数据范围判断

一般ACM或者笔试题的时间限制是1秒或2秒。
在这种情况下，C++代码中的操作次数控制在 $10^7 \sim 10^8$ 为最佳。

下面给出在不同数据范围下，代码的时间复杂度和算法该如何选择：

1. $n \leq 30$, 指数级别, dfs+剪枝，状态压缩dp
2. $n \leq 100$ => $O(n^3)$，floyd，dp，高斯消元
3. $n \leq 1000$ => $O(n^2)$，$O(n^2logn)$，dp，二分，朴素版Dijkstra、朴素版Prim、Bellman-Ford
4. $n \leq 10000$ => $O(n * \sqrt{n})$，块状链表、分块、莫队
5. $n \leq 100000$ => $O(nlogn)$ => 各种sort，线段树、树状数组、set/map、heap、拓扑排序、dijkstra+heap、prim+heap、Kruskal、spfa、求凸包、求半平面交、二分、CDQ分治、整体二分、后缀数组、树链剖分、动态树
6. $n \leq 1000000$ => $O(n)$, 以及常数较小的 $O(nlogn)$ 算法 => 单调队列、 hash、双指针扫描、BFS、并查集，kmp、AC自动机，常数比较小的 $O(nlogn)$ 的做法：sort、树状数组、heap、dijkstra、spfa
7. $n \leq 10000000$ => $O(n)$，双指针扫描、kmp、AC自动机、线性筛素数
8. $n \leq 10^9$ => $O(\sqrt{n})$，判断质数
9. $n \leq 10^{18}$ => $O(logn)$，最大公约数，快速幂，数位DP
10. $n \leq 10^{1000}$ => $O((logn)^2)$，高精度加减乘除
11. $n \leq 10^{100000}$ => $O(logk \times loglogk)$，k表示位数，高精度加减、FFT/NTT

前缀和

差分

3729.改变数组元素

797.差分

798.差分矩阵

100.增减序列：这个题还有一点问题，对于差分数组b[1]位可以用来确定数列的种类有点不理解

二分

789.数的范围　　学到了二分的两个模板，但是对于模板的使用还不是很了解

1221.四平方和　　学到了对结构体的排序，需要重载“<”号，里面又一次用到了二分查找，但是对于边界的确定还不是很熟练。

用空间换时间，将枚举的结果保存，用作下一次查找的数据，查找可以用二分，也可以哈希表

```
struct Sum{
    int s,c,d;
    //重载<，用于结构体排序
    bool operator< (const Sum& t){
        if(s!=t.s)return s<t.s;
        if(c!=t.c)return c<t.c;
        return d<t.d;
    }
}sum[N]; //用来存每一种组合情况
```

1227.分巧克力　切巧克力的边长越大，能切出来的数量越少，这种变化符合二分查找的条件，运用第二个模板，查找右边界，小于等于目标值的最后一个数，这样能保证在满足小朋友数量的情况下使得巧克力的面积最大。

双指针

3768.字符串删减。我用的是模拟的方法，当遇到连续'x'的情况或者其他情况，遇到连续的就统计'x'的个数直到遇到其他情况，将统计的个数清零，并且累加需要删除的字母的个数，然后改变左右边界。最后特判一下，因r超过总长度而没有把最后一段需要删除的字母数量加上。

799.最长连续不重复子序列。也是利用l、r双指针，r一直向右移动，直到找到重复的数，表示已经达到这种情况的最长序列，更新最大值，然后固定r，l向右移动，直到区间内没有数字重复，继续移动r，直到序列被遍历完成。

800.数组元素的目标和。因为是升序的数据，涉及到查找，可以使用二分查找；或者使用哈希，将每个数是否出现存在哈希表中，查找时间复杂度O(1)；也可以使用双指针，枚举其中一个数组，因为每个数组都是升序的，所以一旦a[i]+b[j]>x，那么后面的数就可以不用枚举了。

2816.判断子序列。双指针逐个枚举子序列中的值，判断母序列中是否有值

1238.日志统计。利用双指针，我是用结构体存数值对，对id进行排序，id相同根据ts升序，然后对每一个id计算所有时间段，如果满足“热帖”条件就输出，然后判断下一个，但是这种方法时间复杂度过大，通过13/14的数据，最后一个数据超时了。题解做法：对ts进行排序，对时间段进行枚举，利用双指针，滑动窗口，统计合理时间段内每个id的点赞数，如果规定的时间段超过合理值，就移动左边界，并同时修改移出范围的id点赞数的变化（cnt[**logs[j]**]--，将点赞数减1，因为左边界缩小了），利用滑动窗口来减少重复统计时间段内的id的

点赞数。

还学到了写代码的习惯,减少代码的长度，并且pair默认是对first进行排序，就可以不用结构体内重载<，也是减少代码量

```
typedef pair<int,int> PII;
#define x first
#define y second
```

1240.完全二叉树的权值。这道题比较简单，就是注意是完全二叉树，不是满二叉树。后面一直过不了的原因竟然是变量定义的范围太小了，应该是long long 类型。之前都没注意过，都是看感觉的。

递推

3777.砖块。把全是白色和全是黑色两种情况都讨论一边，只要有满足条件的就打印出来。

1208.翻硬币。和3777.砖块这道题做法类似，比较简单。

95.费解的开关。通过第下一行来修改上一行的灯状态，保证上一行都是满足灯亮的状态的，所以说，只要确定第一行的状态，后面几行灯的状态都是确定的，然后再判断最后一行灯是不是全亮的，因为最后一行的灯是需要靠下一行修改的。通过枚举第一行所有修改的情况，总共有32种，然后递推每种情况下是否满足全亮。

题目收获：

```
//1.
memcpy(gcopy,g,sizeof(g));//拷贝数组，一维二维都可以 #include<cstring>
//2.
//通过设置两个向量可以方便的表示数组的上下左右
int dx[5]={0,-1,1,0,0};
int dy[5]={0,0,0,-1,1};//上下左右
//3.位运算的运用，可以将灯的状态转换成二进制运算
```

递归

1497.树的遍历。运用递归，用后序遍历数列得到根节点的位置，然后递归左右子树。之前看到一句话，就是你在理解递归的时候，不要想着深入去理解栈的情况，因为层数太多了会晕，而是相信程序能完成这件事 。

