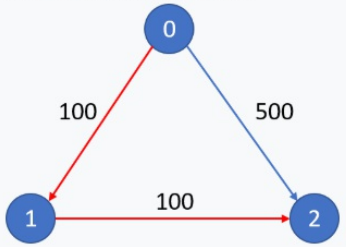
787. Cheapest Flights Within K Stops

Medium 1384 51 Add to List Share

There are `n` cities connected by `m` flights. Each flight starts from city `u` and arrives at `v` with a price `w`.

Now given all the cities and flights, together with starting city `src` and the destination `dst`, your task is to find the cheapest price from `src` to `dst` with up to `k` stops. If there is no such route, output `-1`.

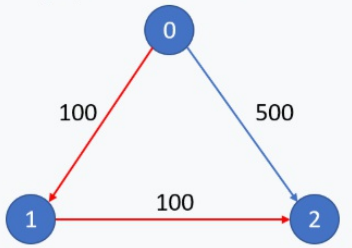
Example 1:
Input:
n = 3, edges = [[0,1,100],[1,2,100],[0,2,500]]
src = 0, dst = 2, k = 1
Output: 200
Explanation:
The graph looks like this:



The cheapest price from city 0 to city 2 with at most 1 stop costs 200, as marked red in the picture.

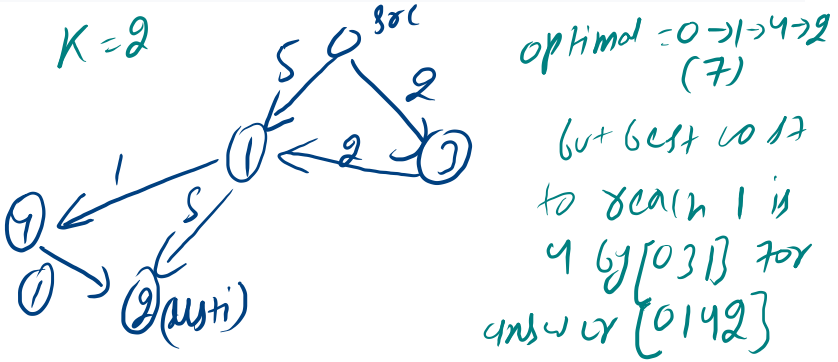
mean one node in between src and dest which is 2 edges in between.
(K+1) = 2

Example 2:
Input:
n = 3, edges = [[0,1,100],[1,2,100],[0,2,500]]
src = 0, dst = 2, k = 0
Output: 500
Explanation:
The graph looks like this:



The cheapest price from city 0 to city 2 with at most 0 stop costs 500, as marked blue in the picture.

0 node in between mean 1 edge in between
(K+1) = 1



optimal = 0→1→4→2
(7)

but best cost to reach 1 is 4 by [0,3] for answer [0,1,4,2]

