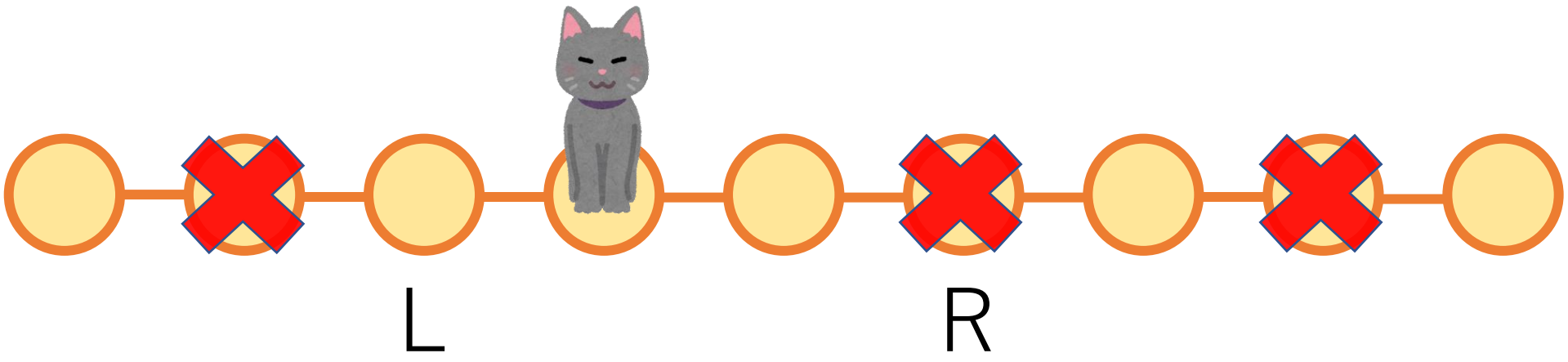
- bitDP
- dp[S][x]:=頂点集合Sに障害物があり、猫が頂点xにいるときの、これからの移動距離の最大値
- O(2^N*poly(N))



小課題 2 (N<=300, パス)

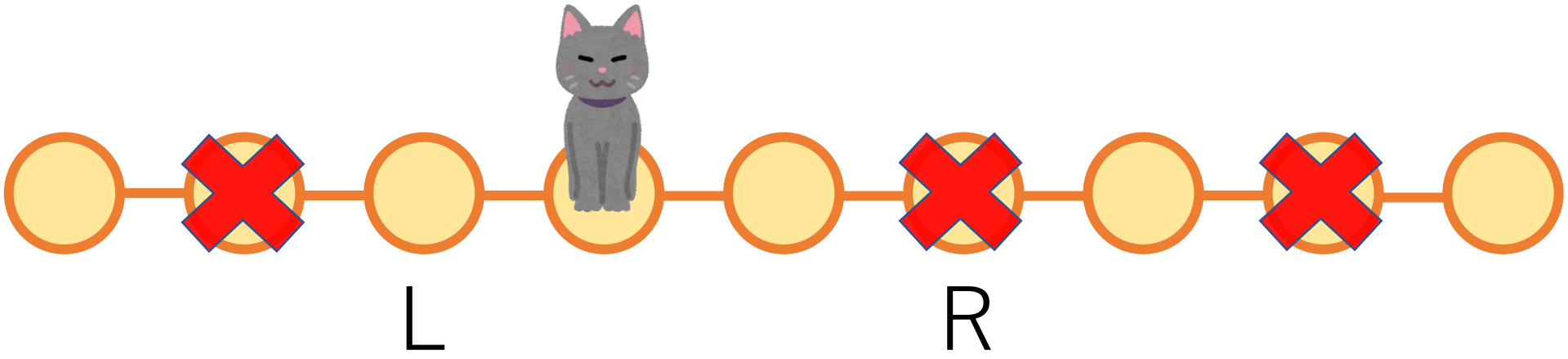
- もう少し情報を絞る
- 猫がいる区間の外でどう障害物が置かれているかは関係ない
- 猫は区間内で最も高い場所にいる

→猫がいる区間だけで状態を区別



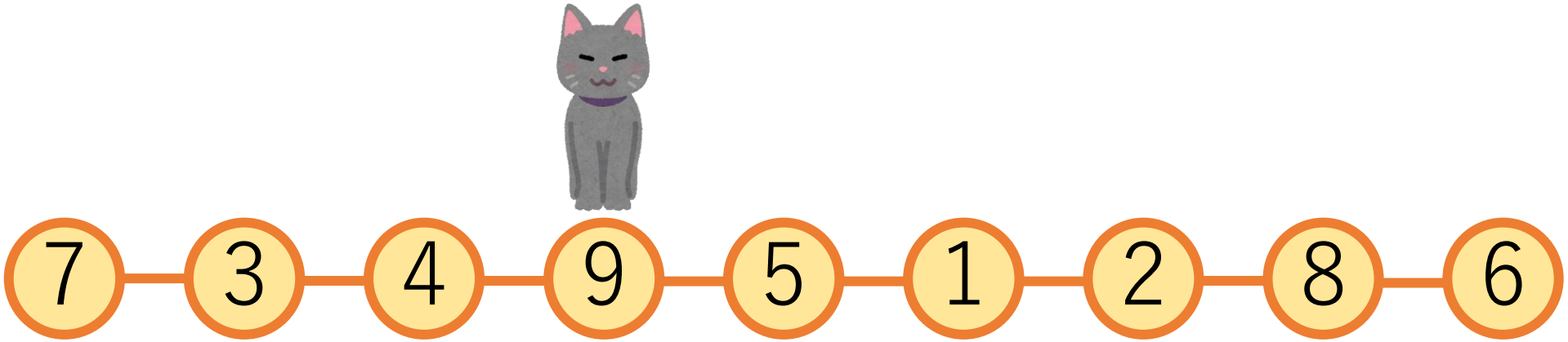
小課題 2 (N<=300, パス)

- 区間DP
- dp[L][R]:=猫がいる区間が[L,R)のときの，これからの移動距離の最大値
- 遷移：mid (L<=mid<R)に障害物を置く
- O(N^3)



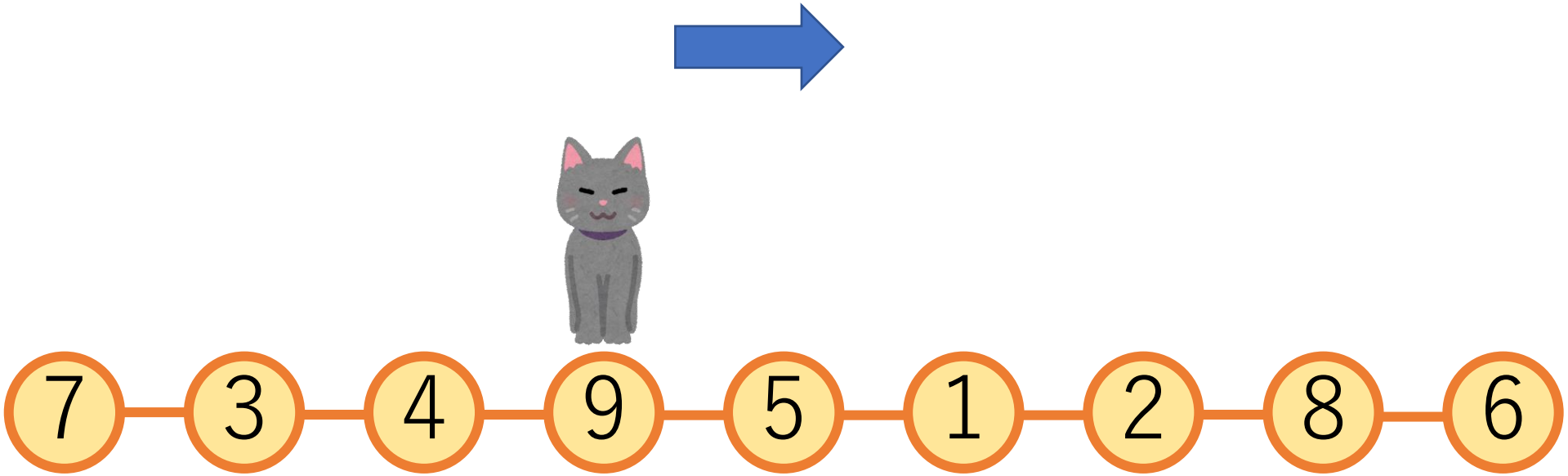
小課題3 (N<=5000, パス)

- 戦略を立てたい



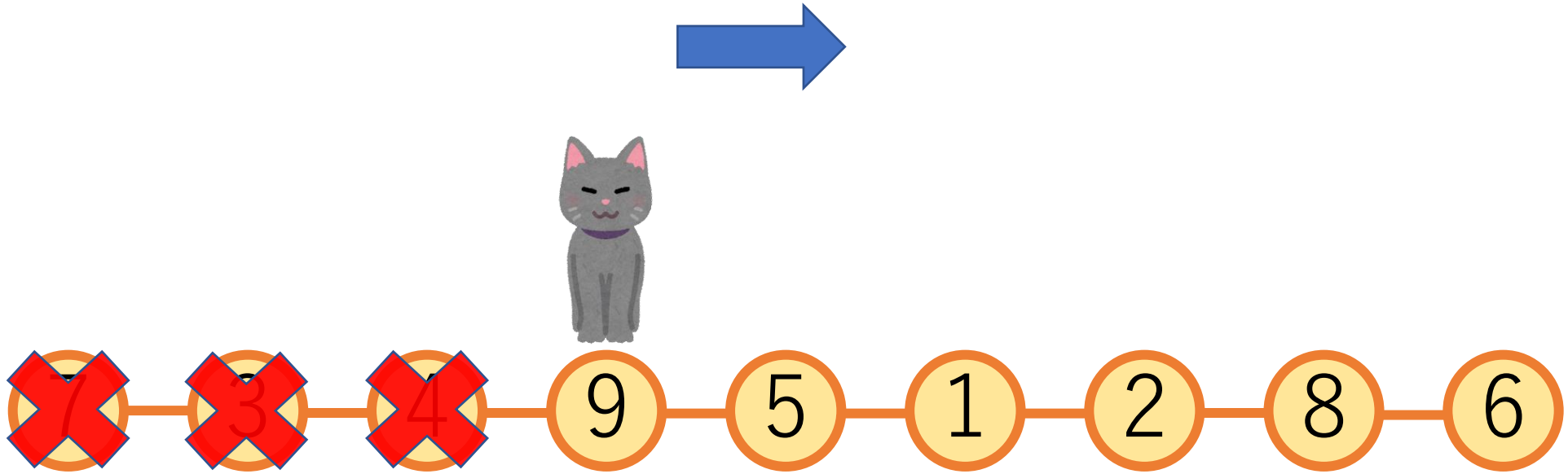
小課題3 (N<=5000, パス)

