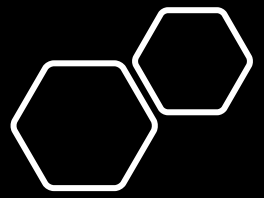
马上开始

35tang-C++竞赛系列四阶课程

三五堂

《35tang-C++竞赛系列四阶课程》



tips

- 找周期
- 找数学规律
- 数论的知识

NOIP2009细胞分裂

- https://www.luogu.com.cn/problem/P1069
- 先看题，理解题目
- 用测试数据验证你是否真的理解了
- 实际上就是n个数打擂，看每一个数不停的累乘，最少多少次可以被m_1的m_2次方整除。

- 对于 50%的数据，有m1^m2 ≤ 30000。
- 对于所有的数据，有1 ≤N≤ 10000,1 ≤m1 ≤ 30000,1 ≤m2 ≤ 10000,1 ≤ Si ≤ 2,000,000,000

观察示例数据

2

24 1

30 12

要求整除24的1次方，也就是24

第一种细胞每次分裂*30，分裂1次后是1*30，2次是30*30，3次是30*30*30，这个时侯可以被24整除。所以第一种需要分裂3次。

第二种细胞每次分裂*12，分裂1次后是1*12，两次是12*12，这个时侯可以被24整除。所以第二种需要分裂2次。

打擂取最小，2次，也就是2s

分析

- 一看就是数学题，如果纯模拟，高精度除法，挨个去尝试一次一次的去整除：比如每一个数字s，过1s就*s，然后去判断是否被m1的m2次方整除，这样肯定可以的一些分数，但是不能满分，会爆掉。两个高精度数的除法不好写。
- 这种题涉及到整除，肯定是数论，质因数等知识。

数论

- 原理1：如果a被b整除，那么b的质因数一定都包含a的质因数中，而且这些相同的质因数：a分解出来的质因数个数一定大于等于b的对应质因数的个数。比如b是20，20=2*2*5；那么如果有一个整数可以被b整除，这个整数分解质因数至少包含2个2和1个5。
- 原理2： 20=2*2*5，20分解质因数包含2个2和1个5，那么20的k次方，一定包含2*k个2和1*1k个5。
- 原理3：如果s分解质因数有k个2，那么分解1次（*自己1次）就有2*k个2，2次分解就有3*k个2…
- 对于每一个s，如果要被m_1的m_2次方整除，首先是分解质因数。举个例子，如果m_1分解质因数有3个2，而s有2个2，那么s至少累乘2个才能得到4个2，才可以整除m_1里面的所有的2，如果m_1的质因数还有5，s就必须分解出5，然后看看s有几个5，分解几次可以得到多于m_1分解质因数里面的5的个数。对m_1的m_2次方，我们不能直接计算（太大了），但是肯定可以看出来，如果m_1分解质因数是3个2，那么m_1的m_2次方就有3*m_2个2。

分析

2

24 1

30 12

m1分解质因数为3个2，1个3．m2是1，所以需要分解出至少3个2，1个3.

第一种细胞每次分裂*30，也就是*一次出来1个2和1个3，可以看出来3次可以出来3个2，3个3，满足条件

第二种细胞每次分裂*12，也就是*一次出来2个2和1个3，可以看出来2次可以出来4个2，2个3，满足条件

取最小，2次，也就是2s

算法：分解质因数的方法不限于下面的方法。

1. 找到所有小于m_1的质因数（m_1很小，所以很快）

2.对m1做质因数分解（用1中得到的质数挨个计算），记录有多少质因数以及每一个质因数的个数。比如分别放在数组primefac和primecount里面。不能整除就是0个。

3. 读入每一个s，对primefac里面的每一个质因数primefac[j],判断 primefac[j]是否整除s如果不整除，s不可选；如果整除，看s可以分解为多少(count)个primefac[j]，需要的时间就是所有质因数计算取最大： ceil(primecount[j]*m2/count)。

