1.气球的颜色代表着题目的易：

（1）亮蓝、金、黄色、花

（2）红

（3）绿

2.搞竞赛的同学都知道，一般酱油队，就是打星号的。不过我感觉是在骗我。

3.动态规划就是使用什么（一般是多维数组）来记录中间的一个个子过程（子问题），那么到数组记录的最后就是我们要求的值。

4.传闻B+树的深度比B树更加短？

数据存储的时候推荐使用B-、B+ Tree或者Hash Table。B- 与Hash都不适合进行范围查找返回，所以数据库中使用的就是B+树。Hash Table单个查找倒是很快，但是如果数据量大了，一旦将自己分配的空间用完，那就要重新进行构建。所以对于数据量较大，会持续增长的情况，推荐使用B-、B+。

5.外存排序的原理和Merge Sorting差不多，是将每一哥部分都先排序排好，然后进行整理合并。Merge Sorting的本质就是dp版本的插入排序。所以对于外存排序，我觉得可以这么实现，记录每一个文件中的已读位置（正是记录文件id与各自索引位置），每读取一个字节就加上一个字节。

6. 一个数字的因数，如果使用遍历的话还是使用 数字/2 还是 开平方，具体要看讲究什么喽。

7.排班表算法： 排课问题在70年代就证明是一个NP完全问题，即算法的计算时间是呈指数增长的

（1）基于回溯，按照需要的事时间段来安排人员（但是可以真正解决问题）

（2）基于优先级：第一优先级是某一个需要的时间段中可以选择的人员很少，应该先安排这样的时间段

第二优先级 早上最早的一班以及晚上最晚的一班，能够找到连贯的人的可能性比其他时间段小，应优先考虑

第三优先级是可以选择的人中人员的空闲时间总体程度也是不同的，应该先从总体空闲时间少的安排

良好的情况下一般这种解法都能够得到结果。但是我的数据库课程设计项目因为数据抽象的原因，不太能支持这种算法。

可以选择稍微损失下参与排班人员的公平性。

8.查找字符串：在一堆字符串里找一个字符串

（1）弄成字符串数组，通过遍历

（2）弄成数组其实不方便，因为字符串你是复制粘贴过来的，需要不停加上引号与逗号，没准可以直接括起来弄成一个大字符串然后找子串。这种方法就需要注意会不会你要寻找的字符串只是其他字符串的一部分。或者可以先试一下按照某一个间隔符划分成为数组。

9.使用缓冲区来存储内容的时候，回退会是一件很简单的事情，系统的输入缓冲区、文件的读取缓冲区（文件指针位置）等等，实在不行就实现自己的缓冲区。除了回退，还可以设立指定位置的还原点。

10. 对一个文本文件每一行采用\_.split(" ")，需要考虑空行

11.