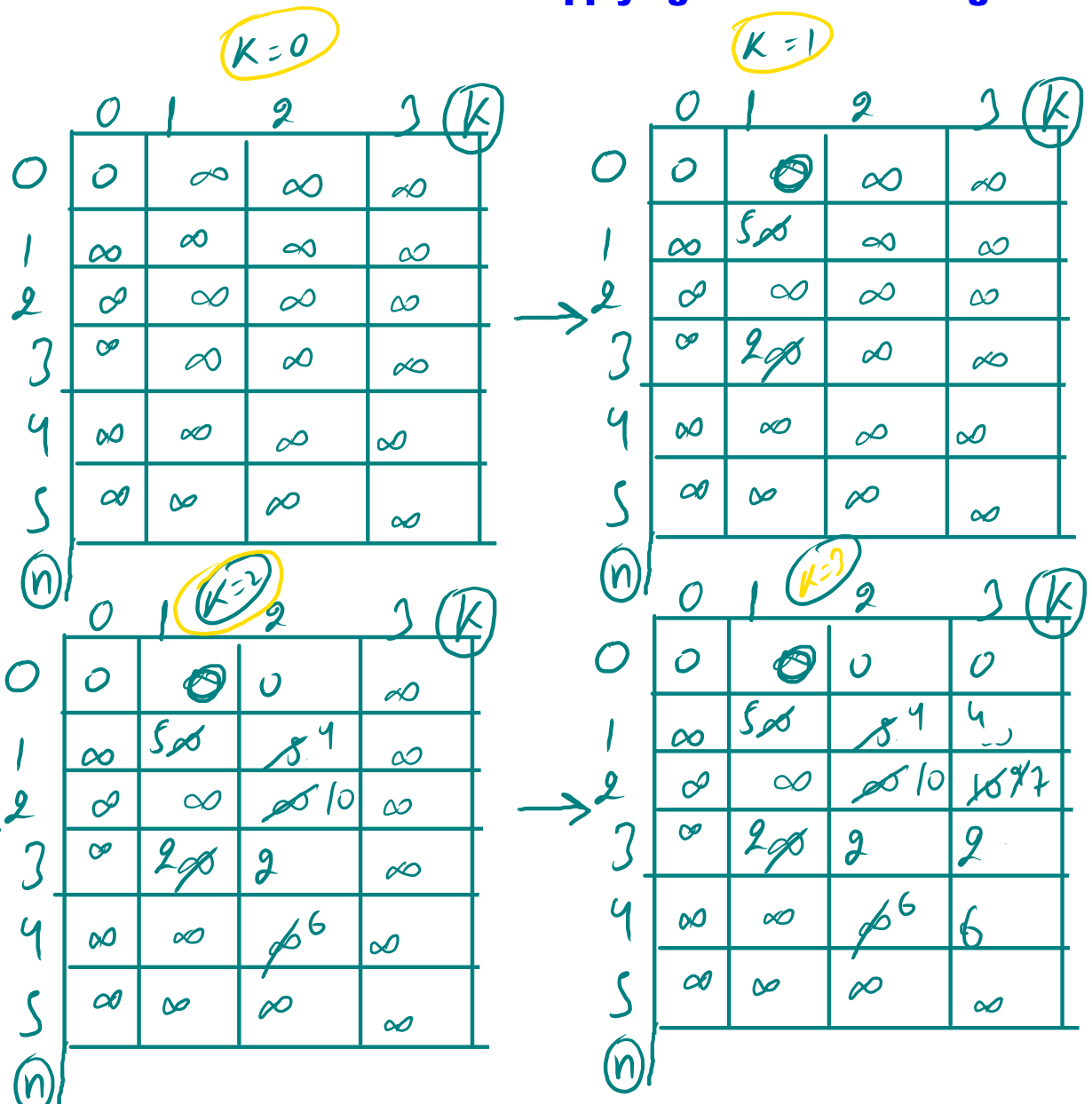
```cpp
int findCheapestPrice(int n, vector<vector<int>> &flights, int src, int dst, int K)
{
    vector<vector<pi>> graph(n);
    for (auto &ar : flights)
        graph[ar[0]].push_back({ar[1], ar[2]});

    priority_queue<vi, vector<vi>, greater<vi>> pq;
    pq.push({0, src, K + 1});

    while (pq.size() != 0)
    {
        auto rvtx = pq.top();
        pq.pop();

        if (rvtx[1] == dst)
            return rvtx[0];
        if (rvtx[2] == 0)
            continue;

        for (auto &e : graph[rvtx[1]])
            pq.push({rvtx[0] + e.second, e.first, rvtx[2] - 1});
    }
    return -1;
}
```

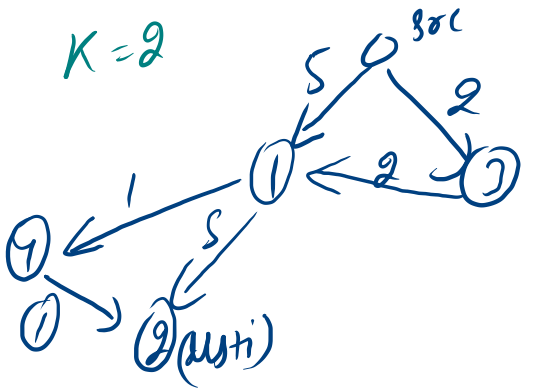
graph

to work on 0th index

idhar vis ke check isliye nahi lgaya because cost and length dono sath hai. optimal cost pe length correct ayye zaruri nahi hai. ek path abhi costly hai aage jake kam cost pe destination reach krva skta hai.

Applying Bellman Ford algo






```
5
[[0,1,5],[1,2,5],[0,3,2],[3,1,2],[1,4,1],[4,2,1]]
0
2
2
```