《计算机算法设计与分析》第六章作业

姓名：任宇　　学号：33920212204567

算法实现题 6-1

6-1　最小长度电路板排列问题。

问题描述：最小长度电路板排列问题是大规模电子系统设计中提出的实际问题。该问题的提法是，将 n 块电路板以最佳排列方案插入带有 n 个插槽的机箱中。n 块电路板的不同的排列方式对应不同的电路板插入方案。

设 $B=\{1, 2, \cdots, n\}$ 是 n 块电路板的集合。集合 $L=\{N_1, N_2, \cdots, N_m\}$ 是 n 块电路板的 m 个连接块。其中每个连接块 N_i 是 B 的一个子集，且 N_i 中的电路板用同一根导线连接在一起。在最小长度电路板排列问题中，连接块的长度是指该连接块中第 1 块电路板到最后 1 块电路板之间的距离。例如，设 $n=8$，$m=5$，给定 n 块电路板及其 m 个连接块如下：

$$B=\{1, 2, 3, 4, 5, 6, 7, 8\};\qquad L=\{N_1, N_2, N_3, N_4, N_5\}$$

$$N_1=\{4, 5, 6\};\quad N_2=\{2, 3\};\quad N_3=\{1, 3\};\quad N_4=\{3, 6\};\quad N_5=\{7, 8\}$$

这 8 块电路板的一个可能的排列如图 6-1 所示。

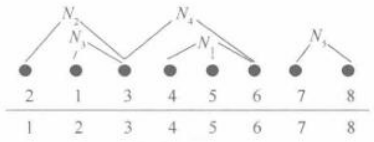


图 6-1　最小长度电路板排列

试设计一个队列式分支限界法找出所给 n 个电路板的最佳排列，使得 m 个连接块中最大长度达到最小。

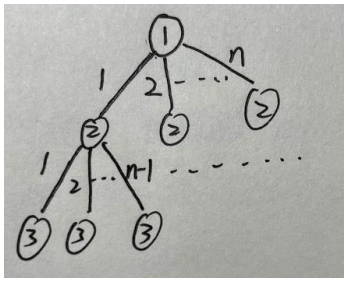
算法设计：对于给定的电路板连接块，设计一个队列式分支限界法，找出所给 n 个电路板的最佳排列，使得 m 个连接块中最大长度达到最小。

数据输入：由文件 input.txt 给出输入数据。第 1 行有 2 个正整数 n 和 m（$1\leqslant m, n\leqslant 20$）。接下来的 n 行中，每行有 m 个数。第 k 行的第 j 个数为 0 表示电路板 k 不在连接块 j 中，为 1 表示电路板 k 在连接块 j 中。

结果输出：将计算的电路板排列最小长度及其最佳排列输出到文件 output.txt。文件的第 1 行是最小长度；接下来的 1 行是最佳排列。

答：

解空间定义：本题是一个典型的排列树问题，解由所有可能的电路板排列组成，总共有 n! 种不同的排列方式，如图：



算法设计：采用队列式分支限界法解决本题，首先创建一个 FIFO 队列，用于存储待处理的结点。每个结点代表一个部分解，即部分电路板的排列。扩展结点时，每次从队列中取出一个结点，然后基于该结点生成新的结点，每个新结点表示在前一个结点的基础上，为另一个未排列的电路板选择一个位置放入。

限界：对每个新生成的结点进行判断，如果一个结点的当前最大连接块长度已经超过当前找到的最佳解的最大长度，就舍弃这个结点。

在扩展结点后，将符合条件的新结点加入队列尾部。

具体步骤如下：

1. 初始化：将一个空的初始结点放入队列。

2. 循环处理：
   - 如果队列为空，算法结束。
   - 取出队列中的第一个结点。
   - 扩展这个结点：为下一个未排列的电路板选择每一个可能的位置，生成新的结点。
   - 对每个新结点计算最大连接块长度，并更新最佳解。
   - 将符合条件的新结点加入队列尾部。
3. 结束：当队列为空时，算法结束。输出最佳电路板排列和相应的最小连接块长度。

时间复杂度分析：使用队列式分支限界法时，时间复杂度仍然取决于解空间的大小，即 n!，所以算法时间复杂度式 O(n!)。尽管队列式分支限界法按照生成顺序处理结点，而非基于优先级，但是有效的限界条件依然可以减少实际需要探索的结点数。