- 猫が次に右に移動するとする



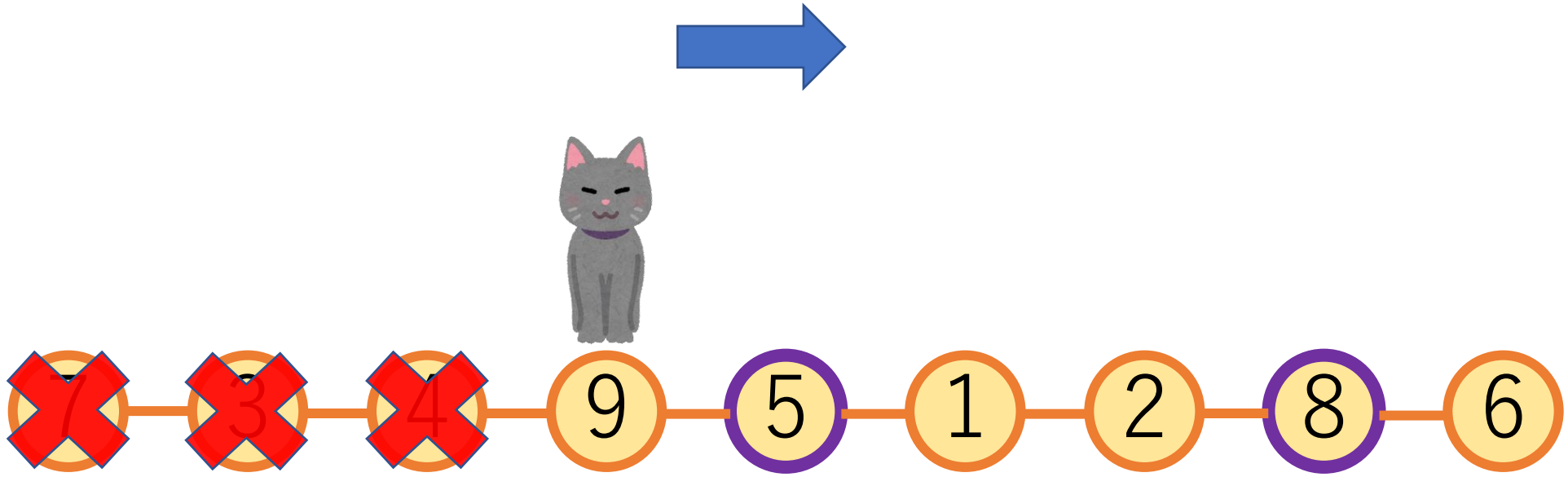
小課題3 (N<=5000, パス)

- 猫が次に右に移動するとする
- 左側の連結成分はすべて×にしてよい



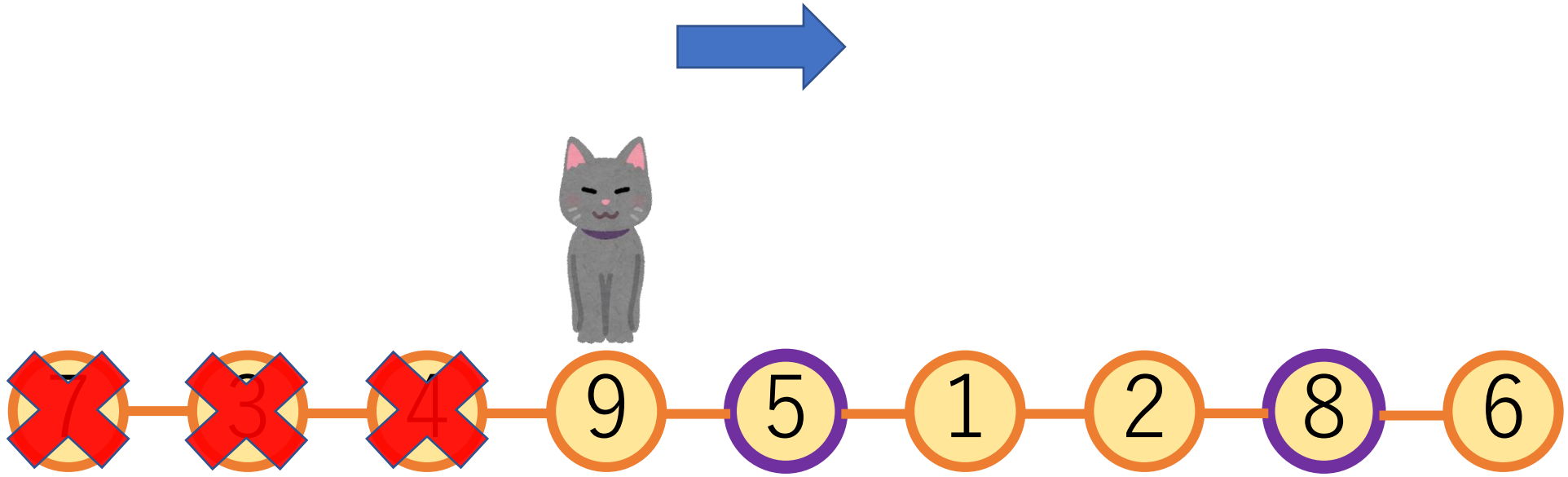
小課題3 (N<=5000, パス)

- 猫の移動先の候補は、左から見て最大値を更新する頂点



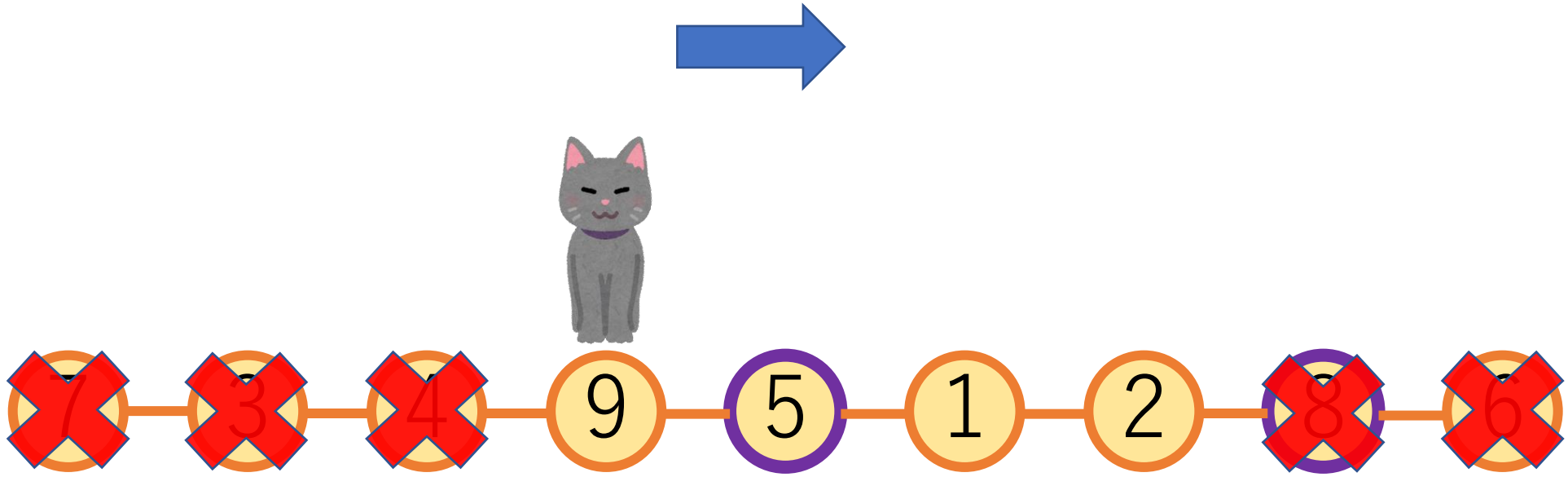
小課題3 (N<=5000, パス)

- 猫が手前の頂点（高さ５）に移動するとする



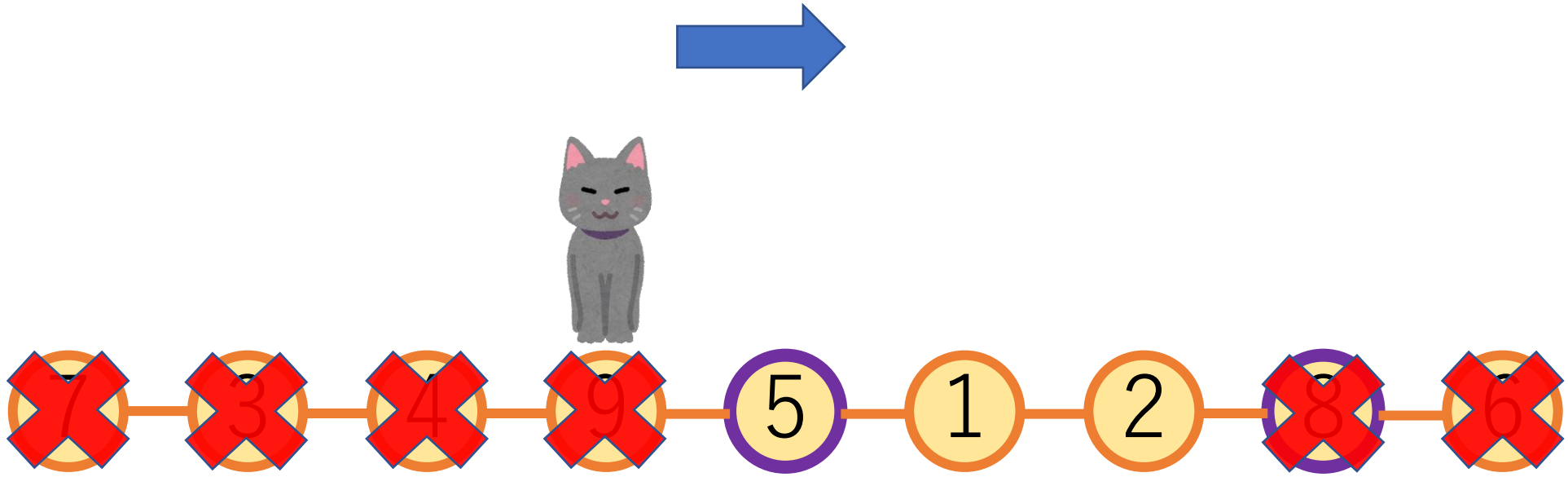
小課題3 (N<=5000, パス)