4.对于所有的s打擂找到需要时间的最小值

看程序

很简单，你能写的更短更快么？

NOIP2006 Jam的计数法

https://www.luogu.com.cn/problem/P1061

先看题，理解题目

用测试数据验证你是否真的理解了

英文字母按原先的顺序，排在前面的字母小于排在它后面的字母。我们把这样的“数字”称为 Jam 数字。在 Jam 数字中，每个字母互不相同，而且从左到右是严格递增的。

读入的一个 Jam 数字，按顺序输出紧接在后面的 5 个 Jam 数字，如果后面没有那么多 Jam 数字，那么有几个就输出几个。

手工模拟 2 10 表示可用字母从b到j

【输入样例】

2 10 5

bdfij

【输出样例】

bdghi

bdghj

bdgij

bdhij

befgh

给定只能从{b,c,d,e,f,g,h,i,j}里面选择，你是怎么推导出这5个的？如果需要第6个，是什么？

需要搜索么？

- 搜索之前先看枚举可以不？这里只有5个输出，完全可以根据上一个输出直接枚举出下一个可能的输出。
- 给定只能从{b,c,d,e,f,g,h,i,j}里面选择，bdghj的下一个是什么？
- 一种直观思路：从后往前挨个尝试。对于每一个位置，尝试换成下一个字母，然后去尝试后面的位置能不能都填上。比如：
  - j后面没有字母了，换不了
  - h后面可以换成i，那么完后下一个必须是i后面的第一个，也就是j，搞定，不用再尝试了，下一个就是bdgij

优化

- 例如，从2到10，只能使用{b,c,d,e,f,g,h,i,j}这些字母。如果再规定位数为5，那么，紧接在Jam数字“bdfij”之后的数字应该是“bdghi”。按照刚才算法，尝试最后一位不行，尝试到数第二位i换成j也不行，最后发现f换成g可以，于是把后面换成i和j，搞定。
- 给定的可用字符是严格递增的，这就是说只要知道一个字母，就知道这个字母后面还有几个字母可用，比如“bdghi”，如果只换最后一位，后面的一个位置只有一个字母可用，因为字符’i’和最大的字符’j’ 只差1个。再比如“bdfij”，如果把其中的’f’换成下一个字母’g’, ‘g’后面的两个位置有3个字母可用，’h’,’i,’j’。也就是说，对于每一个位置，完全可以根据当前字母是什么，以及当前的位置直接判断出来当前位置是否可以替换成下一个字母，而不需要挨个尝试填充后面的字母。从而把枚举尝试变成了if语句的条件判断，然后直接生成下一个字符串。

分析

由于只输出5个，暴力一个一个输出，每一个输出基于前一个字符串判断下一个字符串，输出后更新为新的最新字符串。

对当前字符串：由于每一个字母顺序排列，后面的比前面的大。所以从后往前挨个尝试替换（也就是看某一个位置的字符后面还有几个，够不够把后面的都填上），如果找到了，后面依次加1。比如例子bdfij，从后往前，找到第4个字符i，由于i后面只有1个可用字符了，而后面还有一个字符位置。所以如果把i换成j，那么后面第5个字符无法替换。所以不做。找到f的时候，由于f到结束字符j有5个字符，而f现在在第三个位置，后面只有2个位置需要字符，这就说明可以把第3个字符换成g，于是从第三个开始3个位置按照顺序填入后续的3个字符g，h，i。

给定的是字符范围（数字），和初始的字符串，干脆全部统一成数字来操作，由于字母自己就是ASCII码。

```
for (int i=1;i<=w;i++) {
        cin>>ch;
        rarr[i]=ch-‘a‘+1;
}
```

不转换直接按照字符操作也是可以的，如果这么操作注意输入的可选字符范围需要+’a’-1，比如输入范围2 10，代表字符’b’到’j’，就要修改2 10为 ‘a’+1, ‘a’+9。