97.约数之和。涉及到快速幂和筛质数的算法，另外通过递归分治的方法，求解等比数列，这是一种通用的方式。

98.分形之城。感觉很难，迟点再来看

并查集

836.合并集合。模板题，学会了并查集，就是查找两个元素是否属于同一个 集合

1249.亲戚。这道题和上一题一样，唯一要注意的是，需要加上ios优化。然后在打印时使用puts("Yes")，用cout也会超时。

```
//ios优化
ios::sync_with_stdio(false); cin.tie(0); cout.tie(0);
```

837.连通块中点的数量。计算同一个集合中的数量，可以定义一个sum[N],表示某集合的数量，在合并集合的时候，顺便修改sum。

240.食物链。利用并查集，把所有已知正确的关系都加入到一个集合（就是不管是同类、还是谁吃谁），然后维护一个"距离"数组d，用来判断某个点与根节点之间的关系，因为本题中只有3个关系，所以定义如果d=0表示是与根节点同类的，如果d=1表示可以吃根节点，如果d=2表示被根节点吃。然后d数组的维护是通过find()的路径压缩来实现的，每次查找一次，就更新d[x]为x到根节点之间的距离。先求出fa,fb判断是否已经加入过集合（表示x，y的关系可以根据之前正确的关系推出来），这里fx=find(x),已经经过了一次路径压缩，所以此时的d[x]表示的是x到fx之间的距离（可以确定与根节点之间的关系），然后如果还没有加入过集合，就需要把fx和fy归为一类，相当于出现了一个新的根节点，然后需要更新d[fx]或d[fy]（选择不同的节点当根的话，更新的d不一样），表示dx或fy到根节点之间的距离，然后等下一次查找x的根节点的时候，就会将d[x]更新（更新为x到新的根节点之间的"距离"）**注意：**在比较两个节点到根节点的距离的模的时候，不能简单的先取模在比较是否相等（表示同类），因为可能会有负数的情况。**这道题还提到了并查集可以用来维护额外信息，这一点体会的还不是很深刻（难道是因为并查集是一棵树？不知道不知道hh）**

涉及到数论中的同余定理,例如需要比较d[x]%3是否等于d[y]%3，因为"距离"会有负数存在，所以移项，转化为(d[x]%3-d[y]%3)=(d[x]-d[y])%3　（这里用到了数论的**取模运算**）

238.银河英雄传说。又一次理解了并查集用来维护额外信息的作用，可以用来记录集合中的元素个数，也可以维护节点到根节点之间的距离，可能还有别的，但这道题用到了这两个作用，然后在进行路径压缩的时候修改需要维护的额外信息

快速幂

875.快速幂。模板题，快速计算a^k mod p的值。详细见 Algorithm.md，学会了快速幂算法和乘法的取模运算

876.快速幂求逆元。模板题，学会了逆元的概念。但是快速幂是基于p是质数的条件下，利用费马小定理求解。具体见笔记

1289.序列的第k个数。用快速幂求模

1290.越狱。容斥定理，所有情况-不相同的情况=相同的情况。这里要注意，在两个取模后的数相减，可能会出现负数，需要在加上mod然后再取模把数变成正的。**c++负数取模看的是左边的数**，如果左边的数是负数，比如-3%4，那么它的结果是-（3%4）=-3，不同于数学中计算模，所以我们需要+4，让结果为正。

3625.幂次方。直接快速幂

组合计数

885.求组合数I。递推法求排列组合

886.求组合数II。公式法+快速幂求排列组合。求逆元的预处理还没明白。。 第二天明白哈哈，具体见笔记

1307.牡牛和牝牛。好难啊，还有dp在里面，有点理解不了，先放弃了。

1310.数三角形。暂时先放弃数论了，好复杂，学dp去了。

哈希

2058.笨拙的手指。过了13/16的数据，枚举每一位二进制数，然后记录三进制的每种变化情况，并用一个数组保存，这样子在后面就可以直接访问数组获得将修改的结果值。终于过了，好多需要注意的地方，最终要的一点就是“前导零”的问题，意思就是如果一个正确的数，二进制/三进制最高位是0，那么就一定不会写错，因为最高位的0一般不写，所以在判断最高位的时候需要去掉这些情况。然后就是只有一位的情况，需要单独判断“前导零”。学到了利用哈希表存数据，我这道题用的是数组，可以使用unordered_set这个容器来存放

840.模拟散列表。在模拟散列表的时候，使用拉链法，需要用到用数组模拟单链表，具体见algorithm.md。然后这道题还没有解决，**总感觉不是重点，先跳过去。**

841.字符串哈希。又学到了关于字符串的分析，通过求解字符串前缀哈希值的方式，利用哈希值来比较字符串中子串的情况，可以不用KMP算法，感觉很牛逼这方法，记一下！！具体见algorithm.md。

单调队列

299.裁剪序列。困难题先放一放哈哈

830.单调栈。所谓单调栈，就是在暴力的基础上加以改进，本来找左边第一个比它小值需要遍历所有的数，但是现在通过一些方式减少了搜索区间中的数。删除那些不会被判断的数。单调栈可以用来寻找左边或右边第一个大于或小于的某数的值

154.滑动窗口。与单调栈的思想类似，都是维护一个单调的序列，以求解窗口内的最小值为例，维护一个单调队列，在队列中加入一个新的数时，如果新的数比队列中的数小，则可以将前面那个数剔除（因为最小值不可能会取到比新的数还小的数），最后队列是单调递增的，最小值就是队首元素。队列中的值数元素的下标，我感觉是非常巧妙的，我一开始用元素本身的值存，发现难以判断最小的值是否该滑出窗口了。

135.最大子序和。做单调队列的题，首先用队列来维护一个集合，然后把一定没有用的元素删掉，这样子就会发现队列变成单调的了，然后取**最值**，这样子找最值的时间复杂度是O(1)的。首先是计算前缀和，然后需要求解给定区间的和，首先固定右边界，然后维护一个队列，用来找到最小值，这样子两数相减就会是最大值

