



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача А. КАРТИ

Петър обича да съставя ребуси с числа. Веднъж открил пачка празни картончета в чекмеджето си и написал по едно цяло число от двете страни на всяко картонче. Подредил картончетата по следния начин:

$$\text{SUM} = \square - \square + \square - \square + \dots + \square - \square.$$

и се замислил каква ли е най-малката възможна стойност на SUM, която може да бъде получена чрез разместване на картончетата в произволен ред (и обръщане на някои от тях на другата страна, ако е необходимо). Напишете програма, която размества картончетата така, че стойността на израза SUM да е минимална.

Вход: Програмата трябва да може да обработва няколко тестови примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. На първия ред на всеки тестов пример ще бъде зададено едно четно число N – броят на картончетата в примера, а на i -тия от следващите N реда – двете цели числа a_i и b_i , които Петър е написал на лицето и на гърба на едно от картончетата.

Изход: За всеки тестов пример, на отделен ред на **стандартния изход**, програмата трябва да изведе намерената минимална възможна сума.

Ограничения: $2 \leq N \leq 100000$, N – четно; $-2000 \leq a_i \leq 2000$, $-2000 \leq b_i \leq 2000$.

Пример:

Вход	Изход	Обяснение на примерите
2	-34	В първия тест картончетата могат да се подредят в следния ред: $1^{\text{во}}$, $2^{\text{по}}$, $3^{\text{то}}$, $5^{\text{то}}$, $4^{\text{то}}$, $6^{\text{то}}$ и сумата е: $(-8) - 5 + (-3) - 7 + (-7) - 4 = -34$.
6	-155	
-8 12		Във втория тест картончетата могат да се подредят в следния ред: $2^{\text{по}}$, $1^{\text{во}}$, $4^{\text{то}}$, $3^{\text{то}}$, $5^{\text{то}}$, $8^{\text{мо}}$, $6^{\text{то}}$, $9^{\text{то}}$, $7^{\text{мо}}$, $10^{\text{то}}$ и сумата е: $62 - 70 + 59 - 81 + 40 - 76 + 35 - 85 + 57 - 96 = -155$
0 5		
7 -3		
10 -7		
-2 7		
1 4		
10		
70 70		
62 73		
81 65		
59 77		
99 40		
35 88		
80 57		
76 67		
85 57		
53 96		



Task A. CARDS

Peter likes puzzles with numbers. Once he found a batch of empty paper cards in his drawer, wrote two integers on each card – one number by side, and thought of the following puzzle: what is the smallest possible value which could be obtained by putting all cards in some order (and upturning some of the cards if necessary) in the expression of the following form:

$$\text{SUM} = \square - \square + \square - \square + \dots + \square - \square.$$

After a while Peter came up with a solution. Could you do that too? Write a program to solve the puzzle described above.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of the test cases. The first line of each test case will contain an even integer N – the number of cards. The i -th of the following N lines will contain the two integer numbers a_i and b_i that were written on the sides of the i -th card.

Output: For each test case, on a separate line of the **standard output**, the program has to print the minimal value that can be obtained by ordering all cards to an expression as described above.

Restrictions: $2 \leq N \leq 100000$, N – even; $-2000 \leq a_i \leq 2000$, $-2000 \leq b_i \leq 2000$

Example:

Input	Output	Explanation of examples
2 6 -8 12 0 5 7 -3 10 -7 -2 7 1 4 10 70 70 62 73 81 65 59 77 99 40 35 88 80 57 76 67 85 57 53 96	-34 -155	In the first case the cards could be ordered in the following way: 1 st , 2 nd , 3 rd , 5 th , 4 th , 6 th and the sum is: $(-8) - 5 + (-3) - 7 + (-7) - 4 = -34$. In the second test case the cards could be ordered in the following way: 2 nd , 1 st , 4 th , 3 rd , 5 th , 8 th , 6 th , 9 th , 7 th , 10 th and the sum is: $62 - 70 + 59 - 81 + 40 - 76 + 35 - 85 + 57 - 96 = -155$



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача В. SOS

Кораб в океана подал сигнал за бедствие и няколко кораба се отзовали на сигнала. **Един от тях** пристигнал пръв и спасил екипажа. Напишете програма да определи кой е този кораб, ако са дадени координатите на всички кораби в декартова координатна система и максималните скорости, с които могат да се движат притеклите се на помощ кораби.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Всеки тестов пример започва с координатите на потъващия кораб, последван от броя на готовите да се притекат на помощ кораби и след това, за всеки кораб – името, координатите и скоростта му. Всички числа са цели и се записват с не повече от 4 десетични цифри. Името на кораба се състои от главни и малки латински букви.

