FOI 2021 夏令营结业考试 题解

这是一场简单的结业考试，如果你认真听课了应该可以拿到至少 300 分。

std 文件夹下有标程，有些题目有两个标程。部分标程可能使用了较新的 C++ 语法，如果看不懂可以看另一个标程。

A: 附魔书合成 (enchant)

人口普查题，简单模拟。

样例十分良心，里面已经包含了所有情况。

结合样例应该能很容易找到代码中的错误（如果有的话）。

~~如果没过这题应该被拉出去打。~~

B: 旅行 (tour)

树上差分。

把通过每条道路的花费分开计算，每条道路要不然全买单程票要不然买一张多程票。

对于一条道路，设单程票、多程票花费分别为 a, b，如果 Steve 全程经过这条道路 c 次，那买单程票就需要 $a \times c$ 个绿宝石，多程票就是 b 个绿宝石，选择花费较小的一种方案即可。

把所有道路的花费加起来就是答案。

如何计算每条道路经过次数？边差分模板题。

标程采用链式前向星存图，倍增求 LCA。

C: 技能树 (skill)

树状数组，欧拉序（DFS 序）。

对于每一次询问我们要求某个点的权值，而某个点的权值就是它经历过的被加的那些权值之和。

不妨交换求和符号，改为计算每次修改操作对哪些点有贡献。

可以发现，对于一个点，只有对它子树内的点（包括它本身）的修改才会对它的权值产生贡献。

于是祖先加单点查询就可以转化为单点加子树查询。

具体地，每次修改只需进行单点加，而查询时查询子树内所有修改操作的和即可。

由欧拉序的性质可以知道，任意一棵子树内所有点占据了欧拉序中连续的一段区间。

于是我们可以用树状数组处理这两种操作。

和例题几乎一样，直接欧拉序展开后树状数组维护前缀和即可。

D: 探矿 (explore)

思维，搜索，二分答案。

先考虑没有修改操作时怎么做。

一个容易猜到的结论：玩家只要能抵达 **足够多的** 矿洞区块，就能抵达无穷多个矿洞区块。

- 那「足够多」具体要多少呢？
- 事实上，假设一个地图单元上有 s 个矿洞区块，那只要 $s+1$ 个矿洞区块就够了。

证明如下：

- 假设玩家从起点区块 S 出发能抵达 $s+1$ 个不同的矿洞区块。
- 根据抽屉原理，这 $s+1$ 个矿洞区块中一定至少有两个位于不同的地图单元，并且对应地图单元上的同一位置 T。
- 不妨把这两个区块按到达顺序分别叫做 T1, T2。
- 我们可以先从 S 出发，到达 T1，再到达 T2。
- 接下来，我们用同样的方法到达第三个地图单元上 T 的对应区块 T3，再到 T4、T5、……
- 以此类推，我们可以到达无穷个地图单元上的 T，而这个过程中玩家就可以抵达无穷多个矿洞区块。

于是我们就获得了一个做法：从起点开始 BFS 或 DFS，如果搜到超过 $s+1$ 个区块就停止搜索并输出 `Yes`，否则输出 `No`。建议使用 BFS，因为 BFS 的话停止搜索更容易实现。

根据上面的分析，我们还可以优化一下，只要我们发现经过了不同地图单元中的同一个区块就可以停止搜索并输出 `Yes`，由抽屉原理也能知道这样搜索的区块数一定不会超过 $s+1$ 个，因此可以保证复杂度。

具体的代码实现上：

- 标程采用的是优化后的做法。
- 标程用 **区块所在的地图单元的位置** 和 **区块在地图单元中的位置** 两个二维坐标记录点的位置并进行 BFS。

- 传统的 BFS 一般会维护一个 `vis` 数组记录每个点是否访问过，而标程额外维护了一个 `vis_block` 数组，其中 `vis_block[x][y]` 表示上次访问到地图单元中第 `x` 行第 `y` 列的区块是在哪个地图单元。
- 在 BFS 过程中，每当尝试把一个新的区块加入队列时：
  - 检查之前是否到过不同地图单元中相同位置的区块，如果是则停止搜索并返回 `Yes`；
  - 检查之前是否到过相同地图单元中相同位置的区块，如果是则不入队；
  - 否则将区块入队并记录 `vis` 和 `vis_block`。

现在考虑加上修改操作以后怎么做。

可以发现，答案序列一定是连续的一串 `Yes`（可能没有）后跟着连续的一串 `No`（也可能没有），因为把矿洞区块改成岩石区块后，如果原来不能抵达无穷多个矿洞区块，现在也一定还是不能，因此 `No` 后不可能出现 `Yes`。

这样，我们只要二分答案序列中 `Yes` 与 `No` 的分界线就可以快速求出所有答案了。注意二分的边界，分界线的位置一共有 $q+2$ 种可能。同时，每次求解前要记得清空数组。