1089.烽火传递。这里主要的算法是动态规划，能理解为什么状态转移方程是这样的，但不知道思考的过程该是什么样的，dp[i]=a[i]+min(f[j])（i-m<=j<=i-1），状态表示的意思是第i个烽火台点亮后，1~i个烽火台成功传递消息的最小代价=点亮烽火台i的代价+前面m个代价数组dp中最小的一个。

KMP

831.kmp字符串。学了kmp字符串模式匹配算法，我发现虽然我知道了算法的理论是怎么样的，但是用代码实现起来总是非常的冗余，代码量大，感觉还是需要背一下模板。

Trie

143.最大异或对。学会了用字典树存储二进制数，在判断异或的最大值时，需要两层循环，外循环遍历所有数，内循环通过字典树来计算所有值中与外循环枚举的值能够得到的最大异或值。

835.Trie字符串统计。与143题类似，就是在统计某个字符串出现的个数，需要定义一个cnt[N]数组来统计。

3485.最大异或和。非常的巧妙，将求区间内数的异或值，通过前缀和的思想，转化为求最大异或对的问题。对于Trie树中的其中某个数，定义了一个cnt数组用于统计经过该结点的数的个数。我发现cnt数组放在不同的位置可以统计不同的内容，放在for循环中，可以对每个结点的情况进行统计，放在for循环外，可以对某个数进行统计（因为在for循环外表示已经到达数的叶子结点，可以标记该数字是否已经出现过了）。

BFS

844.走迷宫。

845.八数码。感觉bfs本质上也是暴力的做法，这道题的解法就是把所有的情况全部都列举一遍，然后找出最少次数能恢复原样的情况。还是在队列中进行判断，把所有情况都加入到队列中。因为每种情况用字符串表示，所以用unordered_map存当前情况到原始情况的“距离”，一般都是用一个数组的。我一开始用map超时了，后来换成**unordered_map**才通过了，可见后者的效率要高一些，以后要注意。还有就是在判断矩阵中的上下左右时，可以用一个小技巧，用向量表示：**int dx[]={-1,1,0,0},dy[]={0,0,-1,1};**

1233.全球变暖。进行以此bfs相当于查找一次连通块，每一个连通块就是一个岛屿。最后出现段错误了，也不知道为啥，先不管了。

1562.微博转发。在bfs的基础上，需要记录层数信息，然后判断是否需要加入队列，我用的是**临界矩阵**的方式，但是该矩阵十分稀疏，适合用**邻接表**做。

DFS

842.排列数字。dfs强调顺序，通过递归加回溯的方式枚举所有的情况。

843.n皇后问题。解法一：就是最原始的方式，枚举每一个格子，分两种情况进行递归，放皇后或者不放皇后。这里用到技巧，就是在放置皇后时，需要将行列和对角线都进行标记，这里可以用一维数组的方式进行表示，**bool row[N],col[N],dg[N],udg[N];**行列比较好理解，对角线的话，就相当于用不同的x,y对应不同截距的直线y=x±b，这样就可以区分不同的对角线，**row[x]，col[y]，dg[x+y]，udg[x-y+n]**，这里对角线x-y还要加上n是因为方式数组下标出现负数而进行的偏移。

1209.带分数。纯暴力，先想842.排列数字问题一样，枚举出所有数，然后将数分割成三个部分计算对应的数。

165.小猫爬山。感觉对dfs搜索又有了一些理解，就是枚举所有情况，然后对不可能的情况进行减枝。就像这道题，在一开始判断车的数量是否已经超过最小值，如果超过了，那就直接返回上一层，不用再接下来搜索了。

3502.不同路径数。起点可以是任意位置，所以对每个起点进行dfs，因为可以走重复的路，意思是说可以绕圈，所以不需要加标记数组st判断点是否走过。

拓扑排序

848.有向图的拓扑序列。每次查找入度为0的点加入队列。这道题需要用到**邻接表**存矩阵

1191.家谱树。也是模板题，通过手写队列，如果队列长度和点的数量相同，说明存在拓扑排序；如果数量不相同，则表示存在环

456.车站分级。用到什么差分约束和拓扑排序，不是很懂，先放一放吧

Dijkstra

903.昂贵的聘礼。这道题主要就是把每个物品的价格已经兑换关系抽象出一幅图。这道题用到了一个技巧，构造了一个**虚拟源点**，从原点到每个点的权重表示每个点的原价，Djikstra求解的是单源点问题，通过设置虚拟源点就可以直接使用该算法

342.道路与航线。这道题好像和我想的直接用Dijkstra不太一样，**边权为负**，要用SPFA算法，先去学一下再回来看。

1488.最短距离。这道题有点特殊，就是如果按暴力做法的话，有多个商店，需要对每个商店都执行Dijkstra算法，然后才能选出每个商店离他最近的商店的距离，这样会超时。我们通过构造了一个**虚拟源点**，并连出一条边到每个商店，距离为零，把这个虚拟源点作为算法的起始点，求解到所有商店的距离。真的很巧妙！！！

SPFA

851.spfa求最短路。

852.spfa判断负环。又一次使用了**虚拟源点**，因为判断负环，需要把每个点作为起点执行一遍spfa判断是否有负环。这是我们就利用的一个虚拟源点，连接到所有点一条权重为0的边，这样子执行一遍spfa就可以得出这个**新图**是否有负环了，虚拟源点转变的起点，把多起点的图转换为只有一个起点的新图。

341.最优贸易。需要用到反向建图的技巧，首先需要从1出发，找到买入水晶球的最小值；然后终点能够到达n，找到水晶球的价格的最大值。再进行第二步时，有多个起点可以到达n，要求解从不同起点出发能够卖出的水晶球的价格，所以利用**反向建图**，将多起点转换为单起点，然后利用spfa算法求解。**为啥不能用虚拟源点，感觉也能想通?**

Floyd

854.Floyd求最短路。模板题

1125.牛的旅行。也是直接套用floyd，但是题目还有有点没读懂，过了一部分的数据。

最小生成树

858.Prim算法求最小生成树。

859.Kruskal算法求最小生成树。

1140.最短网络。直接套模板

1141.局域网。这道题也是直接套kruskal的模板，但是就是需要转换一下思路，他要求得的是去掉一些边，使得这些边的权值和最大，而且依然为连通图。也就是说求解最小生成树，因为总权值和不变，所以可以使去掉的边权值和最大。

