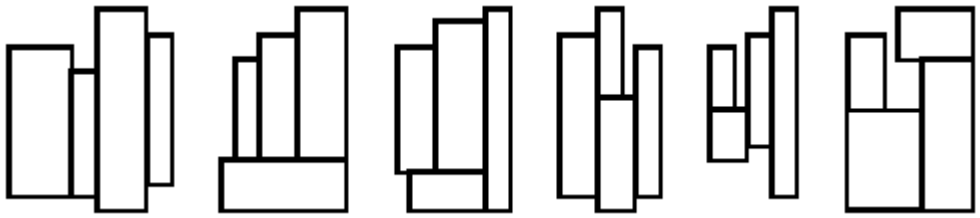
本章主要考深广优先搜索

译题 - packrec

Packing Rectangles 铺放矩形块 (IOI 95)

描述

给定 4 个矩形块，找出一个最小的封闭矩形将这 4 个矩形块放入，但不得相互重叠。所谓最小矩形指该矩形面积最小。



可能存在满足条件且有着同样面积的各种不同的封闭矩形，你应该输出所有这些封闭矩形的边长。

格式

PROGRAM NAME: packrec

INPUT FORMAT:

(file packrec.in)

共有 4 行。每一行用两个正整数来表示一个给定的矩形块的两个边长。矩形块的每条边的边长范围最小是 1，最大是 50。

OUTPUT FORMAT:

(file packrec.out)

总行数为解的总数加 1。第一行是一个整数，代表封闭矩形的最小面积（子任务 A）。接下来的每一行都表示一个解，由数 P 和数 Q 来表示，并且 P≤Q（子任务 B）。这些行必须根据 P 的大小按升序排列，P 小的行在前，大的在后。且所有行都应是不同的。

SAMPLE INPUT

1 2
2 3
3 4
4 5

SAMPLE OUTPUT

40
4 10
5 8

题解 - packrec

分析

只要将示例中的 6 种(其实是 5 种,图 4 和 5 是同一种情况)进行模拟,以求得最优解

这道题的程序实在是……100 多行(主程序极为变态),思路比较好想,可以不使用递归,主要是枚举各种状态,这道题就算是 BFS 吧.

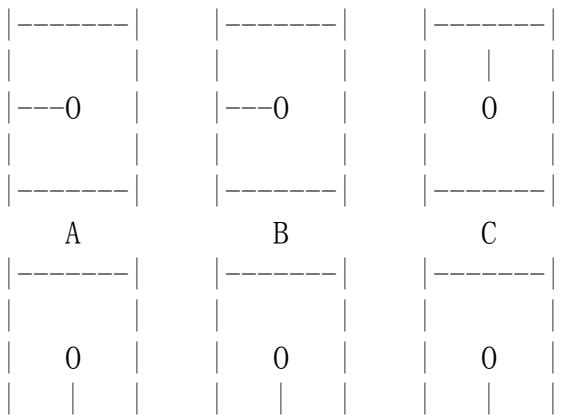
难点不在思想,而是编程能力和条理性.(请看参考程序)

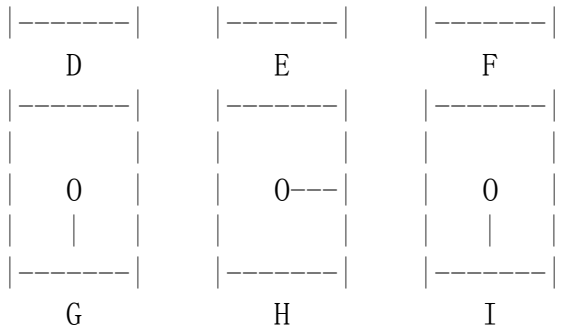
译题 - clocks

The Clocks 时钟 (IOI'94 - Day 2)

描述

考虑将如此安排在一个 3 x3 行列中的九个时钟:





目标要找一个最小的移动顺序次将所有的指针指向 12 点。下面原表格列出了 9 种不同的旋转指针的方法，每一种方法都叫一次移动。选择 1 到 9 号移动方法，将会使在表格中对应的时钟的指针顺时针旋转 90 度。