Изход: За всеки пример програмата трябва да изведе на **стандартния изход** името на кораба, спасил бедстващия екипаж.

Пример:

Вход	Изход
1 100 100 5 Ruse 120 120 3 Varna 100 150 4 Berlin 85 110 5 Rome 90 46 6 Pomorie 130 89 5	Berlin



Task B. SOS

Ship in the ocean transmitted a distress signal and several ships responded to the signal. **One of them** arrived first and rescued the crew. Write a program to determine the ship that was the first, if the coordinates in Cartesian coordinate system of all ships and the maximum speeds of the rescue ships are given.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case starts with the coordinates of the sinking ship, the number of ships willing to help, and for each of these ships – its name, coordinates and speed. All numbers are integer with no more than 4 decimal digits. Names of the ships consist of upper and lower case Latin letters.

Output: For each example, on the **standard output**, the program has to print the name of the ship that rescued the crew.

Example:

Input	Output
1 100 100 5 Ruse 120 120 3 Varna 100 150 4 Berlin 85 110 5 Rome 90 46 6 Pomorie 130 89 5	Berlin



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача С. КОЛЕКТИВИ

Студентите от последния курс в бакалавърската програма Информатика на един университет са N . За да осигури на завършващите студенти работа над реален софтуерен проект, ръководството на Програмата помолило софтуерни фирми да предложат примерни проекти, всеки от които да може да бъде изпълнен за определено време от C студента. От предложените проекти ръководството избрало R така, че $R.C \leq N$. Сега трябва да бъдат сформирани R колектива от по C студенти и това е главната грижа на ръководството. Нека за всеки студент, по време на обучението е определено цяло положително число K_i – **капацитет на студента**, представляващо комплексна оценка на качествата и подготовката му. Според експерти, най-важен за успеха на един проект е **индексът на несъвместимост** на колектива, измерващ се с разликата на най-високия и най-ниския капацитет в колектива (ако колективът се състои от един студент, тогава индексът е нула). Естествено е ръководството да се опита да сформира R -те колектива така, че максималният индекс на несъвместимост в получените се колективи да е колкото може по-малък. Напишете програма, която по зададен брой на студентите и техните капацитети пресмята търсения минимум за R колектива от по C студенти.

Вход: Програмата трябва да може да обработи няколко тестови примера при едно извикване. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. За всеки тестов пример, първият ред ще съдържа естествени числа N , R и C , а всеки от следващите N реда по едно цяло положително число – капацитетът на един от студентите.

Изход: За всеки тестов пример програмата трябва да изведе на отделен ред на **стандартния изход** едно цяло число – най-малката възможна стойност на максималния индекс на несъвместимост на образуваните колективи.

Ограничения: $1 \leq N \leq 100\,000$, $1 \leq R$, $1 \leq C$, $R.C \leq N$, $K_i \leq 1\,000\,000\,000$.

Пример:

Вход	Изход
1 8 2 3 170 205 225 190 260 130 225 160	30



Task C. TEAMS

Students from the last year of the bachelor program Informatics of a university are N . To provide the graduates work on real software project, the management of the program asked software companies to offer sample projects, each of which can be filled for some time from C students. From the proposed projects the management chose R in such way that $R.C \leq N$. Now R teams composed of C students have to be formed and this is a major concern of the management. Let, during the education, to each student is assigned a positive integer K_i – the **student's capacity**, representing a comprehensive assessment of her/his quality and preparation. According to experts, the most important for the success of a project is the **index of the incompatibility** of the team, measured by the difference of the highest and lowest capacity in the team (if the team consists of one student, then the index is zero). Naturally, the management tries to form R teams so that the maximum index of incompatibility for all teams to be as small as possible. Write a program that by given number of students and their capacities to find the possible minimum of the maximal index for R team from C students each.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case starts with a line with the numbers N , R and C . Each of the next N lines contains one positive integer – the capacity of one of the students.