- 猫が手前の頂点（高さ５）に移動するとする
- 次の候補を×にする必要がある



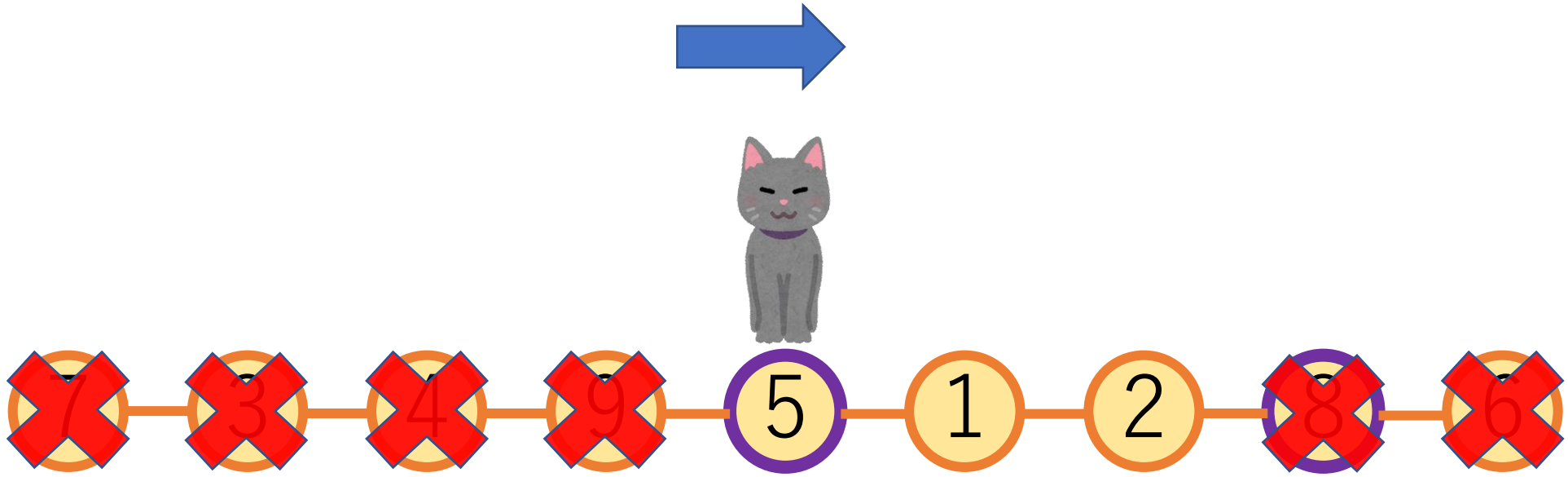
小課題3 (N<=5000, パス)

- 猫が手前の頂点（高さ５）に移動するとする
- 次の候補を×にする必要がある
- 猫のいる頂点を×にして移動させる



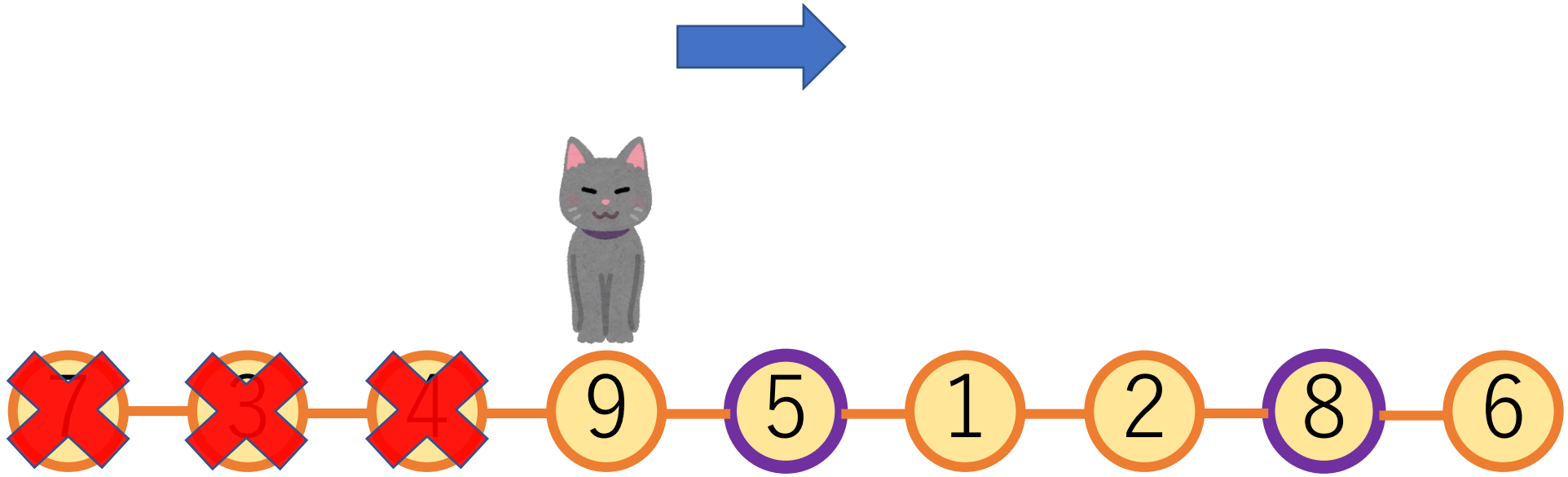
小課題3 (N<=5000, パス)

- 猫が手前の頂点（高さ５）に移動するとする
- 次の候補を×にする必要がある
- 猫のいる頂点を×にして移動させる



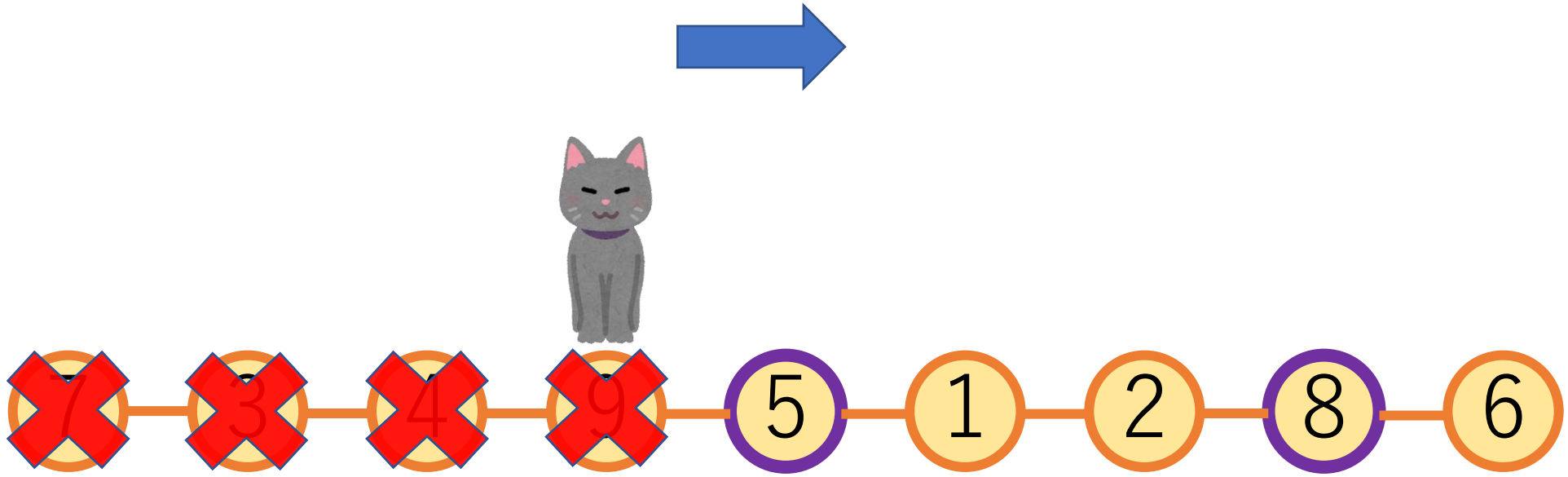
小課題3 (N<=5000, パス)

- 猫が奥の頂点（高さ 8 ）に移動するとする



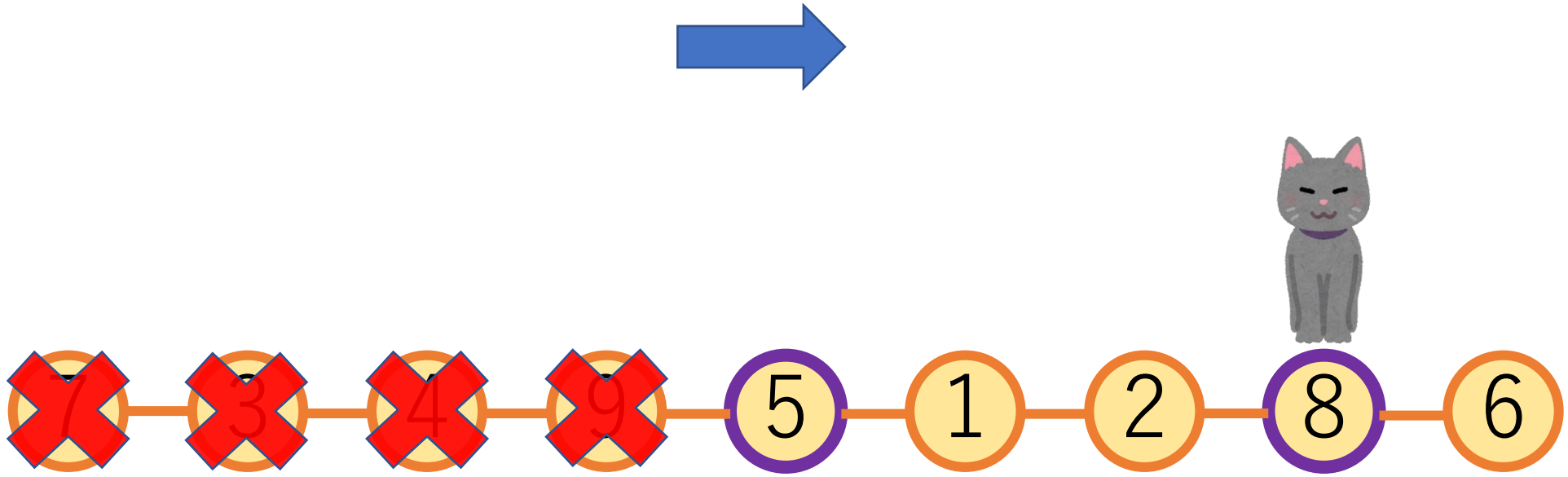
小課題3 (N<=5000, パス)

