Нов български университет

Департамент Информатика

Школа „Състезателно програмиране”

СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача А. Сортиране

Да се напише програма, която сортира дадена редица от цели положителни числа по следния начин: за всеки две числа **X** и **Y**, **X** е по-напред в наредбата от **Y**, ако **X** се среща в дадената редица повече пъти от **Y**. Ако **X** и **Y** се срещат еднакъв брой пъти, то тогава **X** е по-напред в наредбата от **Y**, ако има член на редицата със стойност **X,** който е преди всички членове със стойност **Y** в дадената редица.

Входът съдържа много примери (редици). Всеки от тях е на отделен непразен ред и съдържа числата, които трябва да бъдат сортирани. Всяко от числата е по-голямо от **0** и по-малко или равно на **1 000 000 000**. Числата могат да бъдат разделени едно от друго с повече от един интервал. Общият брой на тези числа не надхвърля **1 000**.

За всеки пример на отделен ред да се отпечати сортираната редица. Всеки две числа трябва да бъдат разделени точно с един интервал. В края на всеки от тези редове не трябва да има интервали или табулации. Изходът не трябва да съдържа празни редове!

Пример:

Вход	**Изход**
2 1 2 1 2 1 3 3 3 2 2 2 1 1 11 33 11 77 54 11 25 25 33 88 99 1 99 99 1 81 99 81 88	2 2 2 1 1 1 1 1 3 3 3 2 2 2 11 11 11 33 33 25 25 77 54 99 99 99 99 88 88 1 1 81 81

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

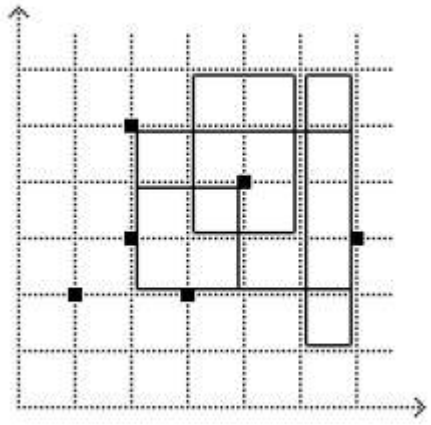
Задача В. Точки върху страните на правоъгълник

В равнината са дадени N точки с целочислени координати. Също така са дадени и P правоъгълника, зададени с целочислените координати на долния си ляв и горния десен ъгъл. Страните на тези правоъгълници са успоредни на координатните оси. За всеки от правоъгълниците търсим броя на точките измежду зададените, които лежат върху страните му. Точки, намиращи се във вътрешността на правоъгълника или извън нея не се броят.

Входът съдържа T ($1 \le T \le 15$) примера. Всеки пример започва с броя на точките N ($1 \le N \le 300\,000$). Следващите N реда съдържат координатите на поредната точка – две цели числа X и Y ($1 \le X, Y \le 109$). Няма повече от една точка с еднакви координати. Следващият ред съдържа броя на правоъгълниците P ($1 \le P \le 100\,000$). Следват P реда, всеки с по четири цели числа X_1, Y_1, X_2, Y_2 ($1 \le X_1 < X_2 \le 109$, $1 \le Y_1 < Y_2 \le 109$). Това са координатите на долния ляв ъгъл (X_1, Y_1) и горния десен ъгъл (X_2, Y_2) на поредния правоъгълник.

За всеки пример да се отпечатат по P числа, всяко на отделен ред, като ред i трябва да съдържа търсения брой точки за i-тия прочетен правоъгълник от входа.

Пример.

Вход		**Изход**
1 6 1 2 3 2 2 3 2 5 4 4 6 3 4 2 2 4 4 2 2 6 5 3 3 5 6 5 1 6 6		3 4 0 1

Нов български университет

Департамент Информатика

Школа „Състезателно програмиране”

СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача С. Гъбките на Червената шапчица

На път за къщичката на баба си, Червената шапчица решила да набере гъбки в гората. Като една модерна Червена шапчица, тя попитала чичко Гугъл, къде растат гъбките в гората? И получила подробна квадратна карта, разделена на *n*x*n* квадратчета и във всяко квадратче – броят на гъбките-манатарки в това квадратче. В горния ляв ъгъл се мъдрела снимката й от Гугъл-чата, а най-долу вдясно – силует с надпис "баба" (колко пъти й беше казвала – сложи си една снимка на профила, да те познават хората!). Като една модерна Червена шапчица, тя се заплеснала с Фейсбука (главно да разглежда снимките на Вълка), и когато тръгнала за бабината къщичка, било вече почти тъмно. Затова решила да не се бави много-много с брането на гъби, Вълкът внимателно следял нейния Туитер и можело да се досети къде отива. Като върви, щяла да следва правилото: съседно квадратче надясно или съседно квадратче надолу – по картата на чичко Гугъл. Така щяла да набере досатъчно гъби минавайки през точно 2*n* - 2 квадратчета преди да стигне до къщичката на баба си. Било и лесно – задаваш на Гармина картата, и той те води по най-добрия маршрут, по който да набереш най-много гъби. Речено-сторено. Жива, здрава и радостна, Червената шапчица стигнала бабината къщичка с пълна кошничка с гъбки-манатарки. Там я чакал лошия вълк, и по-натам приказката я знаете...