算法实现题 6-2

6-2 最小权顶点覆盖问题。

问题描述：给定一个赋权无向图 $G=(V,E)$，每个顶点 $v\in V$ 都有权值 $w(v)$。如果 $U\subseteq V$，且对任意$(u,v)\in E$ 有 $u\in U$ 或 $v\in U$，就称 U 为图 G 的一个顶点覆盖。G 的最小权顶点覆盖是指 G 中所含顶点权之和最小的顶点覆盖。

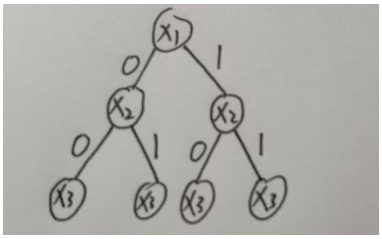
算法设计：对于给定的无向图 G，设计一个优先队列式分支限界法，计算 G 的最小权顶点覆盖。

数据输入：由文件 input.txt 给出输入数据。第 1 行有 2 个正整数 n 和 m，表示给定的图 G 有 n 个顶点和 m 条边，顶点编号为 $1, 2, \cdots, n$。第 2 行有 n 个正整数表示 n 个顶点的权。接下来的 m 行中，每行有 2 个正整数 u 和 v，表示图 G 的一条边(u,v)。

结果输出：将计算的最小权顶点覆盖的顶点权之和以及最优解输出到文件 output.txt。文件的第 1 行是最小权顶点覆盖顶点权之和；第 2 行是最优解 x_i（$1\leqslant i\leqslant n$），$x_i=0$ 表示顶点 i 不在最小权顶点覆盖中，$x_i=1$ 表示顶点 i 在最小权顶点覆盖中。

答：

解空间定义：这是一个典型的子集树问题，对于图中的每个顶点，可以选择包含它或不包含它，因此问题的解空间树是一个 n+1 层的完全二叉树，如图：



算法设计：使用优先队列式分支限界法解决这个问题，首先需要创建一个优先队列用于存储待处理的结点。每个结点代表一个部分解，即包含部分顶点的选择。扩展结点时，从优先队列中取出一个结点，然后基于该结点生成新的结点。每个新结点代表在前一个结点的基础上，为另一个未决定的顶点选择包含或不包含。

限界：对每个新生成的结点进行评估。如果一个结点的当前顶点权值之和加上

剩余顶点的最小可能权值仍然大于当前找到的最佳解的权值，或者该结点不能形成有效的顶点覆盖，则舍弃这个结点。
优先级：结点的优先级基于当前解的顶点权值之和。优先队列优先处理权值之和更低的结点。
算法的具体步骤如下：

1. 初始化：将一个空的初始结点放入优先队列。
2. 循环处理：
   - 如果优先队列为空，算法结束。
   - 取出优先队列中优先级最高的结点。
   - 扩展这个结点：为下一个未决定的顶点选择包含或不包含，生成新的结点。
   - 对每个新结点计算总权值，并检查是否构成有效的顶点覆盖。
   - 将符合条件的新结点加入优先队列。
3. 结束：当优先队列为空时，算法结束。输出最小权顶点覆盖和相应的总权值。

时间复杂度分析： 在最坏的情况下，算法将探索所有可能的子集，所以时间复杂度为 $O(2^n)$,其中 n 是顶点数量。实际的时间复杂度会比 $O(2^n)$更低，但精确的复杂度取决于限界条件的效果。

算法实现题 6-4

6-4 最小重量机器设计问题。

问题描述：设某一机器由 n 个部件组成，每种部件都可以从 m 个不同的供应商处购得。设 w_{ij} 是从供应商 j 处购得的部件 i 的重量，c_{ij} 是相应的价格。设计一个优先队列式分支限界法，给出总价格不超过 d 的最小重量机器设计。

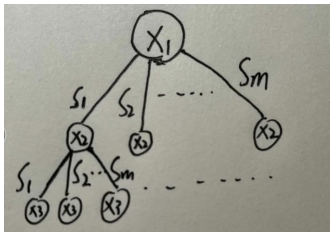
算法设计：对于给定的机器部件重量和机器部件价格，设计一个优先队列式分支限界法，计算总价格不超过 d 的最小重量机器设计。

