最难：应用题（梳理成算法，再用C描述）

首先梳理变量类型和取值（bool可以用无符号短整型代替？）

结果符不符合实际不是需要考虑的问题，描述清楚即可

穷举法——基本思路（计算机强大的计算能力）（后续会学套路）

逗号运算符：逗号

优先级是所有运算符（包括分号？）中最低，从左至右结合，先记住有这个

分号是语句运算符，逗号是把两个表达式连接起来构成一个整体（完整的语句）

int x,y;中的逗号不是逗号运算符

（\*一些时候是指针，一些时候是乘法，语义按照上下文理解）

顺着表达式计算，整个表达式的值是最后一个表达式的值

规则都是人为指定的（只能按照规定执行，不是法无禁止即可为的）

语言是工具，严格按照要求写即可，模糊地带就不能理解

sizeof()是运算符（不是函数）

求表达式结果或某种数据类型的量在内存中所占字节数

可以先测试一下电脑里对应的字长多少

C语言运算符丰富，结合性和优先级很多，不期望背下来

加括号，拆分表达式都可以：编程讲究清晰而非复杂性

语句和表达式差一个分号

有了语句就可以写程序了

C程序最小的组成单元是语句（statement）

程序的语句是按照顺序存储的，按照结构（而非顺序）执行

申明语句：说明变量类型（如int a;）

表达式语句：运算符+……+分号

执行语句：控制、函数、表达式等

空语句：单个分号（占座位）

复合语句：用{}实现多个语句组合（有些变量有作用范围，花括号中的变量只在里面有效）

语句是砖，写程序是盖房子（写代码是最低层次，算法和架构才是更高级的）

结构化→调试方便

早期代码：加goto打乱语句执行顺序，很乱

1968年提出结构化程序设计思想，反对使用goto语句

1. 谨慎使用goto

2. 单入口、单出口（main）

3. 自顶向下，逐步求精细分（函数）（最底下的最小，方便解决问题）

4. 三种基本控制结构：顺序、选择、循环，可以证明这三种结构能够表达一切程序

后续学习控制语句怎么用（控制语句不能用作变量名）

顺序结构：无控制语句，按书写（存储）顺序执行

选择结构：控制语句——if，switch

循环结构：控制语句——for，while，do-while

#pragma warning(disable:4996) 去掉警告4996

srand，rand是系统函数

srand是种子值，rand是产生准随机数（以种子为基础）

若要防止每次rand一样，就让srand加上时间戳（对srand引入随机性）

顺序结构：如何存储就如何执行

复合语句：大括号括起来，作为独立的组成单元，右括号后无分号

{t=x; x=y; y=t;}完成两个变量交换

复合语句可以限定某些变量的作用范围，最小范围是复合语句范围

复合语句中定义的变量只在复合句中有效

若内外侧变量同名，遵循局部优先；如果里面没有这个名的变量，就去外面找

算法是一辈子的事

从代码不会写到算法不会写

scanf普通字符原样输入，一般不在scanf中写普通字符！

数学计算包含<math.h>

选择（分支）结构：

表达式——Y；N

结构内完成则开始进行结构下一个语句

if；switch：如果对则做什么

格式1：

if (条件)

语句1;

else

语句2;

条件！=0：成立，执行1；==0：不成立，执行2

格式2（简化格式）：

if（条件） 语句；

如果是0就执行if下一条语句，跳过if内

格式3：（嵌套格式）

if（条件1） 语句1；

else if（条件2） 语句2；

else if（条件3） 语句3；

……

else 语句n；

（缩进是素养问题）

else向上就近配对

一级一级判断（第三行的else if对应条件2中进行条件3的判断）

表达式太多画的太长，引入N-S图

条件是表达式，一对括号括起来

条件运算符：唯一三目运算符（简化if语句，执行更快）

表达式1？表达式2：表达式3

即以表达式1判断，Y→表达式2；N→表达式3

比关系、算数优先级低，比赋值高；右结合

作业：画流程图！Y是对，N是错，判断用菱形，语句用矩形

写scanf即可，不用scanf\_s

评判代码：可读性好，运行效率高（依需求决定）

计算效率和存储空间的矛盾达成平衡

scanf前加一个printf的提示行