3728.城市通电。这道题就是求解每一个连通块的最小生成树，一个发电站就相当于一个连通块，但是求解多个连通块的最小生成树，难以直接套用算法。这里用到了一个巧妙的方式，构造**虚拟源点**（之前求解多起点最短路径时也有用到过），就相当于用一个虚拟源点，将所有连通块连接起来，虚拟源点到城市的权重是在该城市建设发电站的开销，然后直接套用prim或者kruskal算法求解。本来发电站的权重是在点上，通过引入虚拟源点的方式，将权重转移到边上。但是很奇怪，代码都和y总写的基本一样了，过不了，找不到错误。。。

最近公共祖先

3555.二叉树。计算任意两个点之间的最短距离。可以使用最短路算法、因为边权为1，也可以使用bfs算法。因为是二叉树，每个点之间的路径唯一，可以通过计算两个点之间的公共祖先，来得到两点之间的距离（首先需要得到每个点到根节点之间的距离）。

1172.祖孙询问。具体间公共祖先笔记。

1171.距离。该题就是添加了边的权重，但是有好多细节需要注意，改了一下午。。。看似一模一样的代码，md就是过不了。在初始化dist距离数组的时候，不需要初始化为0x3f。

二分图

860.染色法判断二分图。模板题，染色法就是用dfs或者bfs对顶点进行染色，如果染色出现冲突则标识**存在奇数条边的环**，不存在二分图。这里学到了一个知识，就是这道题有多个连通块的情况，所以需要在bfs或者dfs外再加一个for循环（之前一直很疑惑多连通块该怎么做题）。

861.二分图的最大匹配。模板题，学会了匈牙利算法（NTR算法hh）

257.关押罪犯。这道题是二分+染色法二分图解决。这里学到了一个技巧，就是给定的一个原题并不是一个二分图，但是我们可以通过加一些判断来隐藏掉一部分边，，然后判断子图是否为二分图。这道题就是利用了这个方式，因为要让同一个监狱的冲突事件的影响最小，要让影响大的加入到不同的监狱，也就是**构造二分图**，通过二分枚举影响力的可能性，让<=mid值的罪犯加入到同一个监狱，也就是说忽略这条边进行二分图的判断。

372.棋盘覆盖。这道题唯一的难点就是题目看似最大匹配毫无关系，但是可以用最大匹配来做。**首先利用要利用匈牙利算法的前提是这个图是一个二分图，且二分图的点集能够区分。**然后可以发现这个矩阵要求我们求解的是可以放多少块多米诺骨牌，骨牌之间不相重叠。每个相邻的格子相当于是一条边。然后通过染色法可以推断出，相邻点相连的图是一个二分图，目测一下hh。

筛质数

868.筛质数：学到了三种筛质数的方式：1.传统方式，筛除每个数的倍数，剩下的就是质数。2.埃及筛，筛除每个质数的倍数。3.线性筛，每个合数只会被自己的最小质因子筛掉（还有点不是很理解）

867.分解质因数：模板题，学会了试除法，时间复杂度为O(logn)~O(根号n)之间

1292.哥德巴赫猜想。先用线性筛法晒出范围内的所有质数，然后从小到达枚举即可。我一开始是从大到小枚举的，然后为了找到小于x的最右边的一个数，还用了二分查找。。

196.质数距离。每个合数都一定存在1-50000之间的质因子。任意一个大于1的自然数N，如果不为质数，那么N一定可以唯一分解为有限个质数的乘积。这道题需要求出区间内的质数，如果用线性筛会超时，这道题用到了一个理论，**如果一个数n是合数，那么必然存在两个因子，其中一个因子d<=根号n**，所以数据范围是2^31-1，所以我们只需要找出1~50000区间中的所有质数，因为合数的质因子一定在范围内，然后用这些质因子筛掉区间内的合数，留下来的就是质数。

3792.质数问题。很简单，把质数找出来之后枚举。

最大公约数

872.最大公约数。

200.Hankson趣味题。一点都不有趣，迟点在再来看，有点烦了

4309.消灭老鼠。轻松拿下，把每个斜率用pair存一下，然后遍历老鼠，看一下斜率是否已经存在。

DP问题

背包DP

2.01背包。模板题，稍微学了一下y总对于动态规划的分析方法，确定集合的状态表示以及集合所代表的意思，然后对集合的状态进行计算，这里是物品的拿去，所以将集合分为拿取某物品和不拿取某物品。

3.完全背包问题。每个物品可以放无限个数量，具体推到看笔记

9.分组背包问题。每组只能选一个物品放入背包，f[i,j]的集合含义变为前i组容量小于等于j的背包最大价值。

4.多重背包。 n<=100，可以直接暴力做法，O(N^3)

5.多重背包II。n<=1000，暴力做法会超时，将多种物品拆成多类物品，转化为01背包问题。

1214.波动数组。dp问题是真抽象，对于数列问题要先分析问题，列出数列表达式，然后看一下能不能转化问题求解。学到了一个方式，关于c++负数求模会的负数，所以求**a%b=(a%b+b)%b**就可以得出正确结果

3382.整数拆分。对于背包问题的dp还是不太熟练，只能记住背包的模板，换个壳就不知道了。对于背包dp，本质上就是排列组合问题，问选择哪些数，使得满足某个条件。对于正数拆分是一个完全背包问题，每个数都能选无数次。

线性DP

1051.最大的和。

898.数字三角形。拿下，注意以下边界的特判。但是y总的方式是把f数组全部初始化成负无穷，这样就不用对边界进行判断了，因为加上负无穷肯定是最小的。

895.最长上升子序列。f[i]集合的含义为以第i个数结尾的最长上升子序列的长度。

897.最长公共子序列。f[i] [j]表示A字符串1-i和B字符串1-j的最长公共子序列的长度。将f[i] [j]集合分为选或不选a[i]和b[j]。**在求解多个集合去最大或最小值时，集合之间的重叠不会影响结果的正确性，只需要做到不漏即可。**

3656.最大连续子序列。easy

1051.最大的和。先求解连续子序列的最大和，从前往后和从后往前都要求，因为题目要求两个区间。然后需要维护一个max[N]数组用来存前i个数中的连续子序列最大和。然后从前往后枚举，取最大值