- 猫が奥の頂点（高さ 8 ）に移動するとする
- 猫のいる頂点を×にして移動させる



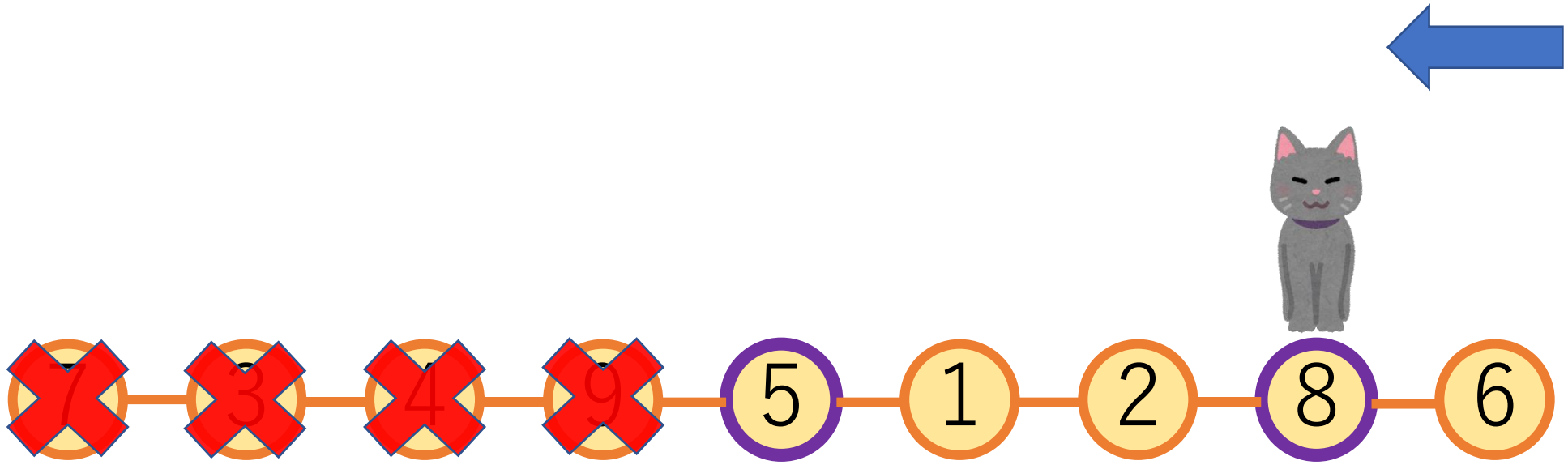
小課題3 (N<=5000, パス)

- 猫が奥の頂点（高さ 8 ）に移動するとする
- 猫のいる頂点を×にして移動させる



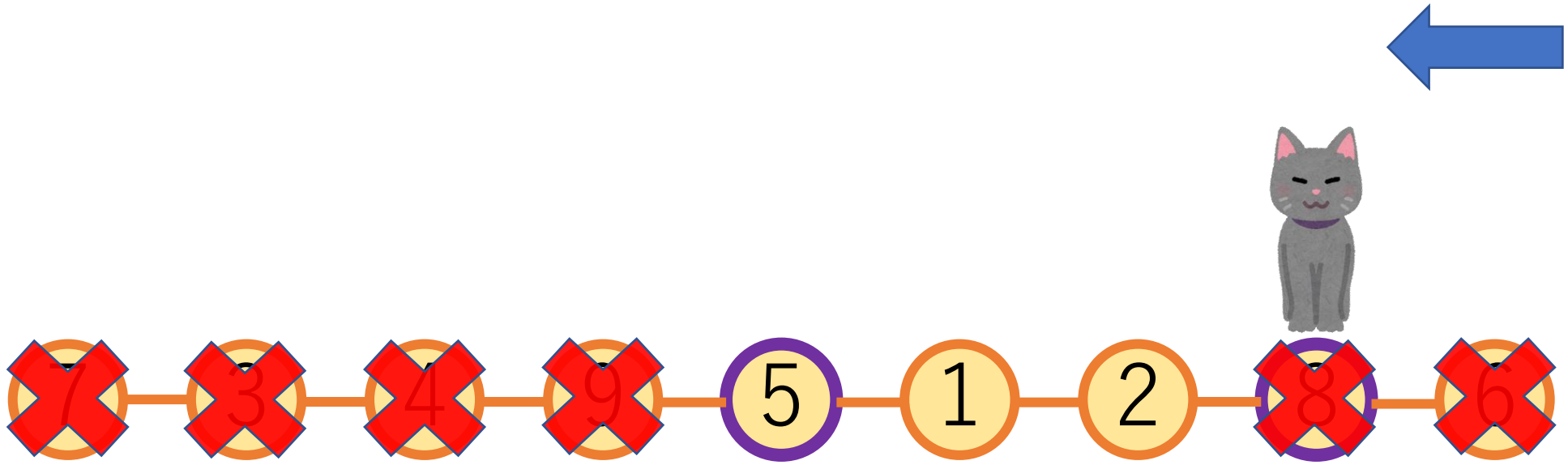
小課題3 (N<=5000, パス)

- 猫が奥の頂点（高さ 8 ）に移動するとする
- 猫のいる頂点を×にして移動させる
- 猫を右から追い詰めてみる



小課題3 (N<=5000, パス)

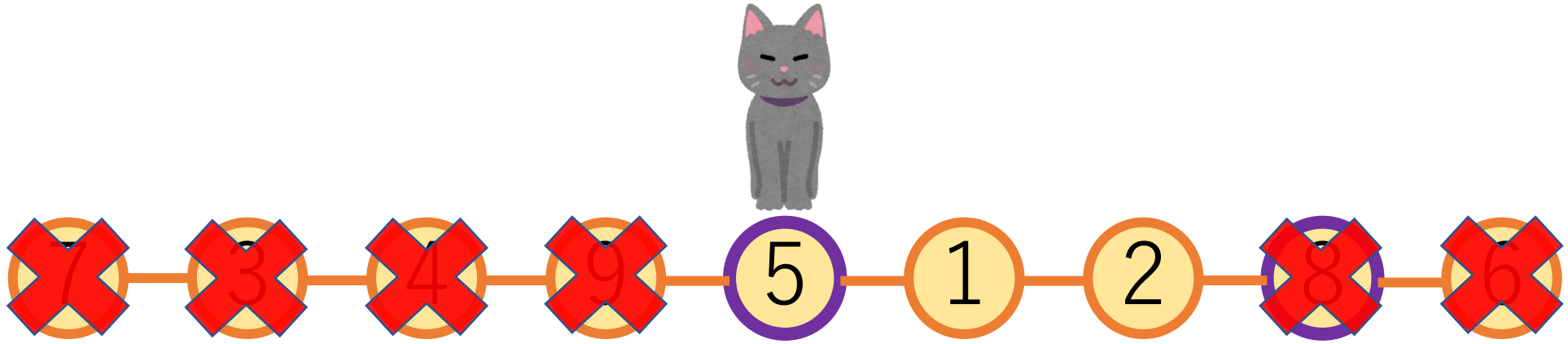
- 猫が奥の頂点（高さ 8 ）に移動するとする
- 猫のいる頂点を×にして移動させる
- 猫を右から追い詰めてみる



小課題3 (N<=5000, パス)

- 猫が奥の頂点（高さ 8 ）に移動するとする
- 猫のいる頂点を×にして移動させる
- 猫を右から追い詰めてみる

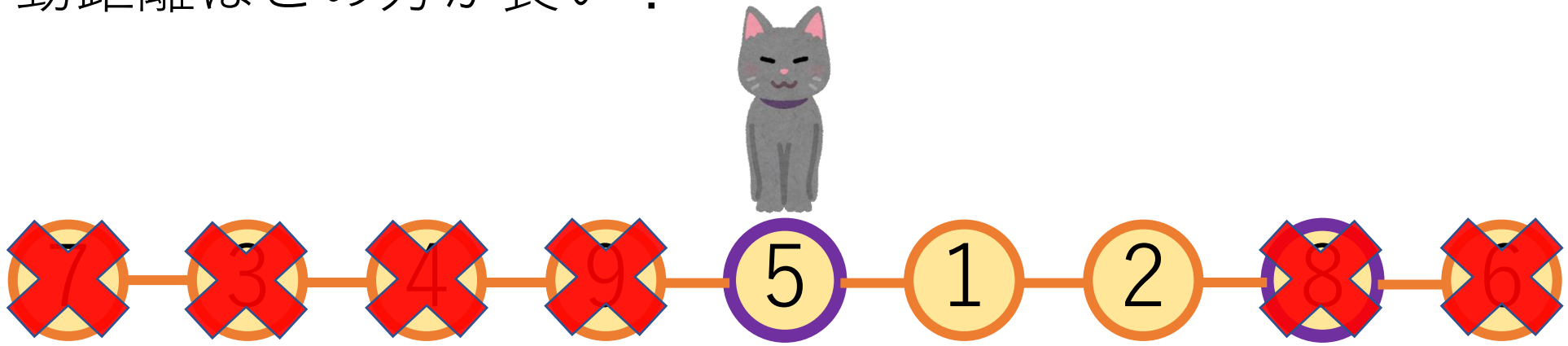
→手前の頂点（高さ 5 ）に移動した場合と同じ状態に！



小課題3 (N<=5000, パス)