验证算法{b,c,d,e,f,g,h,i,j}

输入样例

2 10 5

bdfij

输出样例

bdghi　找到”bdfij”中的f是可以增的，所以就是f变成g，后面顺序为h和i

bdghj　找到” bdghi”中的i是可以增的，所以就是i变成g

bdgij　找到” bdghj”中的h是可以增的，所以就是h变成i，i后面是j

bdhij　找到” bdgij”中的g是可以增的，所以就是g变成h，后面顺序为i和j

befgh　找到” bdhij”中的d是可以增的，所以就是d变成e，后面顺序为f g h

```
for (int i=1;i<=5;i++) //输出5个
{
        for (int j=w;j>=1;j--) //从后往前找
        {
                if (t-rarr[j]>w-j) //这个字符用后一个替换，后面有足够字符填充串后续位置，t是结束字符对应的整数
                {
                        //后面的位置顺序往后填写新字符，+1填写到下一个位置，再+1再往后一个位置…
                        int cur=rarr[j]+1;//计算当前位置的新字符
                        for (int k=j;k<=w;k++){//填写后续位置
                                rarr[k]=cur;
                                cur++;
                        }
                        break;//第i个搞定，终止j循环
                }
        }
        for (int k=1;k<=w;k++) cout<<char(rarr[k]+‘a’-1);//注意类型强制转换,需要输出字符而不是字符对应的整数
        cout<<endl;
}
```

NOIP2004火星人

https://www.luogu.com.cn/problem/P1088

先看题，理解题目

用测试数据验证你是否真的理解了

其实就是按照全排列的方式找到当前数字序列的后面第m个序列（m<=100），从示例数据开始枚举后面的找找规律

示例输入

5

3

1 2 3 4 5

示例输出

1 2 4 5 3

12345

12354

12435

12453 ---示例输出的答案

12534

12543

13245

13254

13425

13452

13524

相比较前一个数字，红色是发生变化的，红色后面的是剩下的数字递增。注意：哪一个位置变化？变化成哪一个数字？后面的数字怎么形成？比如看12543为什么下一个数字是13245？

找规律

- 怎么找到下一个数字:
- 1 3 5 4 2的下一个：从后往前找第一个下降子序列，这里是5 4 2，在这里面找到比前面一个数字3大的最小数字—4，用4替换3，成为1 4 5 3 2然后4后面的排列倒着排1 4 2 3 5
- 最后一个数字比前面的大，也算下降，比如1 2 3 4 5 ，5是最后一个，也是下降子序列，所以下一个数字就是 1 2 3 5 4

验证程序逻辑

- 设数组是a[]={0,1 3 5 4 2}，数组下标从1开始使用，逻辑就是
  - 1. 从后往前找第一个下降子序列，假设找到的这个下降子序列的第一个数组下标是i（i=3）
  - 2.在>=3之后的数组元素中找到比a[i-1]大的最小数字4，数组下标是j=4
  - 3. 交换a[i-1]和a[j]
  - 4.数组从i开始重新排列，for (int k=i;k<=n;k++) a[k]=b[n+i-k] ，这里需要提前复制一个a到b
  - 5.开始找下一个数字
  - 我们看到这里面的时间复杂度都是O(n)。加上外面的也就是O(n*m), n最大10000，m最大100

还可以怎么做

全排列!

从第一个开始枚举爆掉。如果开始想不到好的方法，直接全排列。找到给定数列开始计数，后面数m个再输出。至少得些分数。

能不能跳过去前面的，从给定数列开始枚举?

