НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране"

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## А. Замервания

Двама приятели предприели дълго пътуване към морето. За съжаление километража в колата им не работи и те не знаят колко мили са изминали. За щастие, единият от тях има хронометър, с помощта на който записват скоростта си и общото време, което са карали. За съжаление стратегията им на водене на записки е меко казано странна и сега на вас се пада честта да изчислите общото пропътувано разстояние. Примерни техни записки са:

| *Скорост – мили в час* | *Общо изминало време в часове от началото на пътуването им* |
|---|---|
| 20 | 2 |
| 30 | 6 |
| 10 | 7 |

Първите два часа са се движили с 20 мили в час, следващите 6 - 2 = 4 часа с 30 мили в час, а последния 7 - 6 = 1 час с 10 м/ч. Следователно изминатото разстоянието е 2 * 20 + 4 * 30 + 1 * 10 = 40 + 120 + 10 = 170 мили. Забележете, че общото изминало време е винаги спрямо началото на пътуването.

Всеки тест започва с ред, съдържащ цялото $N$, $1 \le N \le 10$. Следват $N$ двойки цели числа - скоростта в мили в час S, $1 \le S \le 100$ и общото изминалото време T, $1 \le T \le 12$. Двойките са винаги подредени във възходящ ред на T. -1 маркира края на входа.

За всеки тестов пример извеждайте на отделен ред на стандартния изход търсеното изминато разстояние в мили. Спазвайте показания изходен формат.

| **Вход** | **Изход** |
|---|---|
| 3<br>20 2<br>30 6<br>10 7<br>2<br>60 1<br>30 5<br>4<br>15 1<br>25 2<br>30 3<br>10 5<br>-1 | 170 miles<br>180 miles<br>90 miles |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**
Школа „Състезателно програмиране”
**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## B. Суми от степени на 2

Вчера Иван видя във Facebook, че 2016 може да се запише като сума от степени на двойката по следния начин: $2016 = 2^{10} + 2^9 + 2^8 + 2^7 + 2^6 + 2^5$. Той се замисли дали няма и други такива представяния и ако има, колко са те. За числата от 2 до 5 той откри следните представяния:

$2 = 2^1$ и $2 = 2^0 + 2^0$
$3 = 2^0 + 2^1$ и $3 = 2^0 + 2^0 + 2^0$
$4 = 2^2$, $4 = 2^1 + 2^1$, $4 = 2^1 + 2^0 + 2^0$ и $4 = 2^0 + 2^0 + 2^0 + 2^0$
$5 = 2^2 + 2^0$, $5 = 2^1 + 2^1 + 2^0$, $5 = 2^1 + 2^0 + 2^0 + 2^0$ и $5 = 2^0 + 2^0 + 2^0 + 2^0 + 2^0$

Нататък обаче му е трудно да продължи, а за 2016 очевидно сметките няма да станат само с лист и химикал. Поради тази причина Иван ви моли да напишете програма, която да пресмята броя на начините, по които цяло положително число $N$ може да се запише като сбор от естествени числа, всяко от които е степен на двойката. Обърнете внимание, че редът на събираемите в сумите не е от значение (например сумите $2^1 + 2^1 + 2^0$, $2^1 + 2^0 + 2^1$ и $2^0 + 2^1 + 2^1$ са едно и също представяне на числото 5 като сбор от естествени числа, които са степени на двойката).

За всеки тестов пример от стандартния вход се въвежда цялото число $N$ ($1 \le N \le 3000$).

За всеки тестов пример извеждайте на отделен ред на стандартния изход броя на начините, по които $N$ може да се запише като сбор от естествени числа, всяко от които е степен на двойката.

| **Вход** | **Изход** |
|---|---|
| 6 | 6 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**
Школа „Състезателно програмиране”
**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## C. Скоби

Даден ви е низ от скоби - тоест низ, съставен от символите {'(', ')', '{', '}', '[', ']'}. Пита се дали зададеният низ е валиден низ от скоби - тоест ако се поставят числа и аритметични знаци между скобите би могло да се получи валиден аритметичен израз.