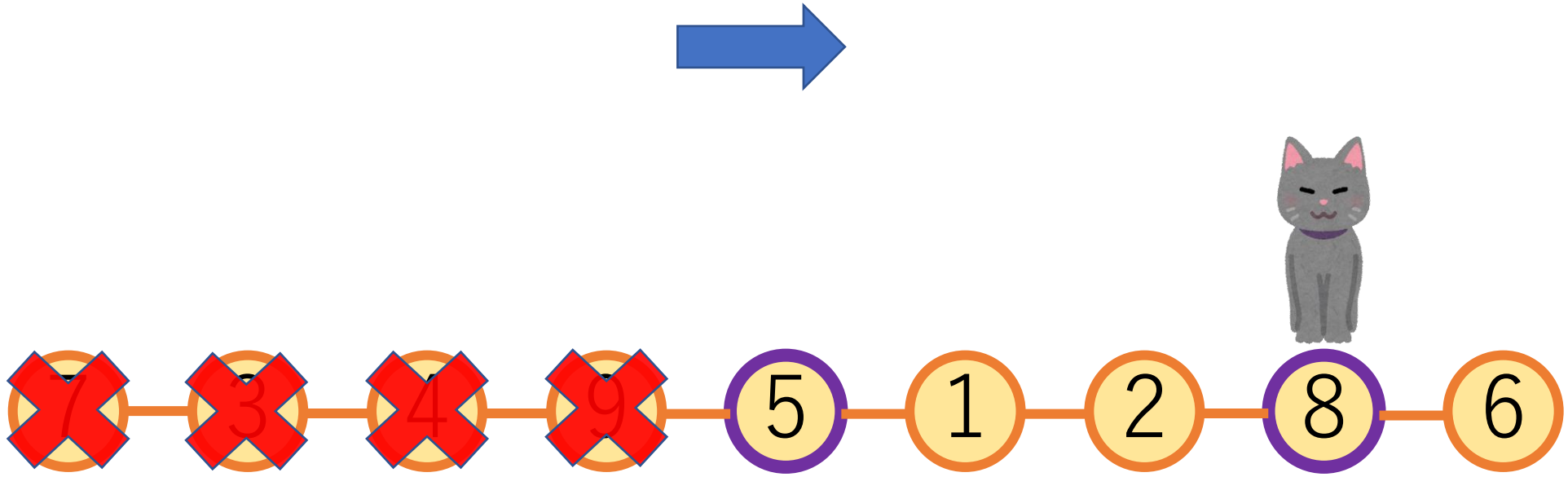
- 猫が奥の頂点（高さ８）に移動するとする
- 猫のいる頂点を×にして移動させる
- 猫を右から追い詰めてみる

→手前の頂点（高さ５）に移動した場合と同じ状態に！
　移動距離はこの方が長い！



小課題3 (N<=5000, パス)

- 何が言えるか？
- 左右それぞれで、最も高い頂点に移動するのがよい
- 移動した後は、元と同じ形の問題になる

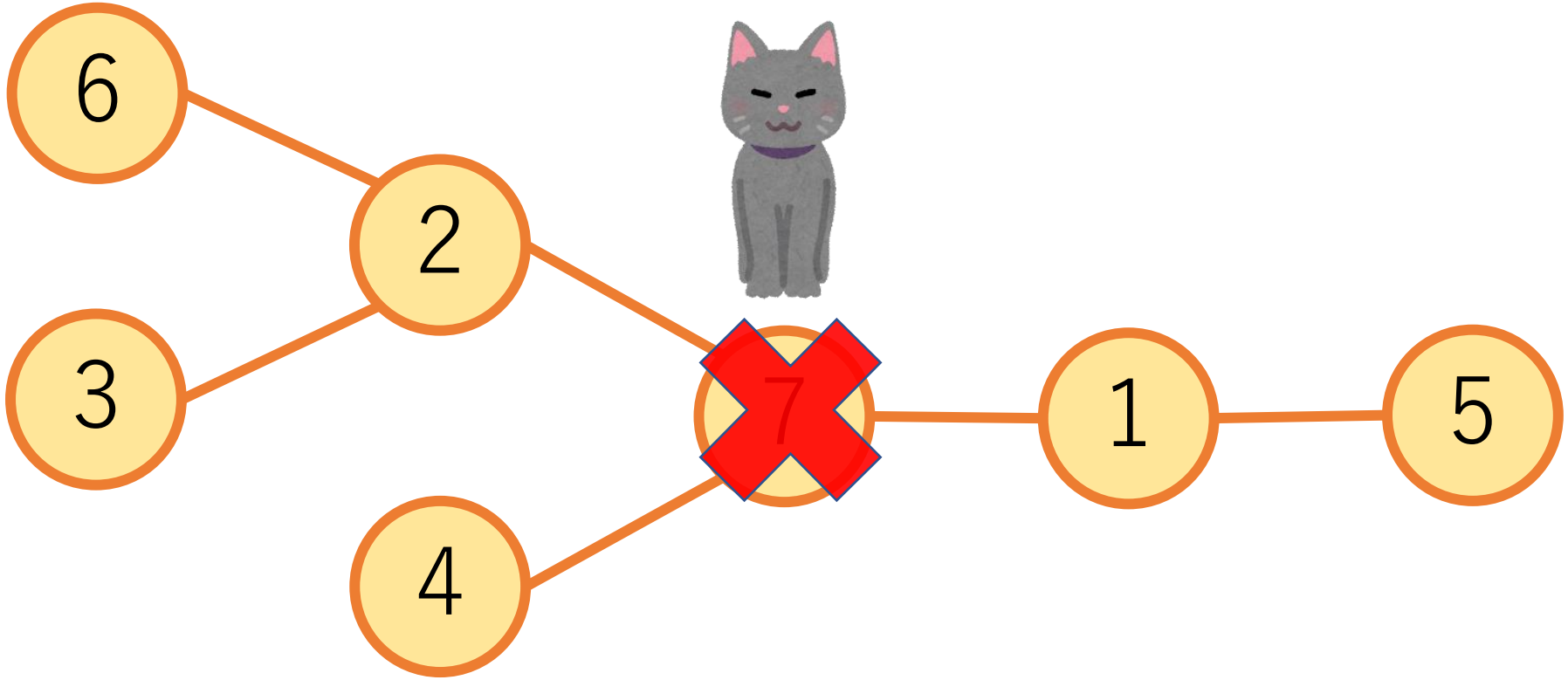


小課題3 (N<=5000, パス)

- dp[v]:=猫がいる頂点がvのときの，これからの移動距離の最大値
- 頂点 v で左右に分けたときの、左右の各区間の最も高い頂点 u1, u2 を求める ←**for文でO(N)**
- dp[u1], dp[u2] を再帰的に求める
- dp[v]=max{ dp[ui]+dist(ui,v) }
- **O(N^2)**

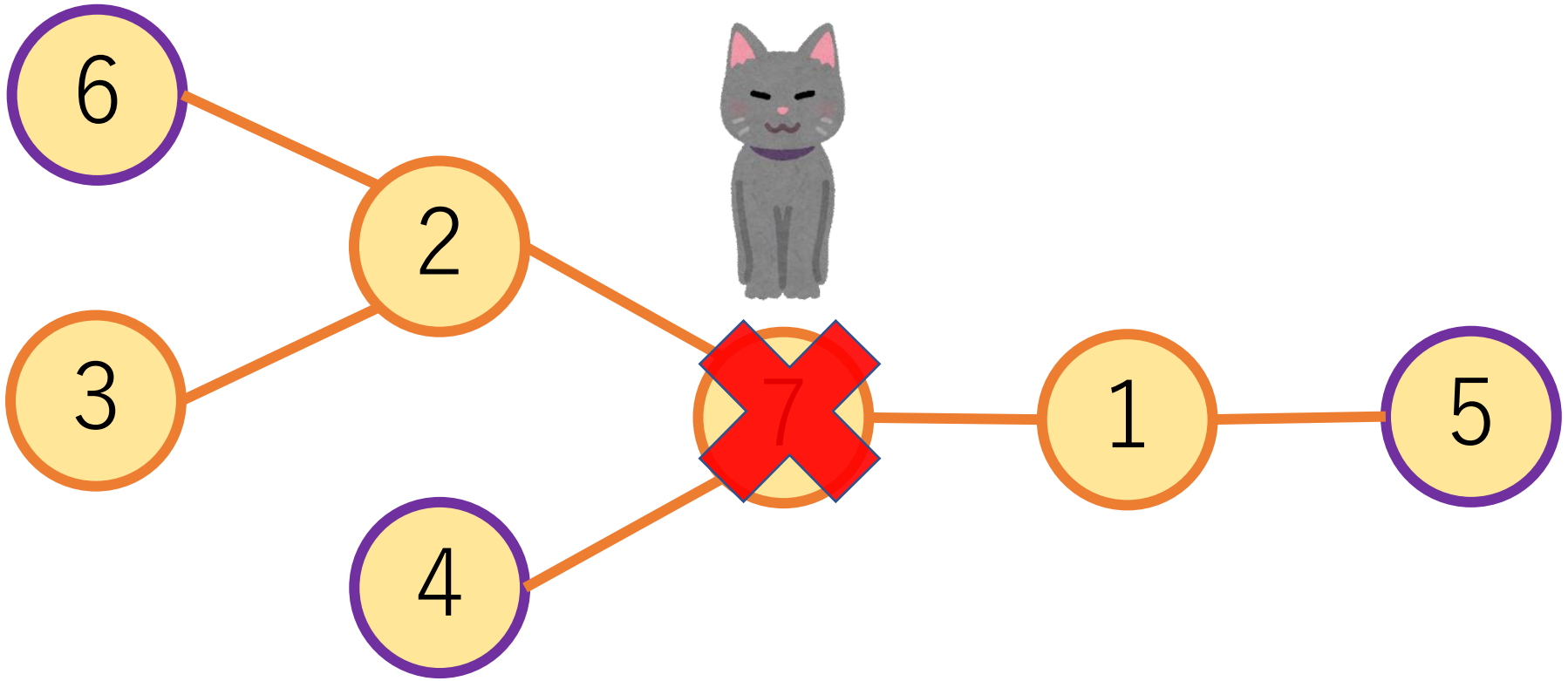
小課題4 (N<=5000, 木)

- 木の場合も、猫のいる頂点で分けたときの各連結成分で最も高い頂点に移動させるのが良い



小課題4 (N<=5000, 木)

- 木の場合も、猫のいる頂点で分けたときの各連結成分で最も高い頂点に移動させるのが良い



小課題4 (N<=5000, 木)

- dp[v]:=猫がいる頂点がvのときの，これからの移動距離の最大値
- 頂点 v で分けたときの、各連結成分の最も高い頂点 u1, u2, … を求める ←**DFSでO(N)**
- dp[u1], dp[u2], … を再帰的に求める
- dp[v]=max{ dp[ui]+dist(ui,v) }
- **O(N^2)**

小課題5 (N<=200000, パス)