先看前面全排列的标准程序

给定231，如果跳过前面的123 ,132,213，从231开始枚举后面m个？

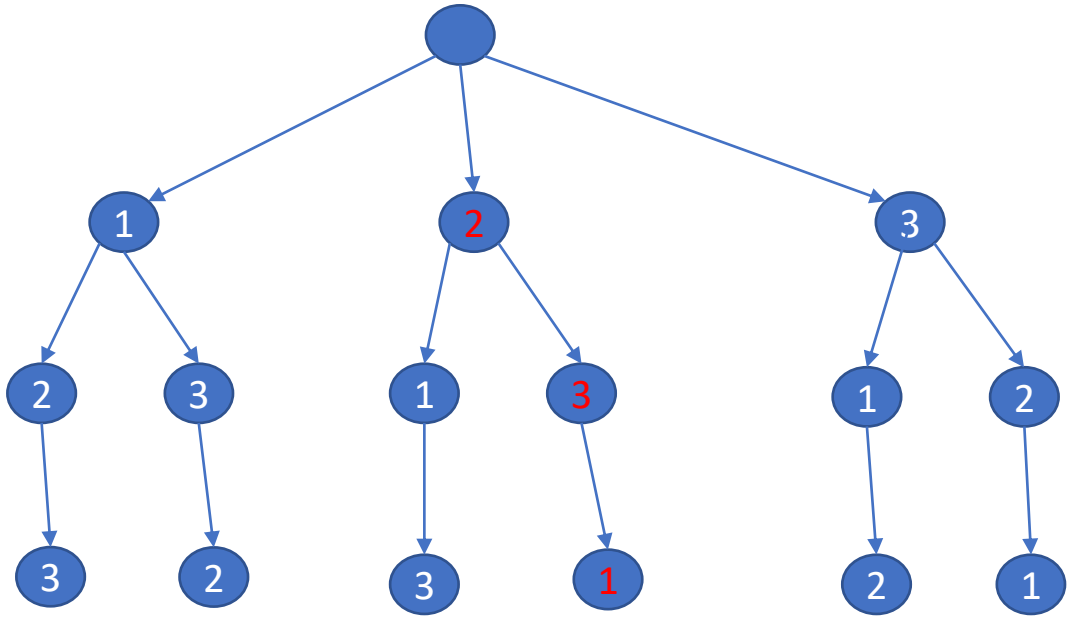
123
132
213
231
312
321

j=0选第一个数

j=1选第2个数

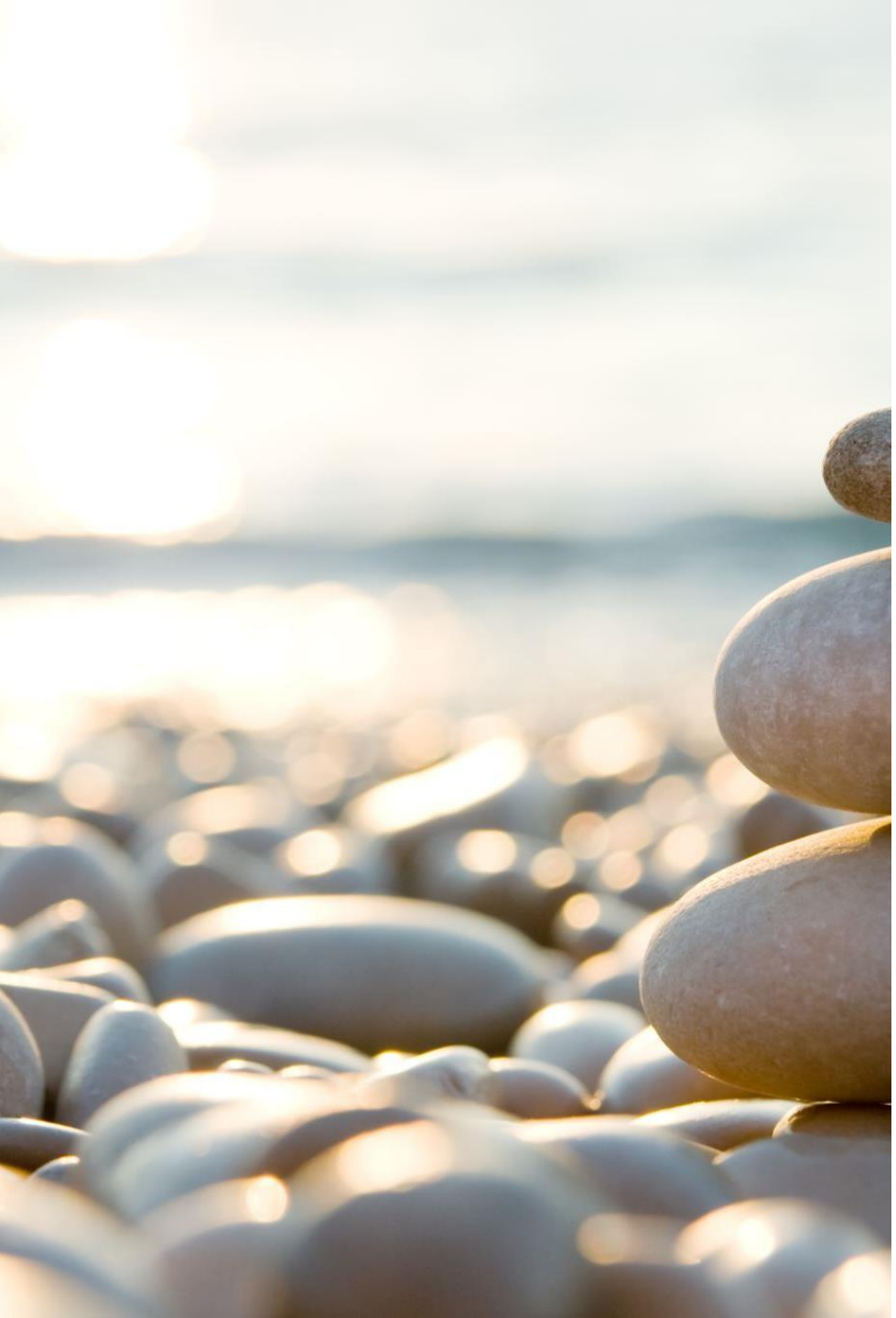
j=2选第3个数

对于每一个位置，按照顺序
枚举前面没有用过的




```
for (int l=1;l<=n;l++) {
    if (!visited[l]){
            visited[l]=true;  outp.push_back(l);
            dfs(j+1);
            outp.pop_back();visited[l]=false;
} }
```

for (int l=1;l<=n;l++) {
这里对于每一个位置j都要去尝试所有没有用过的数字，比如对于位置1，数字1，2，3挨个尝试一次。但是观察，实际上我们不需要数字1的所有分支路线，是否可以直接从2开始尝试？ 下面找位置2类似，直接从数字3开始尝试。这样相当于2 3 1这条路线左面的全剪掉了。



修改思路

- for (int l=1;l<=n;l++)

全排列这里对于每一个位置j+1，用l枚举j位置放1到n中没有用过的，然后走j+1的位置。可以修改一下让l直接等于给定序列里面j位置的数，等于跳过了前面的数字。而不是一个一个去尝试了。一旦n个位置都填好了，开始计数，输出后面第m+1个j到达位置n的完整序列。

一起修改一下

最简单的办法：系统函数

```
#include<bits/stdc++.h>
using namespace std;
int main(){
        int n,m;
        cin>>n>>m;
        int a[10001];