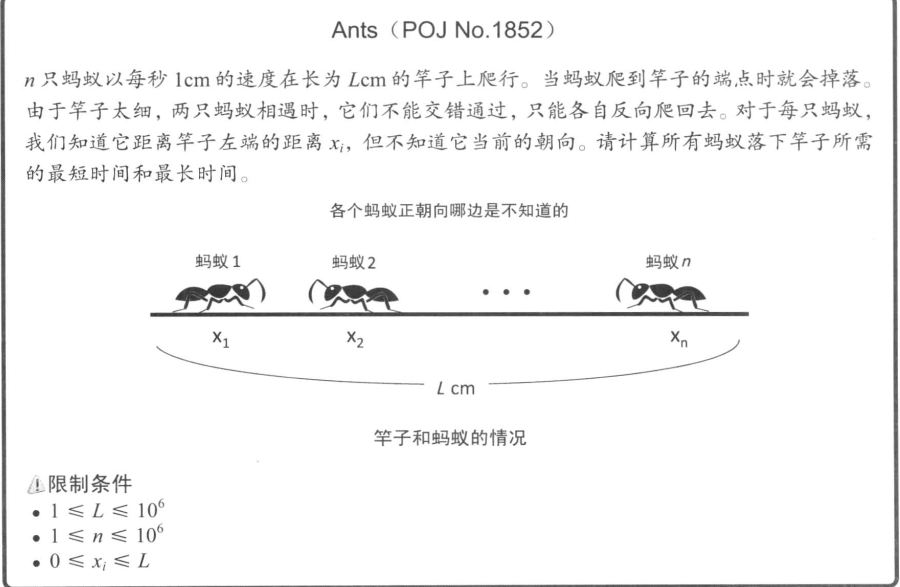
外部归并排序算法： 包含通过胜者树、败者树去优化K路归并

<http://www.360doc.com/content/15/0719/22/22633806_486084820.shtml>

<http://blog.chinaunix.net/uid-26602509-id-3331691.html>

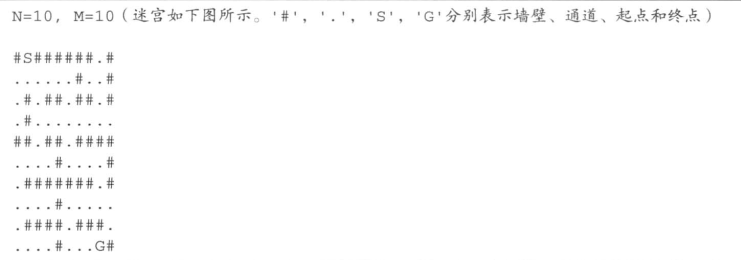
<https://www.cnblogs.com/charlesblc/p/6138908.html>

1. 两个字符串之间的最长子串问题（本子上也有记录） <https://www.cnblogs.com/ider/p/longest-common-substring-problem-optimization.html>



这道题目的关键思路是如何看待蚂蚁的运动，他们相撞然后各自调头返回？是不是会有一种感觉，有一只蚂蚁会不停地撞来撞去，然后这个问题就没有办法解决掉。为何不认真点，完整点做一次模拟，模拟不了大的n，那就模拟小一些的，然后就会发现蚂蚁的相撞可以认为是互相穿过。拓展一下，所以会造成速度交换的，在空间上面都可以认为是穿过。

1. 起点到终点的最短路径，广度优先遍历、当然通过栈实现的深度有限遍历也行



深度可能会先找到这条符合要求的路径，但也有可能会是广度先找到

所以具体选择哪种方法看你喜欢使用队列还是喜欢使用栈

1. 去商店购物，选择合理的优惠方案

输入商店商品与价格二元组对

输入要去商店的购物产品

输入商品编号的组合方案以及优惠价格

根据以上信息去输出得到如何使用优惠能够让自己最受利

我想到用了贪心，计算每一个优惠方案中单个商品的优惠力度，然后排序，从高到低选取

但是贪心果然是不一定完美的，我举个例子：

商品1,2,3的分别要购买5,3,1个。

优惠方案

5个1和2个2能够优惠5元

1个1和1个3能够优惠2元

按照我的判断我会选择优惠2元，因为相当于每一个商品优惠1元，但是优惠5元才是最大的优惠。所以这种贪心只能用在商品购买量足够的前提下。

1. 背包不是一种解法，而是指一类问题，这类问题可以通过贪心、动态规划来解决。
2. sqrt算法的实现：

介绍网站： <https://blog.csdn.net/qq_26499321/article/details/73724763>

下面速度只是反应了一个大概的级别

（1）二分法猜测结果（比系统方法慢了100倍左右）

（2）牛顿迭代法（比系统方法慢了10倍）

（3）系统提供的库函数（编译器的函数都经过了严格的优化）

（4）曾经游戏商3D引擎提供的一个算法（比系统库函数还要快）

float InvSqrt(float x)

{

float xhalf = 0.5f\*x;

int i = \*(int\*)&x; // get bits for floating VALUE

i = 0x5f375a86- (i>>1); // gives initial guess y0

x = \*(float\*)&i; // convert bits BACK to float

x = x\*(1.5f-xhalf\*x\*x); // Newton step, repeating increases accuracy

return x;

}

（4）这个方法本质上还是，关键在于 0x5f3759d这个魔法数字

18.DAG 有向无环图

最短、最长、路径计数都采用动态规划

1. 求多维数组连通块的问题，又称为种子填充（floodfill）。
2. 若把变量之间的关系看成一条有向边，那么多条关系表达式就能组成一个有向图，即拓扑排序。
3. 对工作人员成绩进行评定：<18为A 18<=且<27为B 27<=且<36为C 36<=为D。数据等差，差为9，所以设置原点为9（18-9），然后除以9使用switch。总的来说就是用一种等差划分，让数据范围边界除以划分点。