Напишете програма, която по зададен символен низ проверява дали синтаксисът на скобите е правилен – т.е. дали за всяка отваряща скоба има съответна затваряща и обратно, както и дали са правилно разположени.

Примерни валидни скобни низове са "()()", и "({{[]([])}}())[[{}]]", докато невалидни са "((", ")(", "([})", "({)}", и "(){}[()]]".

Всеки тест ще се състои от един непразен стринг, зададен на отделен ред на стандартния вход и ще съдържа не повече от 3000 символа.

За всеки тест извеждайте на нов ред на стандартния изход “yes”, прочетен низ е валиден низ от скоби и “no”, в противен случай.

| **Вход** | **Изход** |
|---|---|
| ({{[](([])}}())[[{}]] | yes |
| (){}[()]] | no |
| ({)} | no |
| { | no |
| [] | yes |
| {}{} | yes |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## D. Нарушения

В един град всички улици са еднопосочни. В града има $n$ кръстовища, номерирани с целите числа от 1 до $n$. Даден е списък на двойки кръстовища $p$ и $q$, които са свързани с еднопосочна отсечка от улица, така че по тази отсечка от улица няма други кръстовища. За всеки две кръстовища $p$ и $q$ съществува най-много една еднопосочна отсечка от улица в посока от $p$ към $q$, или в посока от $q$ към $p$.

Опитваме се да спазваме правилата за движение в града, но невинаги е възможно да отидем с кола от кръстовище $a$ до кръстовище $b$. С колко най-малко нарушения, обаче, може да се придвижим от $a$ до $b$? Всяко навлизане в посока обратна на разрешената в еднопосочна отсечка от улица се брои за едно нарушение.

Напишете програма, която намира минималния брой на нарушенията.

На първия ред на стандартния вход е зададен броят на тестовете. Всеки тест започва с целите $n$, $m$, $a$ и $b$, където $n$, $1 \le n \le 200\,000$ е броят на кръстовищата, $m$, $1 \le m \le 400\,000$ е броят на еднопосочните отсечки, $a$ е номерът на кръстовището, от което тръгваме и $b$ е номерът на кръстовището, в което трябва да отидем. Следват $m$ реда, като на всеки от тези редове са дадени по две числа $p$ и $q$ – номерата на кръстовищата, за които съществува еднопосочна отсечка от улица с посока от $p$ към $q$.

За всеки тест програмата трябва да изведе на отделен ред на стандартния изход едно цяло число – намерения минимален брой. Ако е възможно да се премине без нарушения – програмата трябва да изведе 0. Ако въобще не е възможно да се премине – програмата трябва да изведе главната латинска буква X.

| **Вход** | **Изход** |
|---|---|
| 2<br>4 4 1 4<br>4 3<br>3 2<br>1 2<br>4 2<br>4 3 1 4<br>4 3<br>3 2<br>4 2 | 1<br>X |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

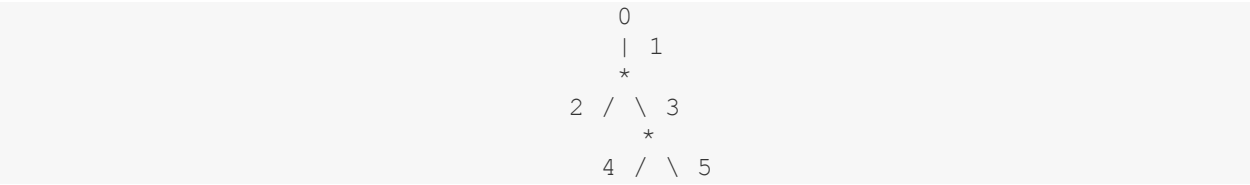
Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## Е. Дълбочина на дърво

Дадено е двоично дърво с $N$, $3 \le N \le 99999$ върха, номерирани с числата от 1 до $N$. Всеки връх на дървото има 0 или 2 наследници, като дължината на свързващите ги ребра е 1. Коренът на дървото е с номер 1. Той е свързан с фиктивен възел с номер 0, като разстоянието между тях е 1.

