Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача А. Формула

Дадена е формулата 1?2?3?...?n = k. Вместо всеки въпросителен знак трябва да се постави знак "+" или "–" така, че да се получи верен израз. Например за n = 4 и k = 2 един възможен верен израз е 1 + 2 + 3 – 4 = 2. Напишете програма **`A.EXE`**, която въвежда цялото число k, $0 < k < 200$ и извежда най-малкото цяло положително n, за което съществува вярна формула от описания вид. Ако такова n не съществува, програмата трябва да изведе 0.

На всеки ред от стандартия вход се задава по едно число k за поредния тестов пример. Краят на входа се отбелязва с числото 0. Резултатът от всеки пример се извежда на отделен ред в стандартния изход.

Пример

Вход
```
1
2
0
```

Изход

```
1
3
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача В. Лампи

N лампи са номерирани последователно (N ≤ 30). Всяка лампа има свой собствен електрически ключ. Те са свързани с помощта на релета, така че ключа на $n^{-та}$ лампа да не може да действа (да се включва или изключва), освен когато $(N–1)^{-та}$ лампа е запалена, а всички други лампи с номера от 1 до (*N*–2) са изгасени. Първата лампа може да се запали или изгаси по всяко време. За ход се приема едно светване или изгасване на лампа. Първоначално всички лампи са изгасени. Напишете програма **`B.EXE`**, която намира за колко хода всички лампи ще бъдат запалени.

На всеки ред от стандартия вход е записан броят на лампите за отделен тестов пример. Резултатът от всеки пример се извежда на отделен ред в стандартния изход.

Пример

Вход

```
6
8
11
```

Изход

```
42
170
1365
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача C. Празно пространство

Подът на склад е правоъгълник със страни N ($5 \le N \le 1000$) и M ($5 \le M \le 1000$) като ъглите му са представени с точките (0, 0), (0, M), (N, 0) и (N, M) в ортогонална координатна система. Нека K паралелепипедни кутии са поставени в склада ($1 \le K \le 70$) по такъв начин, че страните им са паралелни на координатните оси. Кутиите може да се припокриват, могат да имат общи страни или части от страни, както и да се допират до стените на склада. Напишете програма **`C.EXE`**, която да намери правоъгълник F върху пода на склада с максимално лице, върху който да няма кутии. Правоъгълникът F може да има общи страни или части от страни с кутиите или да се допира до стените на склада.

Данните ще бъдат на стандартния вход. Първият ред ще съдържа цяло положително число T – броят на тестовите примери, които програмата трябва да реши. В първия ред на всеки тестов пример ще бъдат зададени числата N, M и K, разделени с по един интервал. Всеки от следващите K реда на тестовия пример съдържа по четири неотрицателни цели X_1, Y_1, X_2, Y_2, разделени с по един интервал, които описват положението на една кутия – (X_1,Y_1) са координатите на долния ляв, а (X_2, Y_2) – на горния десен ъгъл, $X_1 < X_2$, $Y_1 < Y_2$.

За всеки тестов пример програмата трябва да изведе на стандартния изход лицето на намерения максимален правоъгълник

Пример

Вход
```
2
20 20 4
4 0 9 10
16 4 19 20
6 7 15 14
4 7 6 10
20 20 3
16 6 20 10
8 0 12 12
0 12 14 16
```

Изход
```
96
96
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача D. Хипотезата на Голдбах

Хипотеза: За всяко четно число n, по-голямо или равно на 4, съществува поне една двойка от прости числа p_1 и p_2, такава, че $n = p_1 + p_2$.

Тази хипотеза нито е доказана, нито е отхвърлена. Никой не знае дали е вярна или не. Въпреки това, за дадено четно число може да се намери такава двойка, ако съществува. Задачата е да се напише програма, която за дадено четно число намира броя на всички двойки прости числа, които удовлетворяват хипотезата на Голдбах.

На входа е зададена редица от четни числа. За всяко число от редицата програмата трябва да изведе броя на двойките, споменати по-горе. Интересуваме се само от същесвено различните двойки, така че НЕ трябва да броим (p_1, p_2) и (p_2, p_1) като две различни двойки.

Всеки ред от входа съдържа по едно цяло число n. Числата са четни, по-голями или равни на 4 и по-малки от 1 000 001. Края на входа се маркира с числото 0.

Всеки ред от изхода съдържа по едно цяло число – броят на двойките за съответното число от входа.

Пример

Вход

```
6
10
12
36
0
```

Изход

```
1
2
1
4
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача Е. Десант

Една десантна група трябвало да проникне във военен обект, който много строго се охранява. Едно от най-големите препятствия били насочените фенери, постовени в точката P, които осветявали територията около обекта в **N** (1 <= **N** < 9) прави линии. Все пак разузнавачите доложили, че зоните, оставащи между тези линии не са осветени. Те доставили и списък на ***M*** (1 < ***M*** < 99) удобни за прикритие места, като ги номерирали от 1 до M и задали техните координати. С номер 1 означили обекта на нападението. Операцията трябвало да се извърши, така че десантчиците да не пресекат нито една от осветените линии. След продължителното разузнаване оставало само да се напише програма **`E.EXE`**, която по дадените данни да открие през кои от местата за прикритие трябва да премине групата, така че да достигне до целта без да пресича осветените линии.

