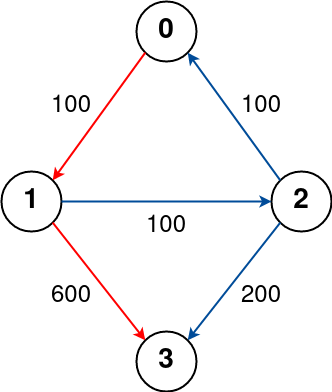
Cheapest Flights Within K Stops [LeetCode](https://leetcode.com/problems/cheapest-flights-within-k-stops/description/)

There are n cities connected by some number of flights. You are given an array flights where flights[i] = [fromi, toi, pricei] indicates that there is a flight from city fromi to city toi with cost pricei.

You are also given three integers src, dst, and k, return ***the cheapest price****from*src*to*dst*with at most*k*stops.*If there is no such route, return-1.

Example:

 Source = 0, Destination = 3, k = 1

Output: 700

**Approach 1: Function to find the cheapest price from source to destination with at most k stops**

* **Explanation:**
  + The **findCheapestPrice** function constructs an adjacency list representing flights and initializes a vector to store the cost of reaching each node from the source.
  + It uses a BFS traversal with a queue to explore possible flights, updating the cost if a cheaper path is found.
  + The traversal stops when the destination is reached with at most k stops, and the cost of reaching the destination is returned.
  + If the destination is unreachable, the function returns -1.
* **Time Complexity:**
  + **The time complexity is O(V + E), where V is the number of vertices (nodes) and E is the number of edges (flights).**
    - **Both the creation of the adjacency list and BFS traversal contribute to the time complexity.**
* **Space Complexity:**
  + **The space complexity is O(V + E), where V is the number of vertices (nodes) and E is the number of edges (flights).**
    - **The adjacency list, cost vector, and the BFS queue contribute to space usage.**