移动方法	受影响的时钟
1	ABDE
2	ABC
3	BCEF
4	ADG
5	BDEFH
6	CFI
7	DEGH
8	GHI
9	EFHI

Example

```
9 9 12          9 12 12         9 12 12          12 12 12        12 12 12
6 6 6   5 ->    9  9  9   8->   9  9  9   4 ->   12  9  9   9-> 12 12 12
6 3 6           6  6  6         9  9  9          12  9  9        12 12
12
```

[但这可能不是正确的方法，请看下面]

格式

PROGRAM NAME: clocks

INPUT FORMAT:

(file clocks.in)

第 1-3 行: 三个空格分开的数字，每个数字表示一个时钟的初始时间，3,6,9,12。数字的含意和上面第一个例子一样。

OUTPUT FORMAT:

(file clocks.out)

单独的一行包括一个用空格分开的将所有指针指向 12:00 的最短移动顺序的列表。

如果有多种方案，输出那种使的连接起来数字最小的方案。(举例来说 5 2 4 6 < 9 3 1 1)。

SAMPLE INPUT

```
9 9 12
6 6 6
6 3 6
```

SAMPLE OUTPUT

```
4 5 8 9
```

题解 - clocks

分析

可以用 bfs，dfs，枚举，数学方法解

DFS

显而易见地，方案的顺序并不重要。而每种方案最多只能选择 3 次，如果 4 次相当于没有选择。这样，总的搜索量只有 4^9=262144)，用 DFS 搜索所有状态，选择其中一个最小的输出即可。

可以采用位运算加速``下面 C++的程序使用了位运算``

BFS

多么标准的 BFS 啊！

1.一个移动方法使用超迫次就舍弃 2.从移动方浪开始搜，只需拓展大于等于当前结点移动方法的移动方法。这样就避免了搜索 1__ 又搜 2__。而且顺序都排好了。

数学方法

假设时钟 abcdefghi 中除了 a 都到了某一状态，现在有一系列转换可以使 a 转 90 度而其它钟不变，进行这一系列的转换，虽然行动次数多了，但钟没有变，已知没有可以直接转 a 的转换方法，所以其它的钟都是转动 360*n 度的(0<=n 且 n 为整数就是自己独自转了 4*n 周)0<=n 且 n 为整数，既然是 4 的倍数，那么就可以 mod 4(and 3 来的更快亿来抵消掉这些操作而钟的位置没变。这样就得出一个结论了，用枚举或 bfs 得出的最优解，它的解的前一个步骤为之一个状怿用刚才的理论，对于每个没到 12 点的钟作只让它转的一次的一系列操作，这样达到了全为 12 的一种状态，然后将每种操作 mod 4，用枚举或 bfs 得出的最优解的前一个步骤时所做的步骤+最优解的最后一个步骤，这样继续向前推，就可以得到从初始状态开始，直接执行对于每个没到 12 点的钟作只让它转的一次的一系列操作，然后最后结果每种操作 mod 4，就是枚举或 bfs 得出皿最优解。

枚举

这题其实写个 9 重循环枚举就行，然而更简洁的办法是利用递归的函数生成枚举序列，有时可能超时。

译题 - ariprog

Arithmetic Progressions 等差数列

描述

一个等差数列是一个能表示成 a, a+b, a+2b,..., a+nb (n=0,1,2,3,...)的数列。

在这个问题中 a 是一个非负的整数，b 是正整数。写一个程序来找出在双平方数集合 S 中长度为 n 的等差数列。双平方数集合是所有能表示成 p^2＋q^2 的数的集合。

格式

TIME LIMIT: 5 秒

PROGRAM NAME: ariprog

INPUT FORMAT:

(file ariprog.in)

第一行: N(3<= N<=25),要找的等差数列的长度。

第二行: M(1<= M<=250),搜索双平方数的上界 0 <= p,q <= M。

OUTPUT FORMAT:

(file ariprog.out)

如果没有找到数列,输出`NONE'。

如果找到了，输出一行或多行, 每行由二个整数组成:a,b。

这些行应该先按 b 排序再按 a 排序。

所求的等差数列将不会多于 10,000 个。

SAMPLE INPUT

```
5
7
```

SAMPLE OUTPUT

```
1 4
37 4
2 8
29 8
1 12
5 12
13 12
17 12
5 20
2 24
```

题解 - ariprog

分析

这道题就是暴力搜索，时限是 5s，方法是很简单的：枚举所有的可能解，没有剪枝的。

但是在编程细节上要注意，很多时候你的程序复杂度没有问题，但常数过大就决定了你的超时（比如说，你比别人多赋值一次，这在小数据时根本没有区别，但对于 1 个运行 5s 的大数据，你可能就要用 10s 甚至更多）。