区间DP

282.石子合并。拿下。f[i] [j]表示合并区间i~j所需要的最小代价。最外层循环枚举区间长度，第二层循环枚举起点位置，最内层循环枚举区间的分割点，找到代价最小分割点。

320.能量项链。破环成链，然后按照正常区间dp合并来做就行。嘻嘻自己想出来的，舒服了

479.加分二叉树。这道题的限制是树的中序遍历是1~n，所以这道题是一道区间DP问题，f[i] [j]表示结点i~j构成的子树的分数最大值，然后套模板就行。树的前序遍历是先遍历根结点，再遍历左子树，在遍历右子树。

3996.涂色。困难题先不做了，有点浪费时间，先把基础模板学完。

树形DP

285.没有上司的舞会。

323.战略游戏。醉了。。。一开始题目理解错了，一位任意一个点只要有一个士兵看到就行了，原来题目的意思是，每条边之间至少有一个点放士兵。如果按题目的意思，就比较简单了，f[u] [0]表示不在u放置士兵，如果不放置士兵，那么与u相邻的点都要放置士兵。f[u] [1]表示在u点放置士兵。

1220.生命之树。又是题目理解错了，原来就是求一个连通块和的最大值。**f[u]表示以u为根节点的连通块的和的最大值。**然后知道了max函数需要比较大小时需要两个数的类型相同

1079.叶子的颜色。烦死了，和y总都块写的一样了，就是过不了，md滚蛋焯。等我开心了再来过你

状压DP

291.蒙德里安的梦想。状态压缩本质上就是用二进制列举所有状态，并进行状态转移。然后记得要预处理所有的合法状态，以及状态与状态之间转移的合法性。

树状数组

241.楼兰图腾。

差分约束

1169.糖果。

362.区间

1170.排队布局

393.雇佣收银员。这道题有一个坑就是，其中有一个约束条件包含多个变量，并且其中一个变量就是要求解的答案。为了消除其中一个变量，采用了枚举的方式，用常量代替这个变量，跑spfa，如果能满足条件，那么就表示找到答案。

算法提高课

图论

单源最短路

1129.热浪。

1128.信使。

1127.香甜的黄油

1126.最小花费

920.最优乘车。对于边权为1的图可以用bfs来做。然后这道题在输入站点路线的时候，没有告诉我们该路线的站点数量，所以用到了stringstream类来分割读入的站点

```
#include<sstream>
string s;
getline(cin,s);//首先读入字符串
stringstream ssin(s);//可以自动分割字符串的字符串流(也是一个容器)，将字符串送入
stringstream流对象
int cnt = 0,p;
while (ssin >> p) stop[cnt ++] = p;//移出字符串流后会自动类型转换
```

903.昂贵的聘礼。一开始题意理解错了，以为相邻的等级差满足限制即可，原来是任何点之间都要满足，所以需要枚举区间跑dijkstra算法

1135.新年好。这道题亲戚数量很少，完全可以枚举所有亲戚的拜访情况，然后求出最小值

340.通信线路。这道题的本质，就是要找到第k+1大的花费的最小值。可以通过二分枚举所有花费的情况。求出1~N中大于x的边数为y，如果满足y=k，就表示有k条边都比x大，那么x可以作为第k+1大的数。首先定义一种情况，将集合划分为两个部分，一边是满足条件，一边是不满足条件，y总把条件定义为，最短路中大于x的边数y<=k。那么x的右半边都是满足条件的，x的左半边都是不满足条件的，所以需要求解的x位于左边界。

342.道路与航线。学习到了spfa的SLF优化，具体见algorithm.md

341.最优贸易。有一次复习了反向建图，有时候正向不好求解的时候，试一试反过来。这道题目要求最后的终点一定是n，如果正向求解的话，n不一定是可达的，所以通过反向建图求解的方式，把n作为起点，那么求出来的最短路中一定是可以到达n的（因为我们是把n当作起点求解的）

1137.选择最佳线路。直接引入虚拟源点。也可以反向建图，因为终点确定，有多个起点，最后比较一下多个起点取最小值就行（这种方法没试过）

1131.拯救大兵瑞恩。这道题是bfs+状态压缩+动态规划。所谓状态压缩，就是将二维坐标压缩成一维（为了降低空间复杂度）。这道题比基础的bfs找最短路加了限制，需要加上一维判断状态，表示持有钥匙的情况。基础的bfs本质上也是动态规划，dist[i,j]表示到达(i,j)转移的最小次数。而现在加上了限制条件，需要判断是否可以从前一个状态转移过来。dist[i，j，k]表示到达(i,j)状态是k的最短的转移次数。

11340.最短路计数。md思维定式，其实就跑一下bfs，然后顺便统计一下次数就行。平常的bfs每个点只被更新一次，就是当dist[j]>dist[t]+1的时候，如果dist[j]=dist[t]+1，就代表这个点已经被之前的路径更新成最短路的值了，现在是一条新的最短路，只用加上数量就可以了。

383.观光。图论求方案数就有一些DP的影子了。DP前面计算出的子问题的结果，后面基本上不会改变，而且会用到后续的求解中。而DP问题的求解需要求解的问题满足**拓扑序。**而bfs和dijkstra都是满足拓扑序的，以dijkstra为例，每次从优先队列中取出的点，表示已经达到最小值，后续不会再更新，然后用这个点更新邻边。这道题要求解的是最短路和次短路的方案数，都放在dijkstra里面求解就可以了，每次更新，就把点加到队列中；如果没有更新，就表示出现最短路长度相同的边，也就是有另一种方案。

负环

904.虫洞。spfa判断负环

361.观光奶牛。01分数规划+二分。具体见algorithm.md

1165.单词环。都快和答案改成一样了，还是WA。。

差分约束

1169.糖果。md为什么段错误啊焯，第二次做还是段错误，和别人的对照着看来都一样的了，就差最后一个数据焯！！服了滚了滚了

362区间。

最小生成树

1140.最短网络。prim模板题

1141.局域网。kruskal模板题

1142.繁忙的都市。kruskal模板题

1143.联络员。kruskal模板题。找了半天错误，才发现是数组开小了。

1144.连接格点。真是奇怪了，在本地编译运行的时候，卡在while(cin>>x1>>y1>>x2>>y2)这里，还以为这样写有问题。但是放在AcWing上跑就可以出结果，直接AC了。