Вашата задача е да познаете колко гъбки има в кошничката на Червената шапчица.

Първият ред на входа съдържа едно число N – броя на тестовете. Първият ред на всеки тест съдържа числото *n* – размера на квадратната карта (3 < *n* < 101). Всеки от следващите *n* реда се състои от *n* числа – броя на гъбките в малкото квадратче на съответната позиция от картата на чичко Гугъл.

За всеки тест да се изведе на отделен ред броят на събраните от Червената шапчица гъбки-манатарки.

Пример:

Вход	**Изход**
2 3 0 1 1 0 4 2 1 1 1 5 1 1 1 1 1 0 0 3 4 3 0 1 2 0 1 1 1 1 0 1 2 4 0 4 0	8 15

Нов български университет

Департамент Информатика

Школа „Състезателно програмиране”

СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача D. Максимуми и минимуми на функция

Дадена е функция *f*, дефинирана върху множеството от целите числа. Дефинираме минимум на такава функция в точка *n*, ако $f(n) < f(n-1)$ и $f(n) < f(n+1)$. Аналогично се дефинира и максимум на функцията. Да се намери броят на минимумите и максимумите на такава функция, зададена с редица функционални стойности.

От стандартния вход трябва да се прочетат всички примери. Всеки пример започва с цяло положително число *N* – броят на точките, където е определена функцията ($N < 10^{10}$). Следват стойностите на функцията в точките 1, 2, 3, ..., *N* – цели числа в интервала[-1000, 1000].

За всеки пример на стандартния изход да се изведат две числа – броя на минимумите и броя на максимумите на функцията, отделени с един интервал.

Пример:

Вход	**Изход**
10 3 2 3 2 2 7 8 2 9 1	2 3

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача Е. Събиране на прости дроби

Гошко е ученик в 5-ти клас, и както повечето деца – гений на компютрите и зле по математика. Простите дроби съвсем не му се виждат прости. А има за домашно събиране на прости дроби. Нито му се пише, нито знае как да събере проклетите дроби. Е – изходът е само едни – да си напише програма за написване на домашното. Сам едва ли ще може, но за батковците от Фейсбука това сигуно ще е съвсем фасулска работа. И на стената си Гошко написва:

Помоооощ – някой да напише програма за събиране на две прости дроби! И непременно да получи цяло число или несъкратима дроб! И искам да си отпечатам домашното на принтера! Ето ви и пример, а числата не се големи – най-големите са трицифрени. Който ми напише програмата, ще получи от мен помощ за фермата!

Пример:

Вход	**Изход**
1/2+1/2= 1/2+1/3=	1/2+1/2=1 1/2+1/3=5/6

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача F. Ротация

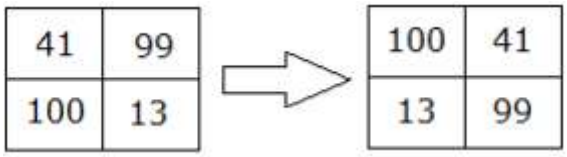
Разглеждаме таблица с размери 2x2, съдържаща целите положителни числа **A**, **B**, **C** и **D**, всичките не по-големи от 100.

A	**B**
C	**D**

Дефинираме **стойност** на таблица, равна на:

A/C – B/D

Търсим **минималния** брой 90 градусови ротации по посока на часовниковата стрелка, необходими, за да се **максимизира** стойността на дадена таблица. По-долу е показана една такава ротация:



Всеки четири последователни числа от **входа** са числата **A**, **B**, **C** и **D**, с които запълваме поредната таблица по начина показан по-горе.

За всяка поредна таблица на отделен ред трябва да се **изведе** търсения минимален брой ротации, нужни за максимизирането на стойността ѝ.

Пример:

Вход	**Изход**
1 2 3 4 5 9 7 2 41 99 100 13	2 0 1

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача G. Идентификатори

С цел да упражни по-голям контрол над своите граждани, правителството се решило на радикална мярка – всеки гражданин да има малък микрокомпютър, хирургично имплантиран в лявата си китка. Този компютър ще съдържа цялата лична информация, както и един предавател, който ще позволи да се следи движението на човека и да се записва от централен компютър.

Съществена част от изработката на поредния компютър е генерирането на уникален идентификационен код, състоящ се от поне един и не повече от 50 символа. Самите символи могат да бъдат само някои от 26-те малки букви от английската азбука.