具体来说，预处理把所有的 bisquare 算出来，用 bene[i]记录 i 是否是 bisquare，另外为了加速，用 list 记录所有的 bisquare（除去中间的空位置，这在对付大数据时很有用），list 中的数据要有序。 然后枚举 list 中的数，把较小的作为起点，两数的差作为公差，接下来就是用 bene 判断是否存在 n 个等差数，存在的话就存入 path 中，最后排序输出。 运行时间：case 8 超时 （case 7 4.xs） 费时最多的地方是枚举 list 中的数，所以对这个地方的代码加一些小修改，情况就会不一样：

1. 在判断是否存在 n 个等差数时，从末尾向前判断（这个不是主要的）。
2. 在枚举 list 中的数时，假设为 i,j，那么如果 list[i]+(list[j]-list[i])×(n-1)>lim(lim 是最大可能的 bisquare)，那么对于之后的 j 肯定也是大于 lim 的，所以直接 break 掉。（这个非常有效）

AC，最大数据 1.4xs。 其实输出时可以不用排序，用一个指针 b[i]存公差为 i 的 a 的链表，由于搜索时 a 是有序的，存到 b[i]中也应是有序的，这样就可以直接输出。对极限数据 b 的范围应该不超过 m^2/n=2500,即 b:array[1..2500]of point; 而如果 qsort 的话，复杂度为 n*logn,n<=10,000

同时应注意的是,程序中使用整齐划一的编程风格,比使用函数快得多.

译题 - milk3

Mother's Milk 母亲的牛奶

描述

农民约翰有三个容量分别是 A,B,C 升的桶，A,B,C 分别是三个从 1 到 20 的整数，最初，A 和 B 桶都是空的，而 C 桶是装满牛奶的。有时，约翰把牛奶从一个桶倒到另一个桶中，直到被灌桶装满或原桶空了。当然每一次灌注都是完全的。由于节约，牛奶不会有丢失。

写一个程序去帮助约翰找出当 A 桶是空的时候，C 桶中牛奶所剩量的所有可能性。

格式

PROGRAM NAME: milk3

INPUT FORMAT:

(file milk3.in)

单独的一行包括三个整数 A,B 和 C。

OUTPUT FORMAT:

(file milk3.out)

只有一行，升序地列出当 A 桶是空的时候，C 桶牛奶所剩量的所有可能性。

SAMPLE INPUT 1

8 9 10

SAMPLE OUTPUT 1

1 2 8 9 10

SAMPLE INPUT 2

2 5 10

SAMPLE OUTPUT 2

5 6 7 8 9 10

题解 - milk3

分析

Way 1:DFS 因为牛奶的总量是不变的，所以可以用 a,b 中的牛奶量做状态，初始状态是(0,0)，每次只能有 6 种选择，a 倒 b，a 倒 c，b 倒 a，b 倒 c，c 倒 a，c 倒 b。用一个数组 vis[i][j]判重，s[i]记录 c 中所有可能值（s[i]=true 表示 c 中可能出现 i），如果当前状态是(0,x)，那么 s[mc -x]=true，最后输出 s 中所有 true 的就可以了。

Way 2 这个题目首先让我想到的就是递归…不过我不知道临界点是多少,就乱试了,结果用 i=14 能过…不过我不知道为什么 i 要等于 14,没证明过当 i 等于 14 时能包括大多数的过程…

这里用到了一个输出技巧,当你不知道最后一个输出数据是什么而且又不能留空格时,我先把所有数据加空格存到一个 String 里,然后再把 String 里的最后一位去掉就 OK 了…就像这样

```
for i:=0to
20do
    if bo[i]then
    begin
      str(i,strr);
      tmpstr:=tmpstr+strr+' ';
    end;
  delete(tmpstr,length(tmpstr),1);
```

在 procedure 里的思想是这样的

```
for 倒奶的桶→a to c
for 灌奶的桶→a to c
  if 倒奶的桶=灌奶的桶 or 倒奶的桶的牛奶=0 or 灌奶的桶满了 then
    continue
  else
    开始倒奶(有3种可能,其实可以合并成2种,为了小心起见我还是分成3种)
    到下一个此过程
```