P1618 三连击（升级版）

题目描述

将 1,2,…,9共9个数分成三组，分别组成三个三位数，且使这三个三位数的比例是 A:B:C，试求出所有满足条件的三个三位数，若无解，输出 No!!!。

输入格式

三个数，A,B,C

输出格式

若干行，每行3个数字。按照每行第一个数字升序排列。

输入样例

1 2 3

输出样例

192 384 576
219 438 657
273 546 819
327 654 981

解析1

完全暴力枚举，将9个数字分别从1-9进行尝试，最终将达到99次计算，无法在一秒内完成。所以不可使 用。

解析题目，我们能够推导出两个关键信息： 枚举范围： 123到987。

枚举对象：仅需第一个三位数，对于后面两个数字，我们可以根据比例进行计算得出。

例如第一个数字为123，输入的比例为1,2,3，那么后面的两个三位数就应该分别是246,369，又因为这3 个三位数中存在大量的重复数字，如2,3,6，所以该组合不满足题意。

此题目中还包含第二个问题，如何判断1-9这九个数字完全被使用而没有重复？很简单，我们仅需要使 用一个长度为10的数组桶，来对已使用的数字进行标记即可。

比例小知识：A：B = 2:5，已知A=3，求B?

B = 5 * A / 2

综上，我们应该按照以下三步进行判断：

1、判断比例是否满足条件
2、判断位数是否满足条件(三位数)
3、是否使用了全部的1-9的数字

编码

```
#include <bits/stdc++.h>

using namespace std;

//数字桶，用来判断是否使用了1-9,这九个数字
int b[10];

//分别拆解三位数
void markNum(int x) {
    b[x % 10] = 1;  //存个位
    b[x / 10 % 10] = 1; //存十位
    b[x / 100] = 1; //存百位
}

//判断三个数字是否符合要求
bool check(int x, int y, int z) {
    //清空数组
    memset(b, 0, sizeof(b));
    //数字越界了
    if (y > 999 || z > 999)
        return false;
    //标记三位数
    markNum(x), markNum(y), markNum(z);
```

```
    for (int i = 1; i <= 9; i++) {
        //有的数字没有被使用，所以不符合题意。
       if (!b[i]) {
            return false;
        }
    }
    return true;
}

int main(int argc, char **argv) {
    long long A, B, C, x, y, z, cnt = 0;
    cin >> A >> B >> C;
    for (x = 123; x <= 987; x++) {
        //取余等于0则说明成比例
       if (x * B % A || x * C % A) {
            continue; //彼此之间不成比例， 跳入下一个循环
        }

        //求出来的后两个三位数
       y = x * B / A;
        z = x * C / A;
        if (check(x, y, z)) {
            printf("%lld %lld %lld\n", x, y, z);
            cnt++;
        }
    }
    if (cnt == 0) {
        puts("No!!!");
    }
    return 0;
}
```

解析2

使用内置库next_permutation，它的作用是对数组进行全排列。

编码

```
#include <bits/stdc++.h>

using namespace std;
typedef long long LL;
int a[10];

int main(int argc, char **argv) {
    LL A, B, C, x, y, z, cnt = 0;
    cin >> A >> B >> C;
    //将九个数字存入数组
   for (int i = 1; i <= 9; i++) {
```

```
        a[i] = i;
    }
    do {
        //判断是否满足需求
       x = a[1] * 100 + a[2] * 10 + a[3];
        y = a[4] * 100 + a[5] * 10 + a[6];
        z = a[7] * 100 + a[8] * 10 + a[9];
        if (x * B == y * A && y * C == z * B) {
            printf("%lld %lld %lld\n", x, y, z), cnt++;
        }
    }
        //调用内置库获取下一个循环组合
       //注意，该函数同样是左闭右开
    while (next_permutation(a + 1, a + 10));
    if (!cnt) {
        cout << "No!!!";
    }
    return 0;
}
```