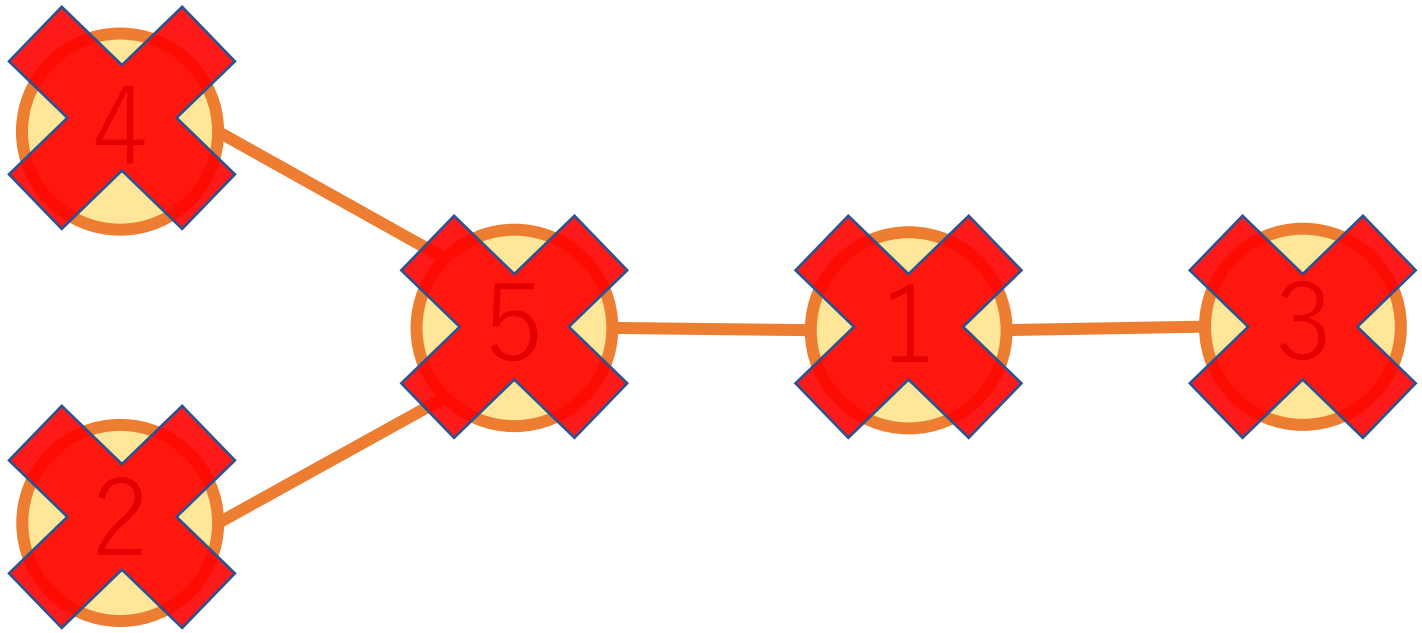
- dp[v]:=猫がいる頂点がvのときの，これからの移動距離の最大値
- 頂点 v で左右に分けたときの、左右の各区間の最も高い頂点 u1, u2 を求める ←**SegmentTreeでO(logN)**
- dp[u1], dp[u2] を再帰的に求める
- dp[v]=max{ dp[ui]+dist(ui,v) }
- **O(NlogN)**

小課題6 (N<=200000, 完全二分木)

- 解法が色々ある
- 連結成分の最大値を速く求める
- 部分木の最大値＆一点更新　がSegtreeもどきでできる
- 最も高い頂点の子の部分木の問題を解く→頂点の値を-infにする→もう一度元の部分木の問題を解く（すべて-infになったら終了）
- ２頂点間の距離…ビット演算でO(1)
- O(NlogN)