Output: For each test case the program has to print on a separate line of the **standard output** one integer – the smallest possible value of the maximal incompatibility index of formed teams.

Restrictions: $1 \leq N \leq 100\,000$, $1 \leq R$, $1 \leq C$, $R.C \leq N$, $K_i \leq 1\,000\,000\,000$.

Example:

Input	Output
1 8 2 3 170 205 225 190 260 130 225 160	30



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача D. ВОЙНА

Мими и Иво играят на война с тесте от 52 карти – 4 бои по 13 карти във всяка, с нарастваща сила в реда 2, 3, 4, 5, 6, 7, 8, 9, 10 (Т), вале (J), дама (Q), поп (K) и асо (A). В началото на играта, на всеки играч се раздават някакви карти от тестето. Един ход се състои в следното – Мими **показва** най-горната карта от своето тесте, а Иво прави същото. Ако картите са с различна сила, играчът с по-силна карта ги **прибира**, като слага първо картата на Мими най-отдолу на тестето си, и после – картата на Иво под нея. Ако двете карти са с равна сила настъпва **война** – всеки играч показва трите най-горни карти в тестето си и се сравнява силата на третите карти. Ако тези карти са с различна сила, играчът с по-силна карта прибира първо картите на Мими в реда, в който тя ги е обърнала и след това картите на Иво в реда, в който той ги е обърнал. Ако силата на третите карти отново е равна се получава нова война и т.н. Ако някой от играчите няма три карти, които да отвори, войната завършва наравно и всеки прибира своите карти в реда, в който ги е отворил. При война, всички действия до определяне на резултата от нея се броят за един ход. Играта печели този, който прибере всички карти. Ако до 10 000 хода не е излъчен победител, играта завършва наравно. Напишете програма, която да позволи на Мими и Иво да определят победителя от едно раздаване, без да си губят времето да разиграват играта.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. На първия ред, за всеки тестов пример, ще бъде зададен броят N на картите, които Мими получава в началото, а на втория, разделени с по един интервал – картите на Мими, от най-горната към най-долната карта в тестето ѝ. Всяка карта се задава със стойността и боята си. Стойността е един от знаците 2, 3, 4, 5, 6, 7, 8, 9, Т, J, Q, K, A, а боята – някой от знаците C (трефа), D (каро), H (купа) или S (пика). На третия ред ще бъде зададен броят M на картите в тестето на Иво, а на четвъртия – картите от неговото тесте, в същия формат, като тестето на Мими.

Изход: За всеки тестов пример програмата трябва да изведе на отделен ред на **стандартния изход** низа Tie, ако след 10000 хода не е излъчен победител, низа Mimmy, ако Мими печели играта или низа Ivo, ако печели Иво.

Ограничения: $1 \leq N < 52$, $1 \leq M < 52$, $N + M \leq 52$.

Пример:

Вход	Изход
1 6 7C 9C 5H 4D 4H KS 6 2C 2D 5D 7S JS 3H	Mimmy



Task D. WAR

Mimmy and Ivo play **war** game with a deck of 52 cards – 4 suits, 13 cards in each, with increasing force in the order 2, 3, 4, 5, 6, 7, 8, 9, 10 (T), Jack (J), Queen (Q), King (K) and Ace (A). Early in the game, each player is dealt some cards from the deck. A move consists of the following – Mimmy **shows** the top card from her deck and Ivo does the same. If the cards are of different strength, the player with the high card **picks up** them by putting first the card of Mimi at the bottom of her/his deck, and then – the Ivo's card below it. If the two cards are of equal strength then a **war** occurs – each player shows the top three cards in her/his deck and the strength of the third cards is compared. If these cards are of different strength, the player with the strongest card pick up first the cards of Mimi in the order they was shown and then the Ivo's cards in the order they was shown too. If the strength of the third cards is equal again this is a new war, etc. If a player has less than three cards, the war ends in a draw and each player picks up her/his cards in the order in which they were opened. In war, all actions to determine the outcome of it are counted as one move. The game wins the player that succeeds to collect all cards in her/his deck. If after 10 000 moves there is no a winner, the game ends without a winner. Write a program that let Mimi and Ivo to determine the winner of a deal, without wasting time to play out the game.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case starts with a line that will contain the number N of cards that Mimmy has in the beginning, and line with her cards – ordered from top to bottom. Each card is described with its strength and its suit. The strength is denoted by one of the characters 2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A, and the suit – by one of the characters C (club), D (diamond), H (heart) or S (spade). On the third line the number M of the cards of Ivo will be given, and on the fourth – list of his cards in the same form as the cards of Mimmy.

