



一个钝角三角形至少需要几个锐角三角形组成?   
即:一个钝角三角形如何划分成有限个锐角三角形，使这些锐角三角形个数最少？给出具体划分方法。

划成7个锐角三角形;   
方法:以钝角为一内角,以对边中一段为一边,作一凸五边形;   
取五边形内一点,连五个顶点,成五个锐角三角形,加余下的两个,一共是7个. 毕.   
注:   
分两步走:   
1)划成一个凸五边形和两个锐角三角形;   
2)把五边形划成五个锐角三角形;

有二十四张扑克牌,八张片朝上,有没有可能在眼睛蒙上的情况下,把这些牌分成两堆,使两堆朝上片数相同

第1步：把这24张扑克牌，蒙眼任意分成2堆，一堆8张（第1堆）、一堆16张（第2堆）。第2步：蒙眼把第1堆（8张的那堆），所有牌全部翻一面。16张的那堆牌不用翻。解释：设第1步分完后，第1堆【正】面向上的牌数为 X （0≤X≤8）。则第1堆【背】面向上的牌数为 8 -X ；第2堆【正】面向上的牌数为 8 - X。 现在执行第2步，把第1堆所有牌全部翻一面。则此时：第1堆【正】面向上的牌数为原来第1堆【背】面向上的牌数 8 -X （第1堆【背】面向上的牌数为原来第1堆【正】面向上的牌数 X） 此时，第1堆【正】面向上的牌数 = 第2堆【正】面向上的牌数 = 8 - X

三个射手,A,B,C,射中概率分别为,0.3,0.7,1.00,由A先射击，自由选择设计对象，接着B射，接着C射，如此两轮后，分别求A,B,C的存活率首先要明确的是~A肯定第一枪不会射杀其余二人~~否则下一枪A死的可能性极高~~因此整个行动应该是 A对天开一枪~~B对C开一枪~~（若C未死）则C杀了B~~（若C已死）则A对B开枪~~~ 因此第一个分歧点就是 B有没有杀死C... 先讨论未杀死的情况~概率为0.3~则C杀了B~~然后AC单挑~~ A能够杀死C的概率也是0.3~~否则A死~~因此A的在这里存活率是0.09 再讨论C被杀死的情况~概率为0.7~~~ 进入AB单挑环节~ 则A的杀死B的概率是0.3+ 0.7\*0.3\*0.3 + 0.7\*0.3\*0.7\*0.3\*0.3 …… 约等于0.38 因此A的存活率约等于0.266 两种情况相加得 0.356 这是A的存活率~~ 相应地~~如果B要存活~则必须打中C~否则马上死~~然后要躲过A的射击并且杀死A~~~ 打中C概率0.7 躲过A概率 0.7 击杀A概率0.7 + 0.3\*0.7\*0.7 + 0.3\*0.7\*0.3\*0.7\*0.7 …… 约等于 0.89 则B的存活率等于0.7\*0.7\*0.89 = 0.4362 然后就是C~ C要躲过B 然后再躲过A...就可以胜利了~~ 0.3\*0.7= 0.21 因此~很可怜地~~枪法最好的C存活率最低~~

**又一道笔试题（多益）**

非本人原泄，一下内容纯属从网络搜到的，本意用来互相学习。

有8号牌和9号牌无限张，请问使用这两种牌组合不出来的最大的数是多少

例如：17可以由一张8号牌和9号牌组合而成，但是19就无法由这两种牌组合出来了

感觉用笔都能算出来，但是现在是要用程序实现（既然能用笔算出来了，程序就跟着思路跑了，代码待给出）

首先我们发现一个规律：n个9比n个8大n（感觉有点像是废话），那么我们就想什么时候n个9比n+1个8还大呢，这样你就会使用表达式9n>8（n+1）来算。

其实这里面最大的一个规律是：从8n到9n这段区间的任何一个数都是可以用8和9来组合得来的，并且9n会随着n的增大而与8n相差更大，好，那么只要找到n个9比n+1个8大的临界点，后面的都不用算了，因为后面的n值都会导致n个9比n+1个8大。