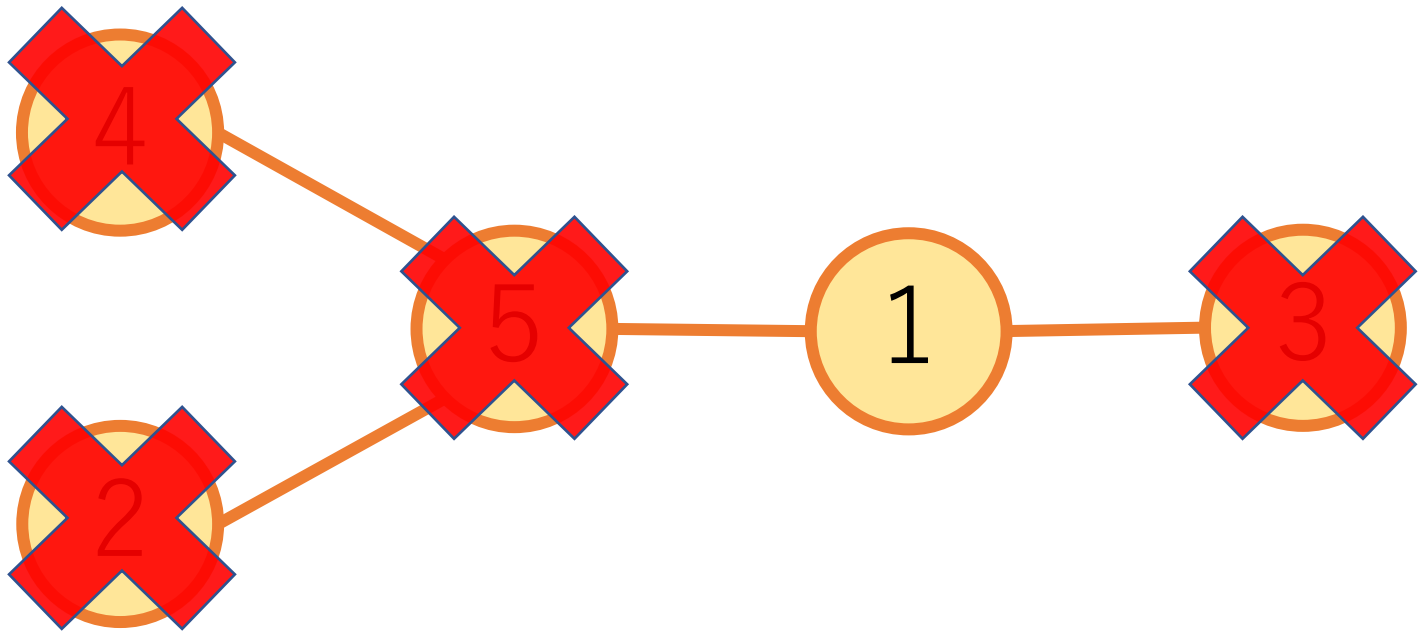
満点解法

- 方針：小課題４のDPを逆順に実行する（連結成分を切る方向ではなく、つなげる方向で考える）
- すべてに障害物を置いた状態から始める



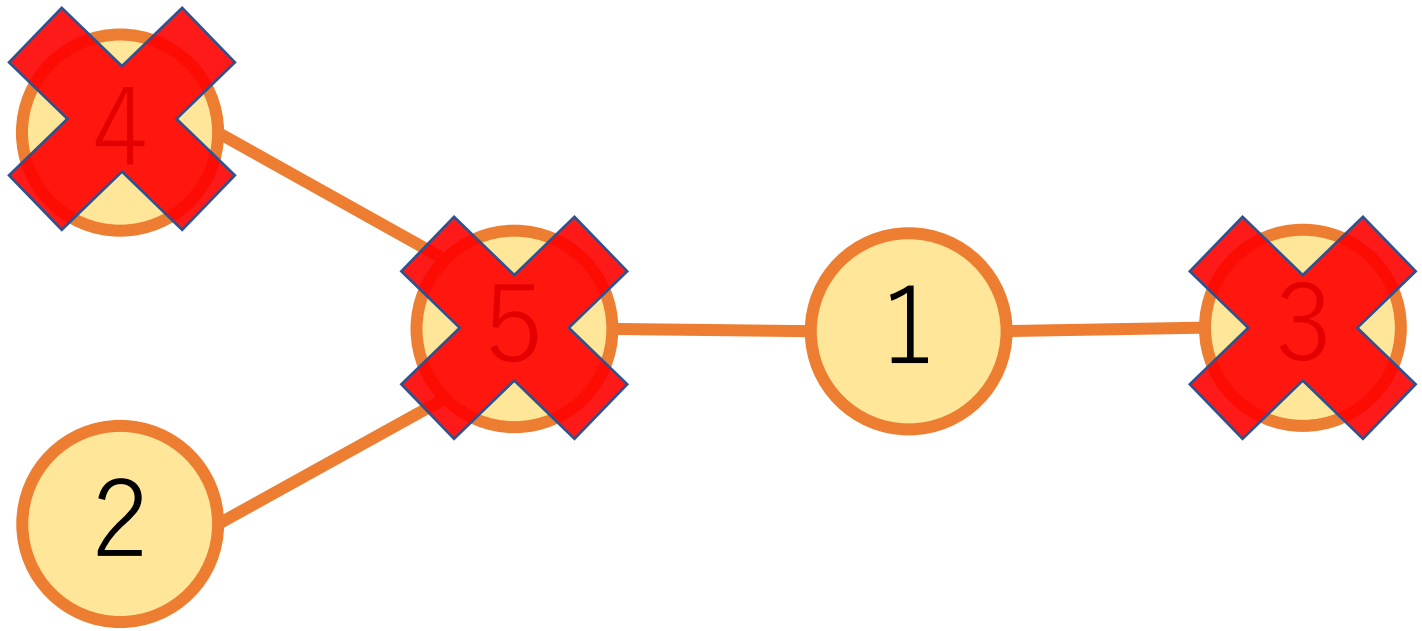
満点解法

- 最も低い頂点の障害物を取り除く



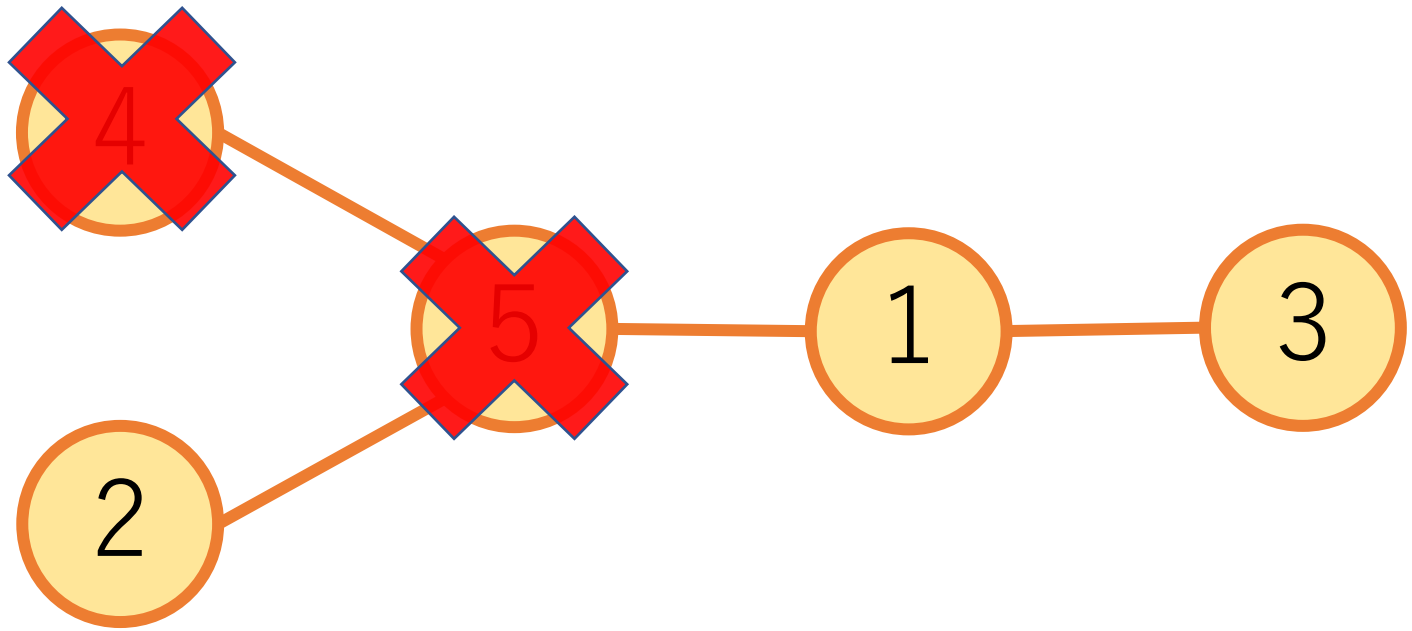
満点解法

- （再）最も低い頂点の障害物を取り除く



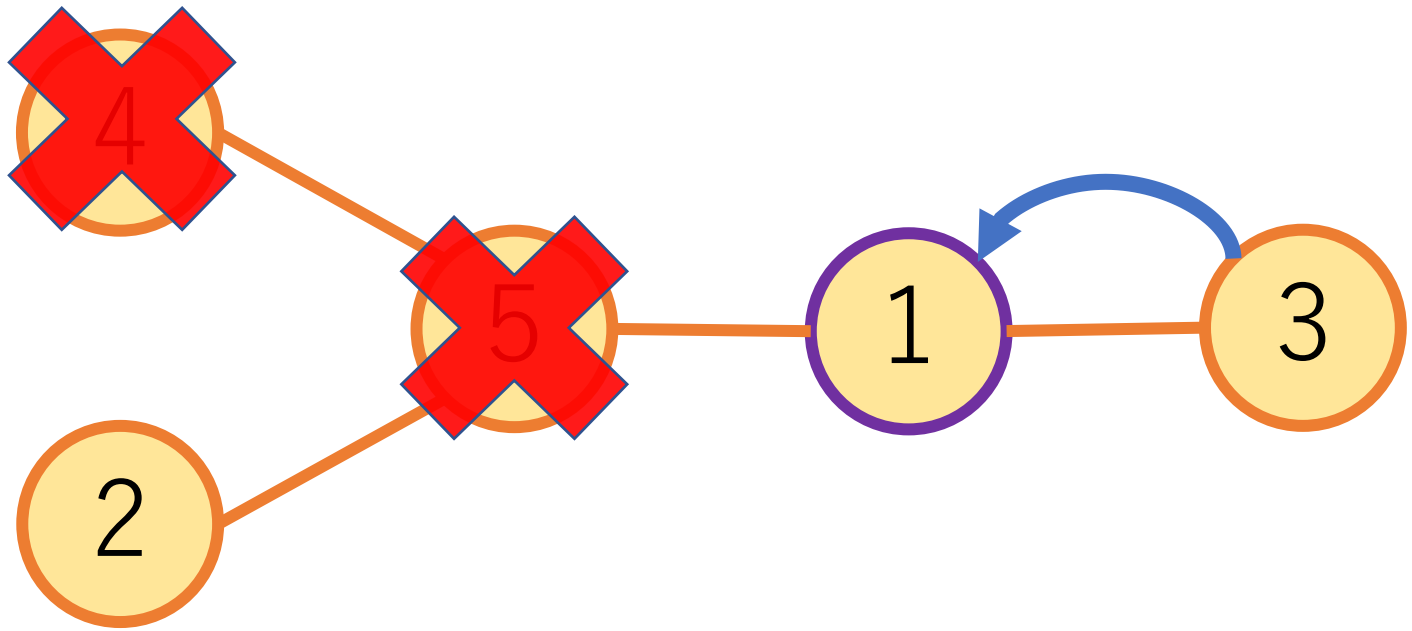
満点解法

- （再）最も低い頂点の障害物を取り除く



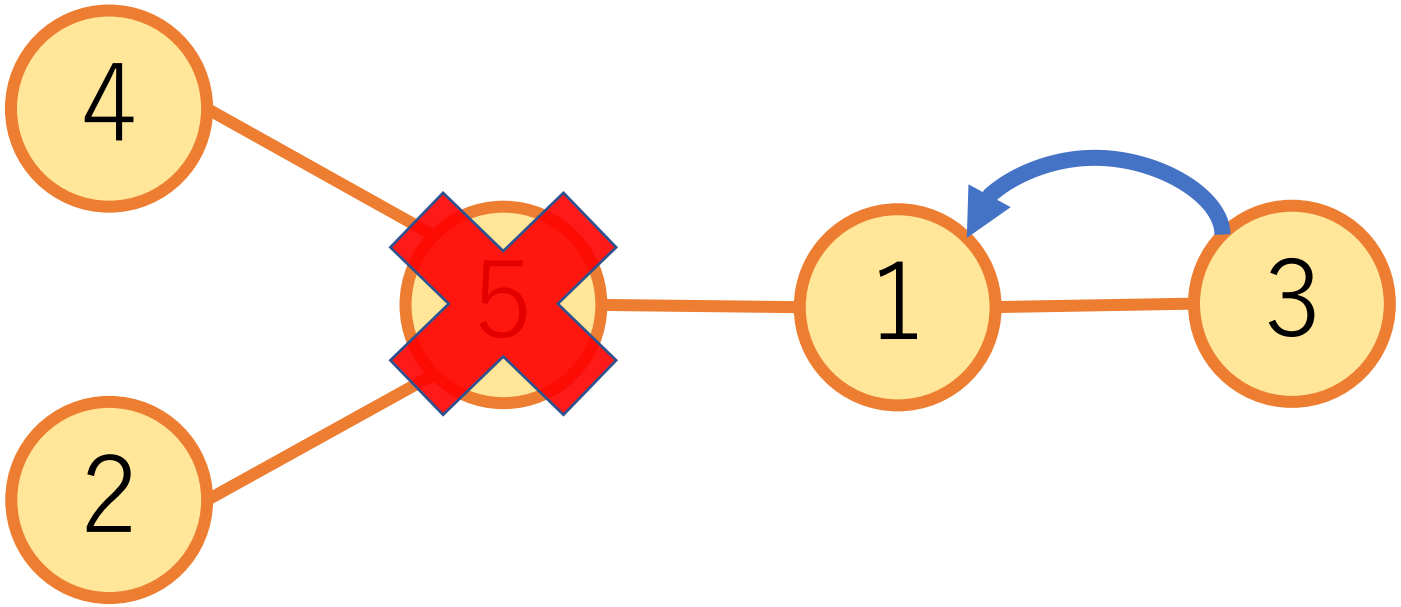
満点解法

- （再）最も低い頂点の障害物を取り除く
- 連結した場合、元からあった各連結成分内で最も高い頂点へ辺を張る



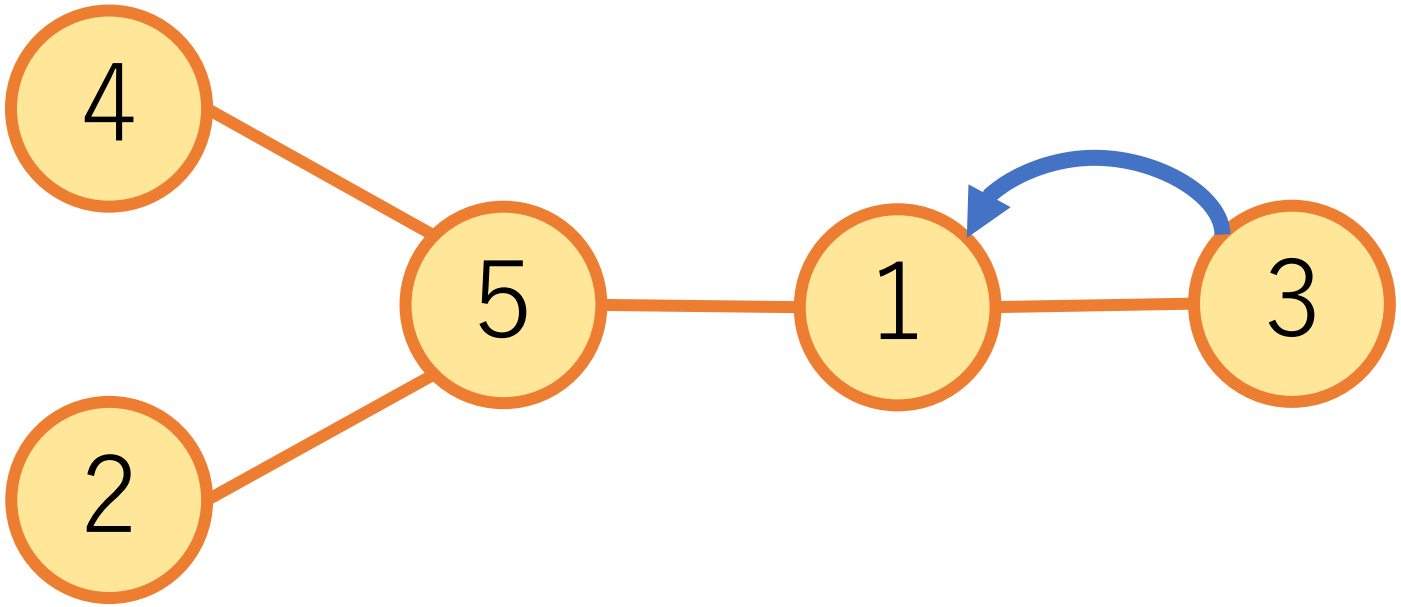