数据输入：由文件 input.txt 给出输入数据。第 1 行有 3 个正整数 n、m 和 d。接下来的 $2n$ 行，每行 n 个数。前 n 行是 c，后 n 行是 w。

结果输出：将计算的最小重量，以及每个部件的供应商输出到文件 output.txt。

答：

解空间定义：解空间是由所有可能的部件购买方案组成的。对于每个部件，有 m 种不同的购买选择（即从不同的供应商购买），所以总共m^n有种不同的购买组合，这是一颗 n+1 层的 m 叉树：



算法设计：使用分支限界法解决这个问题，首先需要创建一个优先队列，用于存储所有待处理的结点。每个结点代表一个部分解，即部分部件的购买方案。每次从优先队列中取出一个结点，然后基于该结点生成新的结点。每个新结点代表在前一个结点的基础上，为另一个部件选择一个供应商。

限界：对每个新生成的结点进行评估。如果一个结点的当前总价格超过 d，或者当前重量加上剩余部件的最小可能重量仍然大于当前找到的最佳解的重量，则舍弃这个结点。
优先级：结点的优先级基于当前解的总重量。优先队列优先处理总重量更低的结点。
算法的具体步骤如下：
1. 初始化：将一个空的初始结点放入优先队列。
2. 循环处理：
   - 如果优先队列为空，算法结束。
   - 取出优先队列中优先级最高的结点。
   - 扩展这个结点：为下一个部件选择每一位可能的供应商，生成新的结点。
   - 对每个新结点计算总重量和总价格，更新最佳解。
   - 将符合条件的新结点加入优先队列。
3. 结束：当优先队列为空时，算法结束。输出最佳购买方案和最小的总重量。

时间复杂度分析：每个部件有 m 种可能的分配方式，所以时间复杂度为 $O(m^n)$，其中 m 是供应商数量，n 是部件数量，但是有效的限界条件可以减少实际需要探索的结点数，从而在实际应用中降低时间复杂度。

算法实现题 6-5

6-5 运动员最佳配对问题。

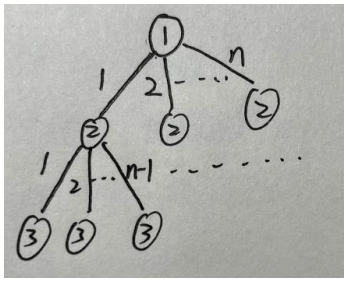
问题描述：羽毛球队有男女运动员各 n 人。给定 2 个 $n\times n$ 矩阵 P 和 Q。$P[i][j]$是男运动员 i 和女运动员 j 配对组成混合双打的男运动员竞赛优势；$Q[i][j]$是女运动员 i 和男运动员 j 配合的女运动员竞赛优势。由于技术配合和心理状态等因素影响，$P[i][j]$不一定等于 $Q[j][i]$。男运动员 i 和女运动员 j 配对组成混合双打的男女双方竞赛优势为 $P[i][j]\times Q[j][i]$。设计一个算法，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

算法设计：设计一个优先队列式分支限界法，对于给定的男女运动员竞赛优势，计算男女运动员最佳配对法，使各组男女双方竞赛优势的总和达到最大。

数据输入：由文件 input.txt 给出输入数据。第 1 行有 1 个正整数 n（$1\leqslant n\leqslant 20$）。接下来的 $2n$ 行，每行 n 个数。前 n 行是 p，后 n 行是 q。

结果输出：将计算的男女双方竞赛优势的总和的最大值输出到文件 output.txt。

答：
解空间定义：解空间是由所有可能的男女运动员配对组成的。对于每个男运动员，有 n 种可能的女运动员配对选择，所以总共有 n!种可能的配对方式,因此本问题是一个典型的排列树问题，解集树可以表示为：



算法设计：对于本题，可以使用优先队列式分支限界法，首先创建一个优先队列用于存储待处理的结点。每个结点代表一个部分解，即某些已经确定的男女

运动员配对。从优先队列中取出一个结点，然后基于该结点生成新的结点。每个新结点代表在前一个结点的基础上，为另一个还未配对的男运动员选择一个未配对的女运动员。
限界：对每个新生成的结点进行评估。如果一个结点的当前总竞赛优势加上剩余未配对运动员的最大可能优势仍然低于当前找到的最佳解，那么舍弃这个结点。
优先级：结点的优先级基于当前解的竞赛优势总和。优先队列优先处理总和更高的结点。
具体步骤如下：