1146.新的开始。建立超级源点，将点权转化为超级源点到所有点的边权，然后跑prim算法就可以了。

1145.北极通讯网络。这道题用了prim算法，首先跑一边prim，把最小生成树中的边权记录下来，然后排序一下，当k=0或者k=1时，卫星用不了，当k>1时，选择边权最大的用卫星通信（这里是贪心的思想），那么就能让最大的d最小。

346.走廊泼水节。一开始所有点都是一个集合，每次合并时，将集合凑成完全图。假设选中的边的权重为w，那么凑成完全图加的边一定是w+1，这样才不会破坏最小生成树。然后需要添加的边的数量是cnt1*cnt2-1*。（cnt分别是两个集合中的点的数量，然后已经有一条边为w，所以要减去1）。

1148.秘密的牛奶运输。具体见algorithm.md

1170排队布局。对于差分约束的求解， 首先需要找到一个源点可以到达所有边，跑一边spfa判断一下是否有解。然后再求解答案，求解答案的时候不一定要选择一开始的源点，因为一开始的源点只是为了判断是否有解用的。一开一直理解错了，以为一定要用这个源点。

雇佣收银员。

最近公共祖先

1172祖孙询问。轻松，一次就把模板写对了嗨嗨。注意根节点的深度一定要设置为1，这样子就不需要考虑倍增时候的边界问题了，因为跳出边界后，根的深度都是0。

1171.距离。在树中点之间的最短路径都是固定的，维护一下每个点到根节点的距离就行，然后找到最近公共祖先，距离就是dist[x]+dist[y]-2*dist[lca]

356.次小生成树。第一步求解最小生成树，第二步建图， 第三步需要求解最小生成树两点间的路径上的权重最大值和次大值。第四步遍历非树边，判断是否可以替换最小生成树上的边。对于1148.秘密牛奶运输，当时是暴力枚举所有点到其他点的路径，首先是复杂度过高，其次是需要开二维空间，空间不够。所以这里用到了倍增的思想，就是在最近公共祖先求解每个点向上2^j个单位的父节点时，同时维护向上2^j个单位的路径上权重的最大值和次大值。总共有四个值，直接枚举一下求一下最大值和次大值。然后遍历非树边时，找lca（最近公共祖先）的时候，同时记录每次跳跃路径上的最大值和次大值，最后枚举一下找到非树边两个端点路径上边权的最大值和次大值。

352.闇の連鎖。挑战一下困难题，好诡异的名字。这道题和蓝桥14四届省赛类似，树上差分。两个点连上一条附加边，相当于在两个点之间的路径上都加了一条边。题目就变成了求解每条路径上附加边的数量，如果等于0，那么砍掉主要边，任意砍一条附加边就可以。如果等于1，就只有一种方案了。

强连通分量

1174.受欢迎的牛。新学了求解强连通分量的Tarjan算法。缩点构造DAG，最后通过求解点的出度，判断是否是拓扑图的终点。

367.学校网络。利用Tarjan算法缩完点后，构造出DAG。然后统计每个连通分量的入度和出度。

1175.最大半连通子图。先tarjan，在缩点建图，在DAG上DP求解最长路和方案数。对于tarjan求解完后，强连通分量序号的逆序就是拓扑序。具体见algorithm.md

368.银河。差分约束+tarjan+缩点建图+拓扑序跑最长路

二分图

257.关押罪犯。二分枚举答案，判断大于mid的边组成的子图是否为二分图

372.棋盘覆盖。放置最多骨牌的数量相当于是寻找最大匹配。前提是图是二分图，判断该图是否能够被二分染色

376.机器任务。在二分图中，最小点覆盖==最大匹配数。

378.骑士放置。最大独立集的概念，最大独立集=图中总结点数量-最小点覆盖。具体见algorithm.md。

379.捉迷藏。

欧拉回路和欧拉路径

1123.铲雪车。证明该图是一个欧拉图，那么一定存在欧拉回路。

1184.欧拉回路。

搜索

Flood Fill

1097.池塘计数。找连通块的数量。

1098.城堡问题。二进制+找连通块

1106.山峰与山谷。没看题解拿下了。

最短路模型

1076.迷宫问题。bfs，pre[]数组记录路径前驱结点

188.武士风度的牛。搜就完了，搜过的就不用搜了。

1100.抓住那头牛。如algorithm.md所述，bfs类似于DP，每个结点的值有可能都别的状态转移过来。

多源bfs

173.矩阵距离。多源bfs，全部加入队列跑bfs

最小步数模型。

1107.魔板。就是把所有可能的状态转移都遍历一遍，由于需要字典序最小，所有操作的顺序要正确。然后还考察了unordered_map的使用。

双端队列广搜

175.电路维修。这道题要求解的问题是如何使电子元器的旋转次数最少，从而形成通路，这里抽象一下问题，是一个从起点到终点的最短路问题，能连通的电线权重为0，不能连通的电线权重为1（表示需要旋转），然后起点到终点的最短路，就是问题要求解的元件旋转次数。由于权重只有01，所以可以利用**双端队列bfs**来求解最短路问题。双端队列bfs具体见algorithm.md笔记。这道题的难点就是转化成双端队列bfs来做。

双向广搜

190.字串变换。双端队列bfs。就是同时开两个队列，独立的搜，直到会和，和一般队列写法类似。

A*

178.第K短路。具体见algorithm.md

179.八数码。单纯bfs也能过。没用A*算法都过了。A*算法的估计函数为每一个数字移动到目标位置的曼哈顿距离(最短需要移动的距离)。

DFS之连通性模型

1112.迷宫。之前遇到的dfs搜索，回溯的时候都要恢复现场。这道题有点不同，只需要找到是否有路径即可，所以不需要重复访问结点。只需要访问所有的点就可以，不需要访问所有的路径。

