G ArcheyChen的狼题

时间限制：500ms 内存限制：65536kb

通过率：6/28 `(21.43%)` 正确率：6/59 `(10.17%)`

题目描述

ArcheyChen现在事情太忙了，但是他过几天就要交离散作业了，他感觉有点凉凉

他的离散作业里面有一道题目是要写出某个集合的所有非空子集

然而，ArcheyChen手头还有别的作业要肝，于是他只能把这个任务交给你

————————

ArcheyChen的集合比较的有规律。是形如{A,B,C,D....}的形式。

他会给你一个数字nn,n∈[1,20]n∈[1,20], 代表这个集合是{A,B,C...}共有nn个元素。

为了更严谨的描述，两个非空子集的顺序，猪脚头子按照如下方式判别两个集合的先后顺序：

1. 若|S1|≠|S2||S1|≠|S2|，则 集合大小较小的集合先输出。
2. 若将集合|S1||S1| 和 |S2||S2|内的元素按照字母大小排序， 记S1[i],S2[i]S1[i],S2[i]分别表示两个集合中第ii小的元素，恰有一个位置ii满足：

对于∀j∈[1,i)∀j∈[1,i) 有S1[j]=S2[j]S1[j]=S2[j] 且 S1[i]≠S2[i]S1[i]≠S2[i]

此时若S1[i]<S2[i]S1[i]<S2[i] 则先输出S1S1 否则 S2S2

请你按照以上排序方式输出

输入

一个数字，n，含义如上

输出

对每个非空子集输出一行，元素按照给定顺序从小到大输出，每两个元素之间有一个空格隔开

输入样例

```
4
```

输出样例

```
A
B
C
```

```
D
A B
A C
A D
B C
B D
C D
A B C
A B D
A C D
B C D
A B C D
```

Hint

对于一个有限集合S，|S| == card(S) == S的元素个数

——lx

提示

二进制是个好东西

—— 艾克臣

可以用qsort莽，但是如何写比较函数，以及对什么进行排序，是个好问题

putchar的输出速度比printf快

__builtin_popcount(x)可以统计x这个数字里面，二进制中1的个数

——奥萨

hint

对于比较小的情况，可以考虑实质上是对集合按照上述排序，

对于比较大的情况，可以考虑是如何不重复不遗漏的按照上述条件枚举。

如果你get了TLE，不妨尝试putchar —— 子猪头脚。

这个题目有个忧伤的故事。

艾克臣肝计组肝得头晕眼花，突然想起还要出题。于是他赶紧在课堂上用ipad写了个题面。

原本他是打算是出个水题的，结果写样例的时候手滑，写错顺序了。

等艾克臣反应过来的时候，猪脚头子已经帮他出好了数据还有测试程序。

没办法，既然题目已经出了，是道狼题。那就把狼题放出来供大家玩耍吧（笑

首先，我们来看看，不管顺序的情况下。如何遍历所有的子集。

一个有n个元素的集合，他的所有子集，本质上是所有元素“出现/不出现”的所有可能。如果用1代表一个元素出现，0代表这个元素不出现。以4个元素为例，就是_ _ _ _（横线上填上0或者1）的所有可能。

看到这里，我们应该很自然地想到了二进制。

没错，我们只要遍历0000~1111的所有数字，即可找到所有的子集。这一题要求是非空子集，所以我们只需要找0001~1111的所有数字就好了。

我们现在解决了如何找到所有子集，那么我们如何排序呢?

题目要求的排序方法是:

1. **一个集合内部，按照字母顺序排序**
2. **集合元素个数不同的情况下，元素个数小的集合排前面**
3. **元素个数相同的情况下，从左到右，第一个不同的字母较小的排前面**

第一点可以在输出的时候实现。所以，我们对0000~1111这些数字排序就行了。

我们先看第二点，元素个数小的优先。提示里面给了一个函数叫做：**__builtin_popcount();** 我们用他来统计数字里面1的个数，自然就可以得到集合里面的元素个数了。

然后是第三点。如果个数相同，那么就比第一个不同的数字的大小。

我们可以看数字的第一位，如果相同的话，则看第二位，直到发现不同位为止。但是在二进制里面这么看比较麻烦。于是我们可以**用右移操作，加上比较第一位来实现。**

这样一来我们就可以写出比较函数了。

写出比较函数之后，我们就可以用位运算来写一个输出函数。把某个数字转换成集合输出。具体实现看代码吧。

```
#include <stdio.h>
#include <stdlib.h>
//By:ArcheyChen
int n;
int a[1055576];
int ba,bb,ta,tb;
void output(int x)
{
    int i=0;//记录当前是第几位
    while(a[x])
    {
        if(a[x]&1)//如果这一位是1，那么输出对应字母
        {
            putchar('A'+i);
            putchar(' ');
        }
```

```
        i++;
        a[x]>>=1;//用右移来代替查找第i位
    }
    putchar('\n');
}
int cmp(const void *A,const void *B)
{
    ba=__builtin_popcount(ta=*(const int*)A);//ba存储的是a中1的个数，ta存储的是a
    bb=__builtin_popcount(tb=*(const int*)B);

    if(ba!=bb)
        return ba<bb?-1:1;//长度不同的情况下
    while((ta&1)==(tb&1) && ta && tb)//长度相同的情况下，则找第一个不同的位
        ta>>=1,tb>>=1;
    return ta&1?-1:1;

}
int main()
{
    scanf("%d",&n);

    int i;
    for(i=1;i<(1<<n);i++)//把00……001~11……111给写进去
        a[i]=i;
    qsort(a+1,(1<<n)-1,sizeof(int),cmp);//排序
    for(i=1;i<(1<<n);i++)//输出
        output(i);
    return 0;
}
```