Output: For each text case, on the **standard output**, the program has to print **Tie** if after 10000 moves no winner, **Mimmy** – if Mimmy wins the game or **Ivo** – when Ivo is the winner.

Restrictions: $1 \leq N < 52$, $1 \leq M < 52$, $N + M \leq 52$.

Example:

Input	Output
1 6 7C 9C 5H 4D 4H KS 6 2C 2D 5D 7S JS 3H	Mimmy



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача Е. КОРЕНОВО ДЪРВО

Всяко *дърво* (т.е. свързан граф без цикли) можем да превърнем в *кореново дърво*, като изберем един от върховете, да го означим с r , за *корен*. Добре известно е, че всеки два върха на дърво са свързани с единствен път и броя на ребрата по този път наричаме *разстояние* между двата върха. Ако върхът u е на разстояние d от r , върхът v е на разстояние $d + k$ от r , $k > 0$, и u е на пътя от r до v , тогава u се нарича *k -ти предшественик* на v , а v – *наследник* на u . Върховете, които нямат наследници, се наричат *листа* на кореновото дърво. Напишете програма, която по зададено кореново дърво изпълнява следните заявки:

- $0 \ x \ y$ – добавя в дървото нов лист x , като го свързва с ребро към y ;
- $1 \ x$ – премахва листа x от дървото;
- $2 \ x \ k$ – запитване за k -тия предшественик на x .

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Първият ред на всеки от тях ще съдържа броя N на върховете на зададеното кореново дърво и броя Q на заявките. Следват N реда с по две числа x и y , указващи че x е пряк наследник (дете) на y . Когато y е 0, това означава, че x е корен на дървото (т.е. няма баща). Следват Q реда, съдържащи по един от трите типа заявки, споменати по-горе.

Изход: За всяка заявка от тип 2, програмата трябва да изведе на **стандартния изход** k -тия предшественик на x . Ако такъв предшественик не съществува или в дървотоняма връх x , тогава програмата трябва да изведе 0.

Ограничения: $1 \leq N \leq 10^5$, $1 \leq Q \leq 10^5$, $1 \leq x \leq 10^5$, $0 \leq y < 10^5$, $1 \leq k \leq 10^5$.

Пример:

Вход	Изход
1	4
8 11	0
4 0	1
1 4	2
7 1	4
3 2	0
2 4	1
9 1	
6 2	
124 9	
2 124 3	
1 124	
2 124 3	
0 55 7	
0 8 55	
0 10 8	
2 10 4	
2 6 1	
2 3 2	
2 9 7	
2 8 3	



Task E. ROOTED TREE

Each **tree** (i.e. connected graph without circuits) could be transformed in a **rooted tree**, by selecting one of the vertices, to denote it by r , for **root**. It is well known that every two vertices of the tree are linked with a single path and the number of edges in this path is called a **distance** between the two vertices. If u is at a distance d from r , v is at a distance $d + k$ from r , $k > 0$, and u is on the path from r to v , then u is called a **k -th predecessor** of v , and v – a **successor** of u . The vertices that have no successors are called **leaves** of the rooted tree. Write a program that, for a given rooted tree performs the following queries:

- $0\ x\ y$ – inserts a new leaf x in the tree and links it with an edge to y ;
- $1\ x$ – deletes the leaf x from the tree;
- $2\ x\ k$ – asks for the k -th predecessor of x .

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. The first line of each of them will contain the number N of vertices of the rooted tree and the number Q of the queries. Then N lines follow with two numbers x and y , indicating that x is a direct successor (child) of y . When y is 0, this means that x is the root of the tree (0 is not a part of the tree). Each of the following Q rows contains one of the three types of queries mentioned above.

Output: For each query of type 2 the program must print to the **standard output** the k -th predecessor of x . If there is no such predecessor or the tree has not vertex x , then the program should print 0.