Да се намерят разстоянията от фиктивния възел до всички върхове на дървото, т.е. да се намери дълбочината (depth) на всеки възел, ако допуснем, че коренът на дървото е с дълбочина 1. Например:




*Връх 1 е на разстояние 1 от фиктивния възел. Върхове 2 и 3 са на разстояние 2, а върхове 4 и 5 – на разстояние 3.*

Всеки тестов пример започва с числата $N$ и $C$, $1 \le C \le N$. $N$ е броят на върховете, а $C$ е броят на зададените тройки – връх, ляв и десен наследник. Следват $C$ реда с по три числа $E_i$ ($1 \le E_i \le N$), $L_i$ и $R_i$ ($2 \le L_i \le N$; $2 \le R_i \le N$). $E_i$ е номер на връх с наследници върховете $L_i$ и $R_i$. За край на входа служат две нули.

За всеки тестов пример трябва да се отпечатат по $N$ числа, всяко на отделен ред, като $i$-то число е дължината от фиктивния връх до върха с номер $i$, $1 \le i \le N$.

| **Вход** | **Изход** |
|---|---|
| 5 2<br>3 5 4<br>1 2 3<br>7 3<br>1 3 7<br>7 5 2<br>3 4 6<br>0 0 | 1<br>2<br>2<br>3<br>3<br>1<br>3<br>2<br>3<br>3<br>3<br>2 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## F. Garbage

Навярно не много от вас знаят, че скоро ще бъде отворен първият завод за преработване на боклук в България. Част от работата, която ще се върши там е да се компресира боклукът. За да се прави това, той се поставя в правоъгълна "стая", като всяка от четирите стени е бутало, което може да натисне боклука от съответната страна. Боклукът, от друга страна, е разнороден и се нуждае от различно количество енергия за натискане, за да се компресира.

Можете да си представите боклука и стаята като правоъгълна матрица с $N$ реда и $M$ колони. Във всяка клетка на матрицата има по едно число $A_{ij}$ между 0 и 9, включително - какъв "натиск" се изисква, за да бъде компресирана тя. Когато някоя от четирите стени "натиска" оставащия боклук, е нужна енергия, равна на най-голямото число в съответната страна на матрицата - краен ред за горната и долната стена, и крайна колона за лявата и дясната.

Например, нека имаме компресираща стая с 3 реда и 4 колони, като боклукът в нея е със следната "твърдост":

| | | | |
|---|---|---|---|
| 6 | 8 | 7 | 2 |
| 3 | 0 | 9 | 1 |
| 4 | 2 | 9 | 1 |

Ако натиснем с горната стена (ред с числа 6, 8, 7, 2) ще ни е нужна сила 8, тъй като това е най-твърдата клетка в този ред. Аналогично, за долната стена ще ни е нужна сила 9, за лявата - сила 6, а за дясната - сила 2. След натискане въпросният ред или колона изчезва - вече се счита за компресиран и не играе роля в следващи натискания. Целта е целият боклук (тоест всички клетки) да бъдат компресирани. За да е ефективен заводът, се изисква това да стане с минимална обща използвана сила.

Оказва се, че има значение кои бутала и в какъв ред ползваме. Ако, например, ползваме само горното бутало ще са ни нужни 8 + 9 + 9 = 26 единици сила. Ако вместо това ползваме горното, дясното, дясното, лявото, долното за 8 + 1 + 9 + 4 + 2 = 24.

Чувайки за вече легендарната креативност на Ели (най-вече в това да предоставя сложните си проблеми на вас), шефовете на завода са я назначили за ръководител на отдела по натискането. Разбира се, момичето ви дава възможността да блеснете, като напишете програма, която намира с колко най-малко енергия може да бъде компресиран целият боклук.

**Вход**

От първия ред на стандартния вход се въвежда броя тестовите примери. Всеки от тях започва с две цели, положителни числа, разделени с интервал - броя редове ***N*** и броя колони ***M*** на стаята. Следват ***N*** на брой реда, всеки съдържащ по ***M*** едноцифрени числа, разделени с интервали - твърдостта на боклука във всяка от клетките.

