Dancing Links

舞蹈链

问题：

集合拥有个成员，现在集合有个子集。在个子集中选择一些组成集合，使中包含的成员可以覆盖集合，即中所有成员都属于中的某个子集。

重复覆盖：集合中的任意成员（允许同时属于两个以上的子集）。例如集合，在子集、、中选择即可重复覆盖。

精确覆盖：集合中的任意成员属于且只属于中的一个子集，不能出现不属于中的任何子集，或者同时属于中两个及以上的子集。例如集合，在子集、、中选择即可精确覆盖。

给定集合和个子集，求其重复覆盖和精确覆盖。

重复覆盖解法：

遍历集合中每个成员，若其尚未被包含在中，则在个集合中寻找一个包含的子集加入中，重复该步骤即可获得重复覆盖。

精确覆盖解法：

对于每个元素（），所有包含它的子集都是一种可能的选择。对于包含元素的所有子集（共个），依次尝试选择每个子集（）作为我们的精确覆盖。当选择子集时，它除了之外还会包含其他元素，设其他元素的集合为，那么所有其他包含元素的子集，都与子集冲突，因此这次选择后，需要将和所有与它冲突的集合都删除。



上图中，当为了元素选择子集后，由于也包含、，所以需要删除所有包含、元素的其他子集，即、。经过这次操作后，被覆盖的元素有，被选中作为精确覆盖的子集为，被删除的子集有。然后重复上述操作，继续考察下一个元素，直到覆盖所有元素，则找到一组精确覆盖；若所有子集都被删除掉时，却无法覆盖到所有元素，则说明最近的一次选择是错误的，这时我们就放弃子集，并放弃之前的操作（被覆盖的元素中去掉，精确覆盖中去掉子集，被删除的子集中去掉），然后考虑下一个包含的子集。

求精确覆盖的算法称为X算法，将集合中的个成员看作列，将个子集看作行，组成一个的矩阵。若子集（其中）包含某个成员（其中），则；若不包含则。

例如对于集合，它有个成员，还有个子集、、、、、的情况，如下图所示：



1. 从1开始遍历集合中每个成员，对于成员1，遍历所有子集，找到第一个满足的子集，即时有，选择该子集作为精确覆盖中的一个子集，已经覆盖的成员有，中已经包含的成员其他子集不能再出现，因此删掉其他包含的子集、、、、，将也删掉；



1. 这时矩阵中所有子集都被删除，成为空矩阵，但并没有完全覆盖集合中所有成员，因此(1)的选择是失败的，撤销(1)中的所有操作。继续从1开始遍历集合中每个成员，对于成员1，遍历所有子集，找到第二个满足的子集，即时有，选择该子集作为精确覆盖中的一个子集，已经覆盖的成员有，中已经包含的成员其他子集不能再出现，因此删掉其他包含的子集、、，将也删掉；



1. 从2开始遍历集合中剩下的成员，对于成员2，遍历剩余子集，找到第一个满足的子集，即时有，选择该子集作为精确覆盖中的一个子集，已经覆盖的成员有，中已经包含的成员其他子集不能再包含，因此删掉其他包含的子集（没有找到），将也删掉；



1. 从5开始遍历集合中剩下的成员，对于成员5，遍历剩余子集，找到第一个满足的子集，即时有，选择该子集作为精确覆盖中的一个子集，已经覆盖的成员有，中已经包含的成员其他子集不能再包含（没有找到），将删掉后矩阵即为空矩阵，并且已经完全覆盖了子集中的所有成员，则精确覆盖的结果为，算法结束；



1. 当算法进行到矩阵为空矩阵，但集合中所有成员并没有被完全覆盖的情况时，说明某一次的子集选择有错误，将该次选择的操作进行恢复，并寻找下一个覆盖要求的子集，继续尝试，直到找到精确覆盖；

回溯法的递归结束条件是矩阵为空，每次递归时选择矩阵中的一列（其中），遍历矩阵中的所有子集（行），找到一子集（其中）满足，选择该子集（行）。由于精确覆盖的要求，其他包含该子集中任意成员的子集，都不能再选择，将其删掉，并将子集也删掉。重复这个操作直到将矩阵删空，检查矩阵为空时是否集合的所有成员都被覆盖到。在选取包含某个成员的子集时，可能有多个选择，若选择其中一个子集无法最终将集合完全覆盖，则在递归函数中返回这一层，尝试其他子集，直到找出精确覆盖。

十字链表是一种方便删除矩阵中的行列、以及恢复行列的数据结构。每个节点有上下左右4个指针指向周围的节点。现在将上文的集合、6个子集以及5个步骤，用十字链表的形式重复一遍。建立十字链表时需要额外对每一列添加头节点，并添加一个总的head节点连接所有列的头节点，如图所示：



1. 选取head节点右边的节点1（第1列），在第1列中从上到下依次考虑每个子集，看是否最终可以得到精确覆盖，第1列有2个选择、，首先尝试选择。根据上文可知，目标是将包含成员的所有子集都删除掉，即删除、、、、、。在十字链表中这个过程分为以下几个步骤来依次进行；
2. 将的所有成员、、、首先删除；



1. 再依次将属于列上的所有节点，以及其所在子集（行）上的所有节点，都删除掉；
2. 这时矩阵为空，所选子集为，覆盖的成员为，没有完全覆盖，因此选择错误，恢复列的所有元素，然后继续尝试第1列（节点1）的下一个节点，直到找到精确覆盖；

舞蹈链算法的时间复杂度与递归的时间复杂度一样，为。