1113.红与黑。dfs遍历所有可以到达的结点。

DFS之搜索顺序

1116.马走日。逆天，方向数组少打了个逗号，找了半天。

1117.单词接龙。搜索所有可能性，记得恢复现场

1118.分成互质组。遍历所有物品可能放置的篮子。

DFS之剪枝与优化

165.小猫爬山。

166.数独。

DP问题

状态机模型

1049.大盗阿福。f[i][j] j=0/1，表示前i家店铺，偷或不偷能获得的最多的现金数。

1057.股票买卖IV。

1058.股票买卖V。

1052.设计密码。不会不会

状态压缩DP

1064.小国王。 做这种状态压缩的题，都需要先预处理出合法方案以及相邻的合法方案。然后就是状态转移。可以滚动数组降低空间复杂度。

327.玉米田。和1064类似，都要先预处理出合法方案

292.炮兵阵地。

树形DP

1073.树的中心。换根DP，第一次dfs指定一个根节点求解每个结点向下到叶子节点的最大值和次大值。题目要求求解的是指定一个点，到所有点的距离中的最大值中的最小值。这里的"换根"思想，其实本质就是DP，我们需要求解从节点u向上移动到达叶子节点的最大距离，而求解这个距离需要找到当前节点和父节点的关系。如何求解u节点向上的最大距离，fu的最大距离不经过u，那么向上的最大距离就是d1[fu]+w，如果fu的最大距离经过u，那么只能利用fu的次大值，d2[fu]+w。

1072.树的最长路径。求每个内部节点到叶节点的最长路和次长路。

1075.数字转换。把数字之间的变换转化为图上找最长路径。

1074.二叉苹果树。树上背包问题。f[i][j]定义为以i为子树的，选了j个树枝的最大苹果树

323.战略游戏。一开始以为也是树上背包问题，搞错了。原来是状态机DP，对f[i][j]表示以i为根节点的子树，j=0/1表示选或不选的最小士兵数。

1077.皇宫看守。状态定义如下

```
/*
    f[i][0]=sum(min(f[son][1],f[son][2]));
    f[i][1]=至少有一个子节点放了守卫-f[son][2] ,其他的子节点要么放，要么被自己的子节点监督
    f[i][2]=sum(min(f[son][0],f[son][1],f[son][2]))
*/
```

数位DP

1081.度的数量。

比赛

Acwing130周赛

5294.最小数和最大数

5295.三元组　　过了一部分的测试，时间复杂度为O（2n*n）。用了y总的方法过了

5296.边的定向　　完全不会

Acwing131周赛

5363.中间数　签到题

5364.奶牛报数　　也是用到前缀和，分三种情况讨论，因为要第一只奶牛报数最小，所以从后面往前列举所有情况。可以用**破环成链**的方式，将环变成线性数组，需要开两倍的数组，计算前缀和。

5365.制作地图　没时间做了，需要利用动态规划

Acwing132周赛

5367.不合群数。就是在判断是否为质数的时候，不需要将a全部枚举，最多只用枚举1e9开根号个数，因为b的最大值是1e9，判断是否为质数只需要判断到最大值开根号的位置。判断质数没有什么简便的方法，只能通过特判减少一些数据量。

5368.最短距离。也不会做，需要用到并查集，图巴拉巴拉的，还没学到，迟点在再来看

Acwing 133周赛

5370.最小和。是一个贪心问题，运用了**排序不等式**，升序和降序的序列，对应位相乘后求和能得到最小值，同为升序或者同为降序能得到和的最大值。又复习了一下结构体排序

5371.素数整除。这道题用的是埃及筛+前缀和，因为埃及筛的思想是筛掉所有质数的倍数，刚好和题目的要求相似

Acwing 144周赛

5474.平均成绩。贪心，一开始想复杂了，还用了dfs巴拉巴拉的，谁知道从小到大排序后贪心就可以了。

5475.聚会。对每个点进行一次bfs，过了5/10的数据。一定要记得给邻接表的表头初始化！！！，找半天错误。学习了多源bfs，一开始是对每一个点到所有点进行bfs然后统计得到s种干草需要的花费，但是这样子的时间复杂度是O(n^2)，会超时。可以发现这道题干草的种类比较少，可以把方向调换一下，对每个干草做bfs，得到每种干草到所有点的最小花费，这样子的时间复杂度为O(k*n)，真的很巧妙！！。

第14届蓝桥杯C++B组题

飞机降落：学到了做提前先看看数据范围，这题N<=10，大概率是爆搜dfs或者是状压DP。

接龙序列：线性DP。

岛屿个数：bfs。首先用bfs找岛屿，然后判断岛屿是否是内部岛，只需要对海水进行八个方向的bfs如果海水能够流到边界，就说明不是一个环。

子串简写。查找的时候如果序列有序，记得用二分！！！！，感觉考的还挺多

整数删除：双链表+优先队列。双链表存相邻节点，优先队列获得序列中的最小值。双链表写的有点生疏，要复习一下。

景区导游：lca倍增优化板子题，唉做的太慢了 不然感觉给我时间都能做出来。

砍树：这道题又是lca+树上差分。对于每一个数对，在树上都会唯一经过条边，每经过一次，我们就把边的权重加一，如果有一条边的权重为m，说明所有的数对都经过它，只要删掉这条边就可以满足条件。这里用到了**边差分**，为了便于计算，我们将边的权重**下移**到点中，转化为点差分。树上差分具体见笔记。

第十四届python B组题

管道：二分枚举，利用二分在logn的时间复杂度内枚举所有答案（前提是答案是有二分性的），然后就是区间覆盖问题

保险箱：又是DP问题，我发现类似这种最少或最多这种问题，然后情况非常多的，都是用DP来做。

第十四届java B组题

矩形总面积：

合并石子：这道题是在区间合并的基础上，加上了限制条件，就是只有同一种颜色的石子才可以合并。构造f[i] [j] [k] 集合表示合并i~j区间的棋子，并且合并后的颜色是k的最小花费。因为开始需要把f数组，w数组（i ~ j 区间的最小花销）都初始化为正无穷，如果两个区间无法合并的话，w的值会一直是正无穷，所以最后还需要遍历一下，把无法合并的区间的w的值进行更新。

第十三届C++B组题

统计子矩阵：枚举+前缀和。需要注意枚举的方法，如果直接枚举矩阵上下端点n^4的时间复杂度会超时。根据观察，前缀和的性质是单调不减的，所以我们可以先枚举行，然后列中利用单调不减的性质判断符合条件的矩阵区间。