Да предположим, че поредното множество от кодове трябва да съдържа точно три повторения на **a**, две на **b** и едно на **c**. Генерирането на първия уникален идентификационен код става като наредим в нарастващ ред тези символи. Т.е първия код е **aaabbc**. Всеки следващ код може да бъде получен от предишния, след разместване на някои два символа, така че получения нов код да бъде уникален. Освен това трябва да е следващият в лексикографската наредба. Например, започвайки от **aaabbc**, след определен брой правилни размествания ще стигнем до кода **aacbba**. Следващите три кода, отговарящи на изискванията са:

abaabc
abaacb
ababac

Напишете програма, която чете от стандартния вход множество от кодове. Всеки код е зададен на отделен непразен ред и може да съдържа интервали преди и/или след него. За всеки входен код се генерира и извежда на отделен ред на стандартния изход код удовлетворяващ вече споменатите изисквания. Ако такъв код на съществува, трябва да изведете **`Impossible!`**.

Пример:

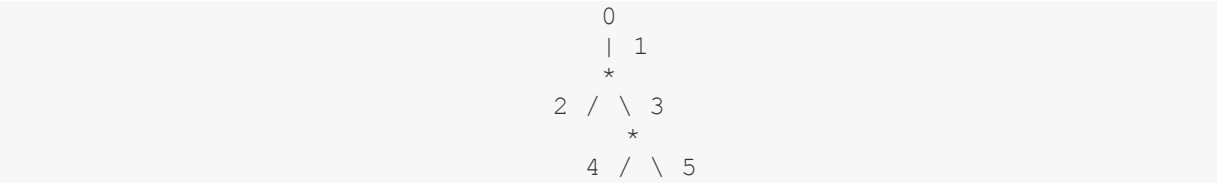
Вход	**Изход**
abaacb cbbaa	ababac Impossible!

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача Н. Дърво

Дадено е двоично дърво с N върха (3 ≤ N ≤ 99999), номерирани 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.

Напишете програма, която да обработи няколко тестови примера. Всеки пример започва с числата N и C (1 ≤ C ≤ 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 ≤ E_i ≤ N), L_i и R_i (2 ≤ L_i ≤ N; 2 ≤ R_i ≤ 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 ≤ i ≤ N).

Пример:

Вход	**Изход**
5 2 3 5 4 1 2 3 7 3 1 3 7 7 5 2 3 4 6 0 0	1 2 2 3 3 1 3 2 3 3 3 2

Нов български университет

Департамент Информатика

Школа „Състезателно програмиране”

СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача I. Антиаритметика

Пермутацията от n е биективна функция на първите n естествени числа: 0, 1, ... n–1. Една пермутация p се нарича антиаритметична, ако в нея няма подредица, която да формира аритметична прогресия с дължина повече от 2, т.е. не съществуват три индекса $0 \le i < j < k \le n$, такива че (p_i , p_j , p_k) формира аритметична прогресия.

Например, поредицата (2, 0, 1, 4, 3) е антиаритметична пермутация на 5. Поредицата (0, 5, 4, 3, 1, 2) не е антиаритметична пермутация, защото първия, петия и шестия елемент (0, 1, 2), както и втория, четвъртия и петия елемент (5, 3, 1) формират аритметични прогресии.

Да се напише програма, която да определя дали дадена пермутация от n е антиаритметична. Входът се състои от няколко тестови примера, последвани от ред съдържащ 0. Всеки тестов пример е ред от входа, съдържащ естествено число n ($3 \le n \le 10000$), последвано от две точки („:“) и n-те числа на пермутацията, разделени с интервал.

За всеки тестов пример на стандартния изход да се изведе съответно `yes` или `no`, в зависимост от това дали пермутацията е антиаритметична или не.

Пример:

Вход	**Изход**

`3: 0 2 1`
`5: 2 0 1 3 4`
`6: 2 4 3 5 0 1`
`0`
	`yes`
	`no`
	`yes`

Нов български университет
Департамент Информатика
Школа „Състезателно програмиране”
СЪСТЕЗАНИЕ, 15 януари 2012 г.

Задача J. Анаграми

Анаграма е вид игра на думи състояща се в пренареждане на буквите от дума или фраза така, че да се образува нова дума или фраза като всяка буква от оригинала се използва само веднъж. Напишете програма, която да проверява дали един стринг е анаграма на дадена дума или не е. Никой от посочените стрингове няма да има дължина повече от 50 символа.

Входът на програмата се състои от няколко тестови примери. Всеки един от тях започва с един ред – началната дума. На следващия ден се задава число n ($1 \le n \le 30000$), броя следващи думи за проверка. Край на тестовите примери се отбелязва със символ „точка“.

За всяка една от думите програмата трябва да извежда на отделен ред дали тя е същата като началния стринг (изписва се `IDENTICAL`), дали е анаграма (`ANAGRAM`) или няма връзка между тях (`NOT AN ANAGRAM`).

Пример:

Вход	**Изход**
cares 5 scare races cares another acres .	ANAGRAM ANAGRAM IDENTICAL NOT AN ANAGRAM ANAGRAM