**1.值日生**

【题目1描述】  
小猴放学了 , 但今天有一些同学要留下来打扫卫生 . 老师说 :" 今天轮到学号是 5 的倍数的同  
学打扫卫生啦 !", 但老师发现人不够 , 于是又说 :" 学号是 7 的倍数的同学也留下来吧 !", 但这样人又  
太多了 , 于是老师决定 " 学号是 11 的倍数的同学今天就不用留下啦 ".  
这时候小猴已经晕了 , 你来写一个程序帮他判断一下他需不需要留下吧 .

【输入格式】  
1 行， 1 个正整数 n, 表示小猴的学号。  
【输出格式】  
1 行，如果小猴需要留下则输出 “YES”, 否则输出 “NO”.

#include <iostream>

using namespace std;

int main()

{

int n;

cin >> n;

//请用或(||),且(&&),非(!)运算符与括号补充代码

if (/\*这里补充代码\*/n % 5 == 0/\*这里补充代码\*/n % 7 == 0/\*这里补充代码\*/n % 11 == 0/\*这里补充代码\*/)

cout << "YES" << endl;

else

cout << "NO" << endl;

return 0;

}

**2抢香蕉**

【题目2描述】  
n*n*只小猴展开了 " 蕉王争霸赛 ", 它们的编号分别为1\sim n1∼*n*, 它们会进行m*m*轮比拼 , 每轮有一个胜  
利者 , 胜利者会得到一根香蕉 . 最终 , 得到香蕉最多的选手将获得 " 蕉王 " 称号 . 你自觉 " 蕉 " 力不  
足 , 只好给大家计分 .  
你会得到每一轮比赛的胜利者的编号 , 请告诉列出1\sim n1∼*n*号选手的得分 , 再给出 " 蕉王 " 获得的  
香蕉数 .  
【输入格式】  
第 1 行， 2 个正整数 n,m, 分别表示选手数和比赛轮数(1\le n,m \le 100)(1≤*n*,*m*≤100)。  
第 2 行， m 个用空格隔开的正整数 , 分别表示每一轮获胜者的编号。  
【输出格式】  
输出共 n +1 行 .  
前 n 行依次为 1-n 号选手的得分 ;  
最后一行为 " 蕉王 " 获得的香蕉数 ( 如果有并列 , 只输出一次 ).

#include <iostream>

using namespace std;

int main()

{

int n, m, a[110] = {0};//这里有一个数组a,你打算怎样使用?

cin >> n >> m;

for (int i = 1; i <= m; i++)//对每一场比赛,你打算怎样计分?

{

cin >> /\*这里补充代码\*/;

}

//你需要更多的变量吗?

/\*这里补充代码\*/

for (int i = 1; i <= n; i++)//对每一只小猴,他得了几分呢?

{

cout << /\*这里补充代码\*/ << endl;

//他能成为"蕉王"吗?

/\*这里补充代码\*/

}

//"蕉王"得到了几根香蕉呢?

cout << /\*这里补充代码\*/ << endl;

return 0;

}

# 3多边形数

【题目3描述】  
多边形数是数学上有趣的一类数 . 三⻆形数 , 四边形数 , 五边形数和六边形数是其中 4 种 , 定  
义如下 :  
第 n 个三⻆形数为 :n(n +1)/2, 前几个三⻆形数为 1,3,6,10,15,…  
第 n 个四边形数为 :n^2 , 前几个四边形数为 1,4,9,16,25,…  
第 n 个五边形数为 :n(3n-1)/2, 前几个五边形数为 1,5,12,22,35,…  
第 n 个六边形数为 :n(2n-1), 前几个六边形数为 1,6,15,28,45,…  
有趣的是 , 有些正整数同时属于其中两类 , 例如 6 既是三⻆形数又是六边形数 , 但不是四边形数或五边形数; 9801 既是四边形数又是五边形数 , 但不是三⻆形数或六边形数 .  
请编程找出 100000 以内的所有恰好属于上述四类多边形数中两类的正整数 .  
【输入格式】  
无输入  
【输出格式】  
从小到大输出所有恰好属于上述四类多边形数中两类的数 , 每行一个

#include <iostream>

using namespace std;

int main()

{

int a[100010] = {0};//这里有一个数组a,你打算怎样使用?

