Dijkstra算法、分层图最短路

前置知识:
讲解025、026、027-堆结构
讲解032-位图，用一个整型变量最多可以表示32个状态，并且非常方便、快速
讲解059-建图、链式前向星
讲解061-最小生成树 prim算法利用反向索引堆做的优化 强烈推荐看一下
讲解062-宽度优先遍历及其扩展
讲解059~讲解065都是【必备】课程有关图的内容，建议从头开始学习

Dijkstra算法及其优化

分层图最短路，又叫扩点最短路

左程云

Dijkstra算法、分层图最短路

Dijkstra算法：给定一个源点，求解从源点到每个点的最短路径长度。单源最短路径算法。
适用范围：有向图、边的权值没有负数

彻底暴力的Dijkstra算法，不讲、时间复杂度太差、无意义

普通堆实现的Dijkstra算法，最普遍、最常用
算法核心过程：
节点弹出过就忽略
节点没弹出过，让其它没弹出节点距离变小的记录加入堆

反向索引堆实现的Dijkstra算法，最快速、最极致
核心在于掌握反向索引堆

对应本节题目1、题目2、题目3

Dijkstra算法、分层图最短路

普通堆实现的Dijkstra算法，时间复杂度O(m * log m)，m为边数

1，distance[i]表示从源点到i点的最短距离，visited[i]表示i节点是否从小根堆弹出过
2，准备好小根堆，小根堆存放记录：(x点，源点到x的距离)，小根堆根据距离组织
3，令distance[源点]=0，(源点，0)进入小根堆
4，从小根堆弹出(u点，源点到u的距离)
   a. 如果visited[u] == true，不做任何处理，重复步骤4
   b. 如果visited[u] == false，令visited[u] = true，u就算弹出过了
      然后考察u的每一条边，假设某边去往v，边权为w
      1）如果visited[v] == false 并且 distance[u] + w < distance[v]
         令distance[v] = distance[u] + w，把(v, distance[u] + w)加入小根堆
      2）处理完u的每一条边之后，重复步骤4
5，小根堆为空过程结束，distance表记录了源点到每个节点的最短距离。

Dijkstra算法、分层图最短路

反向索引堆实现的Dijkstra算法，时间复杂度O(m * log n)，n为节点数，m为边数

1，准备好反向索引堆，根据源点到当前点的距离组织小根堆，可以做到如下操作
   a. 新增记录(x, 源点到x的距离)　b. 当源点到x的距离更新时，可以进行堆的调整
   c. x点一旦弹出，以后忽略x　　d. 弹出堆顶的记录(u, 源点到u的距离)
2，把(源点，0)加入反向索引堆，过程开始
3，反向索引堆弹出(u，源点到u的距离)，考察u的每一条边，假设某边去往v，边权为w
   1）如果v没有进入过反向索引堆里，新增记录(v, 源点到u的距离 + w)
   2）如果v曾经从反向索引堆弹出过，忽略
   3）如果v在反向索引堆里，看看源点到v的距离能不能变得更小，如果能，调整堆；不能，忽略
   4）处理完u的每一条边，重复步骤3
4 反向索引堆为空过程结束。反向索引堆里记录了源点到每个节点的最短距离。

左程云

Dijkstra算法、分层图最短路

分层图最短路，又叫扩点最短路

不把实际位置看做图上的点，而是把 实际位置及其状态的组合 看做是图上的点，然后搜索
bfs 或者 Dijkstra的过程不变，只是扩了点（分层）而已
原理简单，核心在于如何扩点、如何到达、如何算距离，每个题可能都不一样

对应本节题目4、题目5、题目6

Dijkstra算法、分层图最短路

题目1
Dijkstra算法模版
普通堆的实现
反向索引堆的实现
测试链接：https://leetcode.cn/problems/network-delay-time
测试链接：https://www.luogu.com.cn/problem/P4779