Restrictions: $1 \leq N \leq 10^5$, $1 \leq Q \leq 10^5$, $1 \leq x \leq 10^5$, $0 \leq y < 10^5$, $1 \leq k \leq 10^5$.

Example:

Input	Output
1	4
8 11	0
4 0	1
1 4	2
7 1	4
3 2	0
2 4	1
9 1	
6 2	
124 9	
2 124 3	
1 124	
2 124 3	
0 55 7	
0 8 55	
0 10 8	
2 10 4	
2 6 1	
2 3 2	
2 9 7	
2 8 3	



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача F. СНЯГ

Планинска област има N населени места, номерирани с целите числа от 1 до N . Двупосочни пътни отсечки свързват пряко M двойки от тях, така че от всяко населено място може да се стигне до всяко друго. Районът е прочут с това, че при зимни условия използването на пътищата не винаги е възможно. За да осигури колкото е възможно по-стабилен трафик през зимата, областният директор на Агенция „Пътища“ поръчал да се направи изследване на надеждността на отделните пътни отсечки. В резултат от наблюденията на пътищата, специалистите му представили списък, в който на всяка пътна отсечка било присвоено цяло число R между 10 и 900 включително – **надеждност на отсечката**, което означава, че след падане на обилен сняг се очаква в R от 1000 случая пътната отсечка да е проходима, а в останалите $1000 - R$ да не е. Това обаче не било достатъчно за шофьорите. Когато шофьор трябвало да пътува от населеното място A към населено място B , той искал да му посочат очаквания най-надежден път от A до B . Напишете програма, която по зададени надеждности на пътищата да определя най-надеждния път между две зададени населени места.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Всеки тестов пример започва с ред, на който ще бъдат зададени числата N и M . На всеки от следващите M реда ще бъдат зададени двете населени места, които поредната пътна отсечка свързва и нейната надеждност. На последния ред за теста ще бъдат зададени A и B , за които се търси най-надежден път.

Изход: За всеки тестов пример програмата трябва да изведе на един ред на **стандартния изход** редица от номера на населени места, започваща с A и завършваща с B – населените места по един най-надежден път от A до B .

Ограничения: $5 \leq N \leq 1000$.

Пример:

Вход	Изход
1	1 6 4
6 9	
1 2 200	
1 3 100	
1 6 250	
2 3 700	
3 6 200	
3 4 100	
4 6 900	
4 5 800	
5 6 600	
1 4	



Task F. SNOW

Highland has N villages numbered with integers from 1 to N . Bidirectional roads connect directly M pairs of them so that every village can get to each other. The area is famous for that in winter times road use is not always possible. To ensure as much as possible stable traffic in winter, the Regional Director of Agency "Roads" ordered to make a study of the reliability of the roads. As a result of the observations, experts submitted a report in which to each road it was assigned an integer R between 100 and 900 including – the **reliability** of the road segment, which means that after the fall of heavy snow it is expected the road to be usable in R out of 1 000 cases, but in the rest $1\,000 - R$ to be unusable. This information is not sufficient for drivers. When a driver has to travel from location A to location B , he wants to be informed for the most reliable path from A to B . Write a program that, given the reliability of roads, to determine the most reliable path between two specified places.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. First line of each test case will contain the integers N and M . Each of the next M lines will contain the two ends of one road and its reliability. The last line of the test will be set with the villages A and B , for which the most reliable path is asked.

Output: For each test case the program must print on a separate line of the **standard output** a list of villages beginning with A and ending with B lying on a most reliable path from A to B .

Restrictions: $5 \leq N \leq 1000$.

Example:

Input	Output
1 6 9 1 2 200 1 3 100 1 6 250 2 3 700 3 6 200 3 4 100 4 6 900 4 5 800 5 6 600 1 4	1 6 4



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача G. ОЛИМПИАДА

Всяка година домакинът на Републиканската студентска олимпиада по програмиране трябва да решава задача за разпределението на отборите, участващи в олимпиадата, в компютърните лаборатории на университета. За избягване на взаимодействието на отборите от един и същи университет, във всяка лаборатория трябва да има най-много един отбор от университет. Напишете програма, която да дава отговор на въпроса: има ли решение така поставената задача при условията на конкретна година?