总的一句就是，这道题要求解的是一段段的区间什么时候开始相邻或有交集，仅此而已。

说到这里，大家应该都明白怎么做了吧，我就不贴代码了。

**说下多益网络第二轮二笔的题目**

1。二叉排序树的问题

           第一题看到我就晕了，完全没有准备啊，二叉排序树的概念都忘记了，貌似是在树上找一个节点，使得  **（节点左孩子值+右孩子值）/ 2 最小** 不记得是不是这样了。

2。第二题：

**有一串玛瑙项链，项链上面有N个玛瑙珠子，这些玛瑙有M (M < = 9) 种不同的颜色，购买的时候可以只选择购买项链的一段。现在有个苦逼的找不到工作的程序员想花最少的钱买到颜色最多的一段项链，(搜索项链环)，**

**要求时间复杂度O(n),空间复杂度O(1).**

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/lilien1010/article/details/8183963)

1. #include <stdio.h>
2. #include <stdlib.h>
4. #define M 4 //颜色数
6. #define N 11    //项链珠数
8. **int** manao[N] ={1,2,1,2,3,2,4,2,3,4,1};
10. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
11. 判断颜色是否存在
12. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
13. **int** isAll(**int** l,**int** h)
14. {
15. **int** col[M] = {0};
16. **int** i = 0;
17. for( i = l ; i <= h; i++)
18. {
19. col[manao[i]-1] = 1;
20. }
22. for( i = 0 ; i < M; i++)
23. {
24. if( col[i] == 0)
25. return 0;
26. }
27. return 1;
28. }
30. **int** GetMinManao()
31. {
32. **int** l,h;    //{1,2,1,2,3,2,4,2,3,4,1};
33. **int** count = N,ol=0,oh=0;
34. for(h = l = 0 ; h < N &&  l < N ;)
35. {
36. if( isAll(l,h) )
37. {
38. if(count > h - l)
39. {
40. count = h - l +1;
41. ol = l;
42. oh = h;
43. }
44. l++;
45. }
46. else
47. h++;
48. }
50. for(**int** i = ol  ; i <= oh ;i++)
51. {
52. printf("%d ",manao[i]);
53. }
55. printf("  count = %d ",count);
57. return 1;
58. }
60. **int** main()
61. {
62. GetMinManao();
63. }

#include <stdio.h>

#include <stdlib.h>

#define M 4 //颜色数

#define N 11 //项链珠数

int manao[N] ={1,2,1,2,3,2,4,2,3,4,1};

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

判断颜色是否存在

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int isAll(int l,int h)

{

int col[M] = {0};

int i = 0;

for( i = l ; i <= h; i++)

{

col[manao[i]-1] = 1;

}

for( i = 0 ; i < M; i++)

{

if( col[i] == 0)

return 0;

}

return 1;

}

int GetMinManao()

{

int l,h; //{1,2,1,2,3,2,4,2,3,4,1};

int count = N,ol=0,oh=0;

for(h = l = 0 ; h < N && l < N ;)

{

if( isAll(l,h) )

{

if(count > h - l)

{

count = h - l +1;

ol = l;

oh = h;

}

l++;

}

else

h++;

}

for(int i = ol ; i <= oh ;i++)

{

printf("%d ",manao[i]);

}

printf(" count = %d ",count);

return 1;

}

int main()

{

GetMinManao();

}

 3。第三题可就坑爹了，书本后面的题目啊，经典的ACM题目里面的，

**输入一个整型数组，数组里有正数也有负数。  
       数组中连续的一个或多个整数组成一个子数组，每个子数组都有一个和。  
       求所有子数组的和的最大值。要求时间复杂度为O(n)。**

**例如：输入的数组为1, -2, 3, 10, -4, 7, 2, -5，和最大的子数组为3, 10, -4, 7, 2，  
       因此输出为该子数组的和18。**