**Изход**

За всеки тестов пример извеждайте на стандартния изход едно цяло число - минималната енергия, нужна за компресирането на всичкия боклук.

**Ограничения:**

- $1 \leq N, M \leq 100$
- $0 \leq A_{ij} \leq 9$

| **Вход** | **Изход** |
|---|---|
| 2<br>3 4<br>6 8 7 2<br>3 0 9 1<br>4 2 9 1<br>8 7<br>9 5 9 9 8 9 1<br>1 3 7 0 1 7 7<br>6 0 7 3 7 0 3<br>2 2 6 1 5 4 8<br>6 9 9 2 3 2 7<br>4 6 7 3 1 1 3<br>1 6 7 1 2 6 7<br>4 4 7 3 9 8 9 | 24<br>62 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## G. Прости числа

Да се напише програма, която намира броя на простите числа в даден затворен интервал.

Първото число от входа е броят на примерите на входа. Следващите двойки числа a и b, (a < b < 1000000) задават интервала, в който ще се търси. Двете числа са цели и положителни.

За всеки пример от входа се извежда на отделен ред намерения брой.

| **Вход** | **Изход** |
|---|---|
| 3<br>1 5<br>13 14<br>20 22 | 3<br>1<br>0 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## Н. Анаграма

Низът X е анаграма на низа Y, ако X може да бъде получен от разместването на символите на Y в някакъв ред. Не е позволено премахването или добавянето на никакви символи. Например всеки от низовете "baba", "abab", "aabb" и "abba" е анаграма на "aabb", а низовете "aaab", "aab" и "aabc" на са анаграма на "aabb".

По зададено множество от низове S се интересуваме от най-голямото му подмножество, в което няма два или повече низа, които да са анаграми един на друг. S може да се счита за подмножество, ако отговаря на горното условието.

Всеки тестов пример е зададен на един не празен ред, съдържащ низовете от S, разделени с един или няколко интервала. Всяко S съдържа между 1 и 50 низа, всеки от които с дължина между 1 и 50.

За всеки тестов пример на стандартния изход да се изведе по едно число – броя на низовете в исканото подмножеството.

| **Вход** | **Изход** |
|---|---|
| abcd abac aabc bacd<br><br>wlrb m bhc arz wk yhi dqs dxr mowfr sjyb<br><br>ab ba<br><br>z | 2<br>10<br>1<br>1 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## I. Различни стойности

Нека са дадени числата $a_1$, $a_2$, ..., $a_n$. След поставяне на операции "+" и "-" между числата $a_i$ и $a_{i+1}$ за i = 1, 2, ..., n - 1 може да се пресметне стойността на получения аритметичен израз.

Колко различни стойности може да получим по този начин?

На стандартния входа се задават редици от числа - всяка редица на отделен ред.

За всяка редица от входа се отпечатва на нов ред едно число - броят на различните стойности на получените изрази.

**Ограничения**:

- Всички числа редиците са цели в интервала [1, 100].
- $2 \le n \le 20$.

| **Вход** | **Изход** |
|---|---|
| 1 1<br>1 2 3 4 5 6 7 8<br>30 20 40 10 | 2<br>34<br>8 |

НБУ
НОВ БЪЛГАРСКИ УНИВЕРСИТЕТ
София 1635, ул. Монтевидео 21
тел.: 55 81 37, 55 21 35, факс: 957 19 30

**Департамент Информатика**

Школа „Състезателно програмиране”

**СЪСТЕЗАНИЕ, 12 ноември 2016 г.**

## J. Прави

Разглеждаме точките с целочислени координати **(x, y)**, за които 0 ≤ **x** ≤ **a** и 0 ≤ **y** ≤ **b**. Напишете програма, която намира колко прави минават през поне две от тези точки.

Програмата трябва да обработи няколко тестови примера, всеки зададен на отделен ред на стандартния вход, съдържащ числата **a** и **b** (0 < **a**, **b** < 3000).

За всеки тестов пример изведете на отделен ред на стандартния изход броя на търсените прави.

| **Вход** | **Изход** |
|---|---|
| 2 2<br>10 10 | 20<br>3296 |