        for(int i=1;i<=n;++i)    cin>>a[i];
        for(int i=1;i<=m;++i)    next_permutation(a+1,a+1+n);
        for(int i=1;i<n;++i)     cout<<a[i]<<‘ ‘;
        cout<<a[n];
}
```

格式化

- cout<<setiosflags(ios::fixed)<<setprecision(2)<<a<<endl;

//小数点后保留两位

下节课预习

- USACO 2018 December Contest, Silver Convention
- USACO 2018 December Contest, Silver Convention II

复习作业1

[CSP-J2020] 优秀的拆分

- 为了练习文件操作，保证头文件等的输入正确，该题请严格按照文件输入输出并且去下面网站提交保证100%正确
- https://www.oitiku.com/rematch/4/41

作业2—下节课会作为例题讲，自己先试一下

找出字符串内最大的回文串。

输入

第一行：一个数字N，表示字符串的长度。字符串由大小写字母组成。

第二行，一段长度为N的字符串

输出

第一行，一个数字M，表示回文串的长度。

第二行，一段长度为M的回文串，如果有多个解，则输出在原串里最靠前的一个。

样例输入

10

abcdcdcbaa

样例输出

9

abcdcdcba

- 数据范围限制

对于20%的数据，N<=100
对于100%的数据，N<=10000

挑战选做作业 3

- NOIP2009细胞分裂
- 课堂例题用的方法代码比较多，而且效率并不是最高的，你有什么好的速度更快代码更短的方法？

由易到难，思维体系训练
实战结合，创新协作培养
兴趣导向，未来职业引领

https://www.35tang.com

https://www.三五堂.com

扫码关注公众号

添加辅导老师