李白打酒加强版：DP。f[i] [j] [k]表示走到第i个位置，遇到j个花，还剩k斗酒的可能性种数，然后对该集合进行分类讨论为在i位置遇到话花或遇到店，分别判断可以从前面哪个状态转移过来

第十二届国赛pythonB组题

点亮：直接dfs搜索

近似gcd：

蓝桥杯国赛模拟卷（上）

抓娃娃：通过题意可知，所有的区间都比线段长，所以只要线段的中点在区间内，区间就能框住线段。然后维护一下前缀和。这里有个技巧，就是在求线段中点的时候，会出现浮点数，可以通过把求解区间的边界都乘以2，虽然区间大小变了，但是结果是不会变的。

01游戏：就是搜索，然后需要剪枝。一开始在搜索过程中没有剪枝，而是直接判断最终结果是否满足条件，还以为是代码写错了一直不出结果，原来是时间复杂度太高了没跑完醉了。。搜索过程中剪枝就是提前判断一下填这个数后，是否满足条件，如果不满足，那就可以直接剪枝了。可以提前判断是否有连续3个以上的数或者填的数是否已经过半（直接中途剪枝）

不完整的算式：用字符串的find函数，找到=和？的位置，然后if-else判断就行了，不知道为什么当时想不出来，感觉把题目复杂化了。

跑步计划：日期题，easy

数和游戏：

板钉上的正方形：枚举的技巧，固定一个点，然后枚举左下方的点，然后计算出其他两个点的坐标。

蓝桥杯国赛模拟卷（下）

最大区间：对于暴力做法，枚举区间，使得结果最大。这里可以转换一下思路，通过枚举最小值（也就是序列），然后找到满足条件的最大区间。满足条件的最大区间怎么找呢？为了满足这个区间中的最小值随着区间的扩大不变化，我们需要找到前一个第一个比它小的值和后一个第一个比他大的值。

游戏：这道题用的是单调队列，一开始就是对的。但是对于期望的求解公式，没有化简一下，然后还用了二层循环，真是醉了。求出每个区间的最大最小值之后，然后就是最大最小值做差求和，可以发现每个最大值和最小值都被用了len（区间的数量）次，然后期望求解的分母是len*len，这样刚好可以约掉一个cnt，这样期望的求解公式就是sum(max-min)/len。O(n)的时间复杂度就可以搞定了。

翻转：我去，这题想麻烦了，原来是单个小工件翻转，我还以为是拼接后的工件也可以翻转。f[i] [0/1]表示第i个工件翻转或不翻转的最短工件长度。

火车问题：知道是背包问题，一共有两个背包，每个背包枚举一下重量限制。一开始写错的原因如下：就是for循环的边界错了，写了>=w，在一个背包的时候可以写>=w，因为<w是不执行的。但是两个背包就不一样了当其中一个背包<w的时候，另一个背包可以>=w。然后就写错了，思维定式了焯！本来能写对的。

```
for(int j=m1;j>=0;j--){
    for(int k=m2;k>=0;k--){

        if(j>=w)f[j][k]=max(f[j][k],f[j-w][k]+w);
        if(k>=w)f[j][k]=max(f[j][k],f[j][k-w]+w);
    }
}
```

非对称二叉树：这道题好特别，第一次遇到（也是思维定式了，一时间蒙了）。就是求解给定结点数和树的深度，在满足条件下，求解树的方案数。DP求解，f[i] [j]为i个节点，高度为j的非对称二叉树的可能性，然后枚举左右子树的所有情况（要满足节点个数和树的深度的约束），做了这么多题，怎么感觉DP就是看似聪明一点的暴力？！

最大阶梯：动态规划，一开始看官方题解，还以为特别难。有四种阶梯的方向，所以要做4遍DP去最大值。以f[i] [j]为右下角端点为例，可以从左侧方块和上侧方块转移过来，f[i] [j]=min(f[i] [j-1]，f[i-1] [j])+1。初始化f数组都为1，表示阶梯为1。

蓝桥杯14届B组c++国赛题

子2023：线性DP，唉居然没想到，这么简单的用DP数组维护一下以2,20,202,2023结尾的情况数就可以了。知道了一个新函数**to_string()**，可以将数字转换为字符串

班级活动：这道题也想快了，如果相同编号的数量超过2个，那超过2的部分就是全部都要更改。

数三角：考察了STL的使用。枚举一个点作为顶点，把所有与顶点距离相同的点都加入到一个集合中，用map和vector来存。还要判断斜率不存在的情况。

删边问题：？

AB路线：bfs，每个点有多种不同的状态到达，所以st数组需要多开一维，不同状态就是到达这个点已经经过了多少个与之相同的字符，因为题目限定是K个

拼数字：？

逃跑：DP问题。根据题目求解设置f数组，f[i]为i结点到根节点的最短时间的期望。假设j为子结点，u为父节点，那么子结点到根结点的期望等于子结点到父节点的期望+父节点到根结点的期望（**期望的可加性**）。如果父节点是跳板，那么不管成功或是失败,到父节点的时间都是1，f[j]=f[u]+1，如果父节点不是跳板，那么想要到达父节点，子结点踩的踏板一定都是失败的，那么到父节点的时间期望就是1*pow(p,t)。

蓝桥杯14届B组python国赛题：

背包问题：思维定式了一开始，看到拿取问题，就以为是背包DP，但是但从这道题的数据范围来看，不可能是背包DP。这道题通过枚举前两个背包的体积小的物品的所有放置情况，然后算出所有可能性取最大值即可。

2023睿抗本科组省赛：

RC-u5相对成功与失败：线性DP，寻找最长不增子序列。这里设置f数组特别巧妙。对于一般的最长子序列DP，f数组定义为以第i个数为结尾的最长子序列的长度，时间复杂度是O(n^2)，会超时。这道题的分数只有0，1，2，所以可以将f数组定义为以某个分数结尾的的最长子序列长度，可以达到O(n)的时间复杂度

2023睿抗本科组国赛：

RC-u3兰州拉面派餐系统：考察优先队列的使用。

RC-u4拆积木：考察拓扑排序，把方块之间的拿取得约束关系看左边。