左程云

Dijkstra算法、分层图最短路

题目2
最小体力消耗路径
你准备参加一场远足活动。给你一个二维 rows x columns 的地图 heights
其中 heights[row][col] 表示格子 (row, col) 的高度
一开始你在最左上角的格子 (0, 0) ，且你希望去最右下角的格子 (rows-1, columns-1)
（注意下标从 0 开始编号）。你每次可以往 上，下，左，右 四个方向之一移动
你想要找到耗费 体力 最小的一条路径
一条路径耗费的体力值是路径上相邻格子之间，高度差绝对值的最大值
请你返回从左上角走到右下角的最小 体力消耗值
测试链接 ： https://leetcode.cn/problems/path-with-minimum-effort/

Dijkstra算法、分层图最短路

题目3
水位上升的泳池中游泳
在一个 n x n 的整数矩阵 grid 中
每一个方格的值 grid[i][j] 表示位置 (i, j) 的平台高度
当开始下雨时，在时间为 t 时，水池中的水位为 t
你可以从一个平台游向四周相邻的任意一个平台，但是前提是此时水位必须同时淹没这两个平台
假定你可以瞬间移动无限距离，也就是默认在方格内部游动是不耗时的
当然，在你游泳的时候你必须待在坐标方格里面。
你从坐标方格的左上平台 (0，0) 出发
返回 你到达坐标方格的右下平台 (n-1, n-1) 所需的最少时间
测试链接：https://leetcode.cn/problems/swim-in-rising-water/

Dijkstra算法、分层图最短路

题目4
获取所有钥匙的最短路径
给定一个二维网格 grid ，其中：
'.' 代表一个空房间、'#' 代表一堵墙、'@' 是起点
小写字母代表钥匙、大写字母代表锁
从起点开始出发，一次移动是指向四个基本方向之一行走一个单位空间
不能在网格外面行走，也无法穿过一堵墙
如果途经一个钥匙，我们就把它捡起来。除非我们手里有对应的钥匙，否则无法通过锁。
假设 k 为 钥匙/锁 的个数，且满足 1 <= k <= 6，
字母表中的前 k 个字母在网格中都有自己对应的一个小写和一个大写字母
换言之，每个锁有唯一对应的钥匙，每个钥匙也有唯一对应的锁
另外，代表钥匙和锁的字母互为大小写并按字母顺序排列
返回获取所有钥匙所需要的移动的最少次数。如果无法获取所有钥匙，返回 -1 。
测试链接：https://leetcode.cn/problems/shortest-path-to-get-all-keys

Dijkstra算法、分层图最短路

题目5
电动车游城市
小明的电动车电量充满时可行驶距离为 cnt
每行驶 1 单位距离消耗 1 单位电量，且花费 1 单位时间
小明想选择电动车作为代步工具。地图上共有 N 个景点，景点编号为 0 ~ N-1
他将地图信息以 [城市 A 编号,城市 B 编号,两城市间距离] 格式整理在在二维数组 paths，
表示城市 A、B 间存在双向通路。
初始状态，电动车电量为 0。每个城市都设有充电桩，
charge[i] 表示第 i 个城市每充 1 单位电量需要花费的单位时间。
请返回小明最少需要花费多少单位时间从起点城市 start 抵达终点城市 end
测试链接：https://leetcode.cn/problems/DFPeFJ/

Dijkstra算法、分层图最短路

题目6
飞行路线
Alice和Bob现在要乘飞机旅行，他们选择了一家相对便宜的航空公司
该航空公司一共在n个城市设有业务，设这些城市分别标记为0 ~ n-1
一共有m种航线，每种航线连接两个城市，并且航线有一定的价格
Alice 和 Bob 现在要从一个城市沿着航线到达另一个城市，途中可以进行转机
航空公司对他们这次旅行也推出优惠，他们可以免费在最多k种航线上搭乘飞机
那么 Alice 和 Bob 这次出行最少花费多少
测试链接：https://www.luogu.com.cn/problem/P4568