満点解法

- （再）最も低い頂点の障害物を取り除く



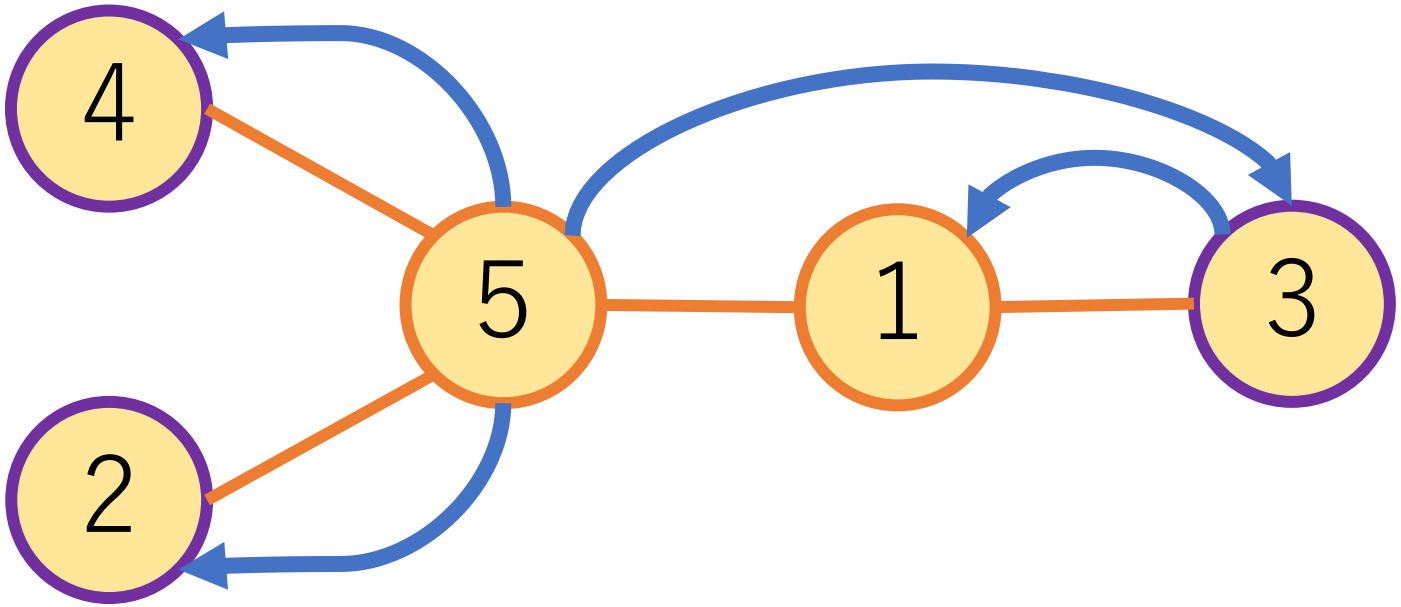
満点解法

- （再）最も低い頂点の障害物を取り除く



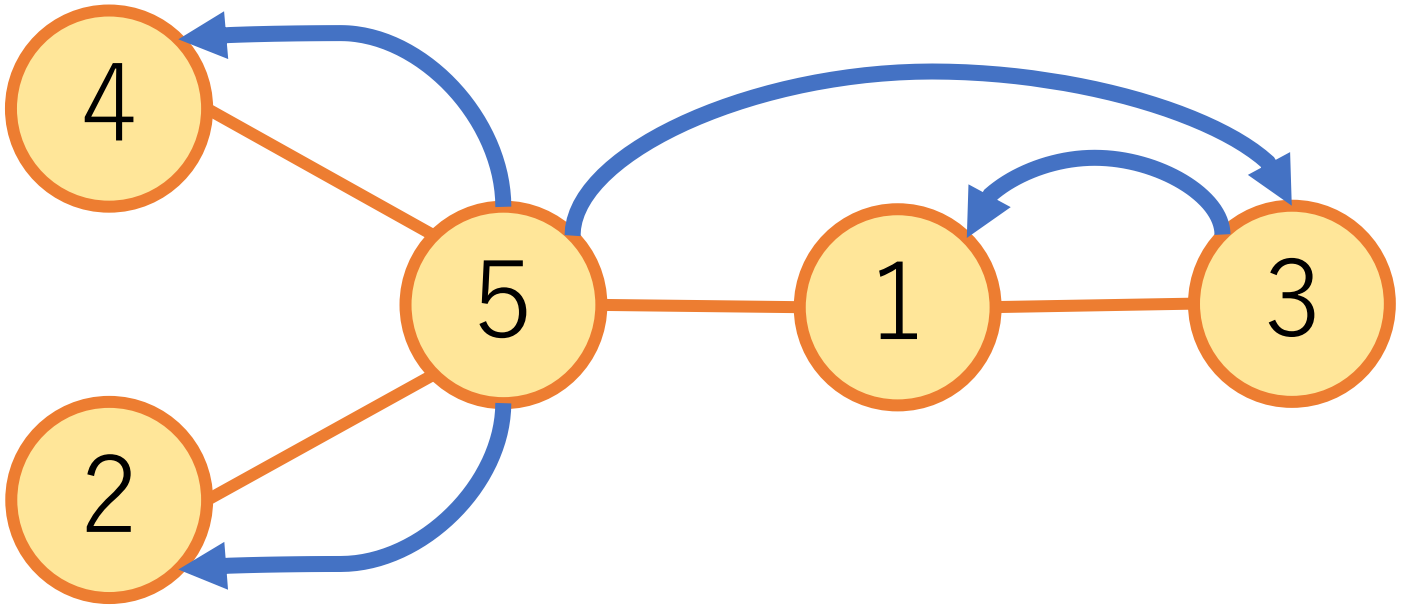
満点解法

- （再）最も低い頂点の障害物を取り除く
- 連結した場合、元からあった各連結成分内で最も高い頂点へ辺を張る



満点解法

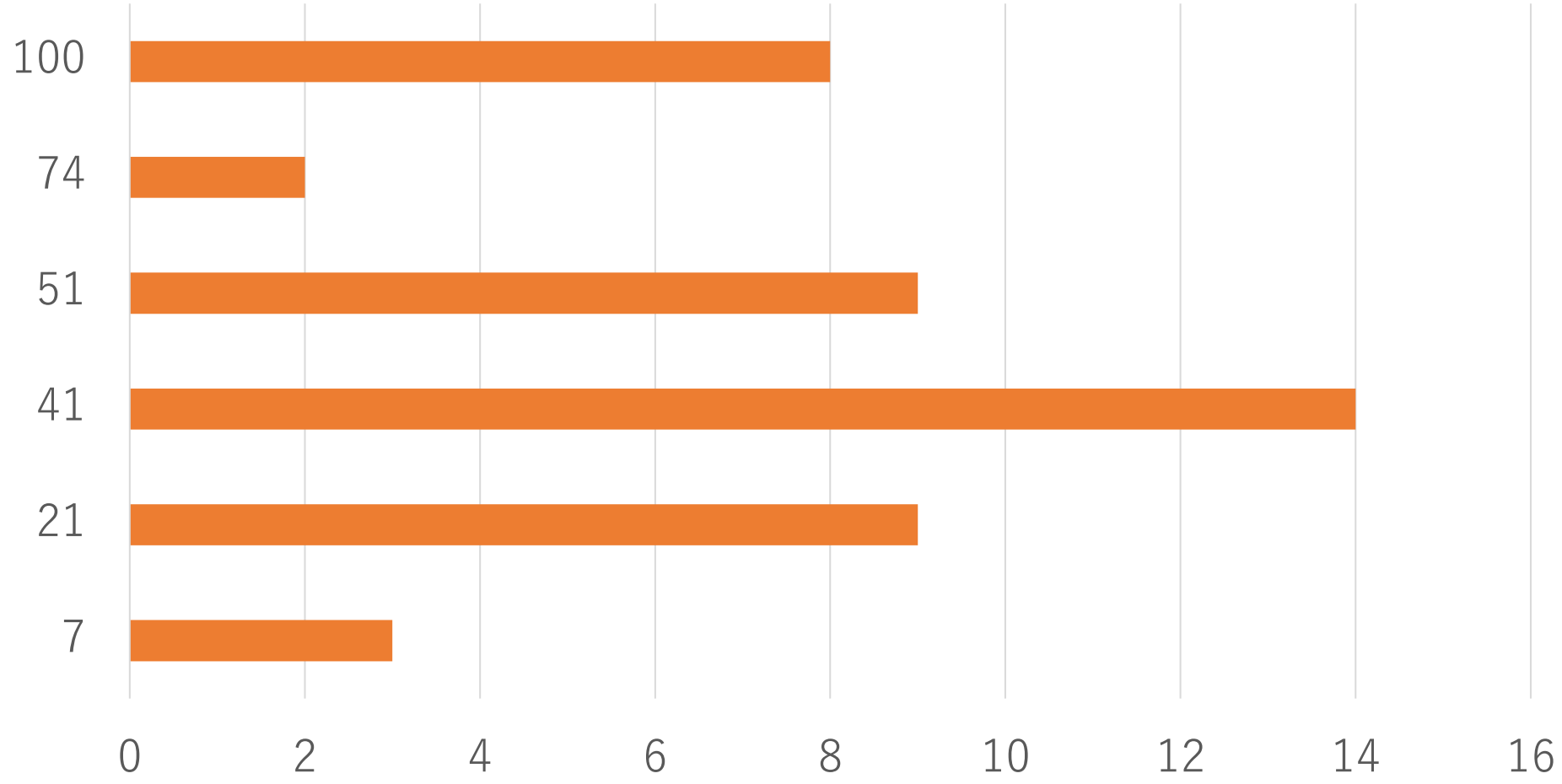
- （再）最も低い頂点の障害物を取り除く
- 連結した場合、元からあった各連結成分内で最も高い頂点へ辺を張る
- こうしてできた木での、根からの移動距離の最大値が答え



満点解法

- UnionFindで連結成分を管理
- 連結成分内で最も高い頂点が分かるようにする（根にするなど）
- 木の二頂点間の距離 dist(u,v)はLCAで一回O(logN)
- O(NlogN)

得点分布



- 0点：130人