/\*\*Sample Input：输入文件的第一行是两个正整数N和M，表示图像大小。接下来N行每行包含M 个整数，表示每个像素是黑色像素的概率为Pij%。0 ≤ Pij < 100。接下来一行有N个非负整数，表示每一行中黑色像素的个数。接下来一行有M 个非负整数，表示每一列中黑色像素的个数。2 290 1020 801 11 1Hint：共有两种可能的图像：0110和1001前者的出现概率是0.1×0.2=0.02，后者的出现概率是0.9×0.8=0.72，故后者是最优图像。题目大意：有n\*m个点，每个点染色为黑点有一个概率。现在给出条件，要求每一行、每一列要有多少个黑点。求满足条件的最大概率的染色方案。题解：这题其实并不难。概率是要相乘的，我们把它们转成某个数为底的对数，就可以变成加法。建n个x点，S到它们的流量为它们这一行所需要的黑点数，费用为0。建m个y点，它们到T的流量为它们这一列所需要的黑点数。x到y连一条流量为1，费用为概率的对数。求最大流量最大费用。然后判断x->y的弧的流量是否流完了，输出答案\*/#include <bits/stdc++.h>#define abs(a) ((a) < 0 ? -(a) : (a))#define fo(i, x, y) for(int i = x; i <= y; i ++)using namespace std;const int Maxn = 205, INF = 1000000000, INF2 = 100000;const double wu = 0.000000001;int n, m, x, y;int final[Maxn \* 3], tot = 1, S, T;struct edge { int next, to, r; double w;}e[Maxn \* Maxn \* 3];int bz[Maxn \* 3];double dis[Maxn \* 3];int bx[Maxn][Maxn];void link(int x, int y, int r, double w) { e[++ tot].next = final[x], e[tot].to = y, e[tot].r = r, e[tot].w = w, final[x] = tot; e[++ tot].next = final[y], e[tot].to = x, e[tot].r = 0, e[tot].w = -w, final[y] = tot;}void Init() { scanf("%d %d", &n, &m); S = n + m + 1, T = S + 1; /// 5 6 fo(i, 1, n) fo(j, 1, m) { scanf("%d", &x); // cout << x << ' ' << INF2 - log2(x) << endl; if(x != 0) link(i, n + j, 1, INF2 - log2(x)); /// 99993.5 } fo(i, 1, n) { scanf("%d", &x); link(S, i, x, INF2); } fo(i, 1, m) { scanf("%d", &x); link(i + n, T, x, INF2); }}int aug(int x, int flow) { if(x == T) return flow; bz[x] = 1; int use = 0; for(int i = final[x]; i; i = e[i].next) { int y = e[i].to; if(!bz[y] && e[i].r && abs(dis[y] + e[i].w - dis[x]) < wu) { int tmp = aug(y, min(e[i].r, flow - use)); e[i].r -= tmp; e[i ^ 1].r += tmp; use += tmp; if(use == flow) return use; } } return use;}bool change() { double minh = INF; fo(i, 1, T) if(bz[i]) { for(int k = final[i]; k; k = e[k].next) if(!bz[e[k].to] && e[k].r) { minh = min(minh, dis[e[k].to] + e[k].w - dis[i]); } } if(minh < -1) return 0; if(abs(minh - INF) < wu) return 0; fo(i, 1, T) if(bz[i]) dis[i] += minh; return 1;}void End() { fo(i, 1, n) for(int k = final[i]; k; k = e[k].next) if(e[k].r == 0) cout << i << ' ' << bx[i][e[k].to - n] << endl, bx[i][e[k].to - n] = 1; /// 若e[k].r == 0满流， 则将第 i 行 第 e[k].to - n 列 置为1 fo(i, 1, n) { fo(j, 1, m) printf("%d", bx[i][j]); printf("\n"); }}int main() { freopen("costflow.in","r",stdin); Init(); do do memset(bz, 0, sizeof(bz)); while(aug(S, INF)); while(change()); End();}