//处理所有三角形数

for (int i = 1; /\*这里补充代码\*/ <= 100000; i++){

/\*这里补充代码\*/

}

//处理所有四边形数

for (int i = 1; /\*这里补充代码\*/ <= 100000; i++){

/\*这里补充代码\*/

}

//处理所有五边形数

for (int i = 1; /\*这里补充代码\*/ <= 100000; i++){

/\*这里补充代码\*/

}

//处理所有六边形数

for (int i = 1; /\*这里补充代码\*/ <= 100000; i++){

/\*这里补充代码\*/

}

//哪些数是我们要的呢?

for (int i = 1; i <= 100000; i++)//对每一只小猴,他得了几分呢?{

cout << /\*这里补充代码\*/ << endl;

}

return 0;

}

**4画三⻆形**

【题目4描述】  
我们把满足a^2+b^2=c^2的一组正整数(a,b,c)称作“勾股数”，在⻄方则称作毕达哥拉斯三元组，这样的一组数可以构成直⻆三⻆形的三边⻓，例如(3,4,5)就是一组勾股数。  
小猴有一把边⻓为n的整数刻度尺子 , 因此他只能画出任一条边⻓都不超过n且三边⻓都  
为正整数的三⻆形 .  
那么 , 他能画出的所有直⻆三⻆形中 , 周⻓最⻓的有多⻓ ?

【输入格式】  
1 行， 1 正整数n.  
【输出格式】  
1 行， 1 个整数，为小猴能画出的直角三角形中周长最长的三角形的周长 .  
【输入样例】

100

【输出样例】

240

【样例解释】  
三边长都在 100 以内的整数边长直角三角形中 , 周长最长的周长为 240, 其三边长为 (60,80,100).  
【数据说明】  
对 40% 的数据1\le n\le 201≤*n*≤20;  
对 80% 的数据1\le n\le 3001≤*n*≤300;  
对 100% 的数据1\le n\le 10,0001≤*n*≤10,000;

# 5质数香蕉

【题目5描述】  
小猴太喜欢吃香蕉了 , 就像喜欢数学那么喜欢 , 于是就在每个香蕉上写上了连续的一些正整数 , 并认为每个香蕉的 " 美味 " 程度恰好为这些数中的质数个数 .  
如果你知道一个香蕉上最小的数和最大的数 , 那你就知道这个香蕉的美味程度了 , 例如一  
个香蕉上最大和最小的数分别是 2,10, 那么这根香蕉的 " 美味 " 程度就是 4, 因为香蕉上的数是2,3,4,5,6,7,8,9,10, 其中恰好有 4 个质数 2,3,5,7.  
小猴有很多香蕉 , 请帮他找出最 " 美味 " 的香蕉的编号吧 , 这根香蕉吃起来一定有真理的味道 !

【输入格式】  
第 1 行， 2 个正整数n,m, 表示小猴有n根香蕉 , 并且香蕉上的所有数大小都不超过m。  
接下来n行，每行 3 个正整数，分别表示每一根香蕉的编号i, 香蕉上最小的数a\_i​和最大的数b\_i​ , 保证香蕉的编号各不相同 .  
【输出格式】  
输出共 1 行， 1 个整数，为最 " 美味 " 的香蕉的编号 , 如果有多个香蕉都是最美味的 , 则只输出  
最小的编号

【输入样例】

3 100

2333 2 10

10108899 29 29

666 1 100

【输出样例】

666

【样例解释】  
编号 2333 的香蕉上写的是整数 2,3,…,10, 美味程度为 4;  
编号 10108899 的香蕉上写的是整数 29, 美味程度为 1;  
编号 666 的香蕉上写的是整数 1,2,…,100, 美味程度为 25, 我们熟知 100 以内有 25 个质数 ;  
因此最美味的香蕉编号为 666  
【数据说明】  
对 40% 的数据1\le n \le 31≤*n*≤3;1\le a\_i \le b\_i \le m \le 1001≤*ai*​≤*bi*​≤*m*≤100.  
对 60% 的数据1\le n \le 101≤*n*≤10; 1\le a\_i \le b\_i \le m \le 10,0001≤*ai*​≤*bi*​≤*m*≤10,000.  
对 80% 的数据1\le n \le 1,0001≤*n*≤1,000;1\le a\_i \le b\_i \le m \le 10,000,0001≤*ai*​≤*bi*​≤*m*≤10,000,000;(b\_i-a\_i )\le 50,000(*bi*​−*ai*​)≤50,000.  
对 100% 的数据1\le n \le 100,0001≤*n*≤100,000;1\le a\_i \le b\_i \le m \le 10,000,0001≤*ai*​≤*bi*​≤*m*≤10,000,000;1\le i \le 2,147,483,6471≤*i*≤2,147,483,647