Входните данни се четат от стандартния вход. В първия ред е записано цялото число ***K*** – броя на тестовите примери, на следващите редове са записани тестовите примери, организирани по следния начин: на първия ред на всеки тестов пример са записани целите числа ***N*** и ***M***, разделени с един интервал. Във втория ред са записани координатите на точката ***P***. В следващите ***N*** реда за всяка от осветените прави са записани координатите (разделени с един интервал) на още една точка от правата, различна от ***P***. Останалите ***M*** реда на файла съдържат координатите (разделени с един интервал) на поредната от зададените ***M*** места за прикритие, в нарастващ ред на номерата им. Всички координати са цели числа между −999 и 999 включително и са относно стандартна координатна система.

На стандартния изход програмата извежда ***K*** – по един за всеки тестов пример, всеки от които трябва да съдържа номерата на намерените от програмата обекти за прикритие, разделени с по един интервал. Ако такива не съществуват или ако обекта на нападението лежи върху някоя от дадените прави, тогава програмата трябва да запише 0 в изходния файл.

Пример

Вход

```
2
1 3
0 0
1 0
1 1
2 2
-1 -1
1 3
0 0
-1 -1
1 1
2 2
1 -1
```

Изход

```
2
0
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача F. Зайче в беда

Веднъж малкото бяло зайче, гонено от един ловец попаднало в лабиринт, които имал форма на квадратна дъска *N* x *N*. В него чакал големия лош вълк, които предварително изкопал дупки, където зайчето да падне и той да го хване по-лесно. В последния момент зайчето с ужас разбрало, че може да се движи само в посока надолу и надясно и че изхода от лабиринта е чак в долния десен ъгъл на дъската.

Зайчето трябвало да разбере каква е вероятността да излезе от лабиринта без да падне в някоя дупка. За целта трябвало да изчисли броя пътища от входа до изхода на лабиринта, като успяло да се снабди с картата на този лабиринт. Картата е зададена с размер *N*, като местата на дупките са означени с 0, а проходимите места с 1. Напишете програма **`F.EXE`**, която пресмята търсения брой пътища.

Данните се четат от стандартния вход, където на първия ред е записано цялото число *K* – броя на тестовите примери, а за всеки тестов пример се подава числото *N* <= 40 следват *N* реда с по *N* числа, като на всяка клетка съответства по едно чило 1 или 0 (проходима клетка или дупка).

На стандартния изход се извеждат *K* числа – по едно за всеки тестов пример, показващи броя на пътищата в съответния тестов пример. Всяко едно от числата се извежда на отделен ред.

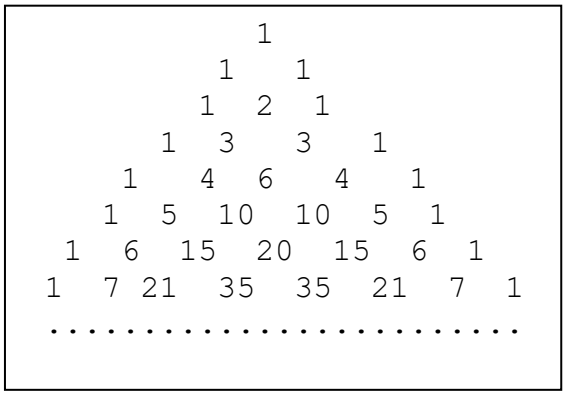
Пример

Вход:	Изход:
1 4 1 1 1 1 1 1 0 1 1 0 1 1 1 1 1 1	2

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача G. Паскал

Триъгълникът на Паскал е триъгълен масив от цели числа, подредени така, че всяко число от по-долния ред е равно на сумата от двата непосредствени съседи отгоре, както е показано в схемата (най-левите и най-десните числа, както и най-горното, са единици):



Всяко число от триъгълника се определя с два индекса (n, k), които показват неговото положение на k-то място (броено отляво-надясно) в n-тия ред (k = 1, 2, 3, ...; n = 1, 2, 3, ...). Две числа от триъгълника наричаме съседни, ако те са непосредствено едно до друго в един и същ ред, или едното от тях е непосредствено отгоре диагонално разположено отляво или отдясно. Област наричаме такова множество от числа в триъгълника, че всеки две от тях могат да се свържат с верига от съседни числа. Вашата задача е при зададено с индексите си (n, k) число да намерите броя на числата в най-голямата област от четни числа, на която то принадлежи.

Вашата програма **`G.EXE`** трябва да прочете няколко реда от входния файл, всеки съдържащ двете цели положителни числа n и k, разделени с интервал ($k \le n \le 10^9$). Краят на входния файл е отбелязан с една нула на последния ред. За всяка от въведените двойки n и k вашата програма трябва да изведе на отделен ред намерения отговор.

Пример:

Вход

```
5 3
6 2
0
```

Изход

```
6
0
```

Шести междууниверситетски турнир по програмиране
Шумен, 10 май 2003 г.

Задача Н. Неизоморфни графи

Разглеждаме неориентирани графи без кратни ребра и без примки. Два графа $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$ са изоморфни, когато съществува взаимно еднозначно изображение f на V_1 върху V_2, такова, че двойката $\{f(v'), f(v'')\}$ е ребро в графа G_2, тогава и само тогава, когато двойката $\{v', v''\}$ е ребро в G_1.

Да се напише програма **`H.EXE`**, която намира броя на всички неизоморфни графи с n върха и m ребра ($1 \le n \le 7$, $0 \le m \le n(n-1)/2$). На всеки ред от стандартия вход са записани числата n и m за отделен тестов пример. Резултатът от всеки пример се извежда на отделен ред в стандартния изход.

Пример

Вход

```
4 2
5 7
```

Изход

```
2
4
```