1. 初始化：将一个空的初始结点放入优先队列。
2. 循环处理：
   - 如果优先队列为空，算法结束。
   - 取出优先队列中优先级最高的结点。
   - 扩展这个结点：为下一个未配对的男运动员选择每一位可能的女运动员，生成新的结点。
   - 对每个新结点计算竞赛优势总和，更新最佳解。
   - 将有效的新结点加入优先队列。
3. 结束：当优先队列为空时，算法结束。输出最佳配对方案和最大的竞赛优势总和。

时间复杂度分析：在最坏情况下，算法可能需要探索所有可能的配对方式，即时间复杂度为O(n!)，但是有效的限界策略可以减少实际探索的结点数。

算法实现题 6-10

6-10　世界名画陈列馆问题。

问题描述：世界名画陈列馆由 $m \times n$ 个排列成矩形阵列的陈列室组成。为了防止名画被盗，需要在陈列室中设置警卫机器人哨位。除了监视所在的陈列室，每个警卫机器人还可以监视与它所在的陈列室相邻的上、下、左、右 4 个陈列室。试设计一个安排警卫机器人哨位的算法，使名画陈列馆中每个陈列室都在警卫机器人的监视下，且所用的警卫机器人数最少。

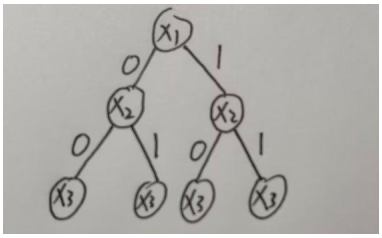
算法设计：设计一个优先队列式分支限界法，计算警卫机器人的最佳哨位安排，使名画陈列馆中每个陈列室都在警卫机器人的监视下，且所用的警卫机器人数最少。

数据输入：由文件 input.txt 给出输入数据。第 1 行有 2 个正整数 m 和 n（$1 \leqslant m, n \leqslant 20$）。

结果输出：将计算的警卫机器人数及其最佳哨位安排输出到文件 output.txt。文件的第 1 行是警卫机器人数；接下来的 m 行中每行 n 个数，0 表示无哨位，1 表示有哨位。

答：

解空间定义：解空间是由所有可能的警卫机器人哨位安排组成的。由于每个陈列室可以选择放置或不放置警卫机器人，因此对于一个 m×n 的陈列馆，总共有$2^{m \times n}$种可能的哨位安排，这是一个子集树问题，可以表示为一棵 m×n+1 层的完全二叉树，如图所示：



算法设计：对于本题，可以使用优先队列式分支限界法，首先创建一个优先队列，用于存储所有待探索的结点。每个结点代表一个部分解，即陈列馆中一部分陈列室的哨位安排。从优先队列中取出一个结点，基于这个结点生成新的结点，每个新结点代表在前一个结点的基础上，为另一个未决定哨位的陈列室选择放置或者不放置警卫机器人。

限界：对每个新生成的结点进行评估，如果该结点不能导致一个有效解，即有陈列室未被监视，或者该结点的警卫机器人数已经超过当前最佳解的机器人数，则舍弃这个结点。

优先级：结点的优先级基于已放置的警卫机器人数和未被监视的陈列室数，优先探索可能性更高且使用警卫机器人更少的结点。

具体步骤如下：

1. 初始化：将初始结点（无哨位安排）放入优先队列。
2. 循环处理：
   - 如果优先队列为空，算法结束。
   - 取出优先队列中优先级最高的结点。
   - 如果该结点表示的哨位安排覆盖了所有陈列室，则记录这个解，并更新当前最佳解。
   - 否则，对于每个未决定的陈列室，创建两个子结点：一个代表在此处放置警卫机器人，另一个代表不放置。计算每个子结点的优先级并加入优先队列。
3. 结束：当优先队列为空或找到最优解时，算法结束。输出最佳哨位安排和警卫机器人的数量。

时间复杂度分析：在最坏的情况下，算法将探索所有可能的安排，所以时间复杂度为$0(2^{m\times n})$，需要注意的是，有效的限界条件可以显著减少需要探索的结点数。