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Всеки пример се състои от ред с броя M на университетите, ред със списък от M числа – броят на отборите на всеки университет, ред с броя N на компютърните лаборатории на университета-домакин и ред със списък от N числа – по колко отбора може да бъдат разпределени във всяка от лабораториите.

Изход: За всеки пример програмата трябва да изведе на отделен ред на **стандартния изход** отговора на поставената задача – yes или no.

Ограничения: Всички данни са цели положителни числа, не по-големи от 20.

Пример:

Вход	Изход
1 11 2 2 2 2 4 4 2 3 3 2 3 5 6 6 6 6 6	yes



Task G. OLYMPIAD

Each year, the Host University of the Bulgarian Collegiate Programming Contest must solve a problem of the distribution of teams participating in the Olympiad in the computer labs of the University. To avoid the interaction of m teams from same university, in each laboratory should have at most one team from the University. Write a program that answers the question: is there a solution of such task for the specific data for the year?

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case consists of line with the number M of universities, another line with a list of M integers – the number of teams in each university, line with the number N of computer labs of the Host University and a line with a list of N integers – how many teams can be assigned to a specific laboratory.

Output: For each test case the program has to print on a separate line of the **standard output** the answer to the question - yes or no.

Restrictions: All data are positive integers no larger than 20.

Example:

Input	Output
1 11 2 2 2 2 4 4 2 3 3 2 3 5 6 6 6 6 6	yes



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача Н. СУПЕРМАРКЕТ

Чичо Скрудж има голямо семейство: трима сина и девет внука. И всички те искат да ядат. Ето защо, чичо Скрудж веднъж седмично ходи в супермаркета. Веднъж, като отишъл в магазина, попаднал на промоция с мото „Всяка най-евтина K -та стока – безплатна!“. След като разучил правилата на промоцията чичо Скрудж разбрал следното: когато отиде на касата с избраните стоки, купувачът получава ваучер. Нека ваучерът съдържа N стоки. Тогава най-евтините $\lfloor N/K \rfloor$ от тях са безплатни ($\lfloor x \rfloor$ е долната цяла част на x). Например, ако във ваучера има 5 стоки за 200, 100, 1000, 400 и 100 пари и $K = 2$, то $\lfloor 5/2 \rfloor = 2$, тогава безплатни ще бъдат двете стоки по 100 пари и клиентът ще трябва да заплати общо 1600 пари.

Знаете, че чичо Скрудж е скъперник и преди да отиде на касата съобразил, че може да раздели избраните стоки на групи и за всяка да получи отделен ваучер така, че да изхарчи по-малко пари. С него, както обикновено, е и Жълтото пате, което помага на чичо си в покупките. Чичо Скрудж му възложил да раздели стоките в групи за да се възползват от промоцията по оптимален начин. Помогнете им, като напишете програма, която определя минималната сума, която може да се заплати за избраните стоки, ако се разделят в няколко групи.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. За всеки тестов пример, на първия ред са зададени две цели числа – броят N на стоките, които чичо Скрудж иска да купи и параметъра K на акцията „Всяка най-евтина K -та стока – безплатна!“. Следващият ред съдържа N цели числа, като i -тото от тях е цената A_i на i -тата стока, която ще купи чичо Скрудж.

Изход: За всеки пример програмата трябва да изведе на отделен ред на **стандартния изход** минималната сума, която чичо Скрудж трябва да плати за стоките.

Ограничения: $1 \leq N \leq 100\,000$, $2 < K \leq 100$, $1 \leq A_i \leq 10\,000$.

Пример:

Вход	Изход
1 5 2 200 100 1000 400 100	1300

Пояснение: В посочения пример чичо Скрудж може да раздели покупките на две групи: в едната ще отидат стоки за 1000 и за 400 пари и тогава стоката за 400 пари ще бъде безплатна, а в другата група ще бъдат останалите стоки и тогава една от стоките за 100 пари ще бъде безплатна и чичо Скрудж ще плати само 1300 пари.



Task H. SUPERMARKET

Uncle Scrooge has a large family: three sons and nine grandchildren. And they all want to eat. Therefore, Uncle Scrooge weekly walks to the supermarket. Once, as he went to the store, got a promotion with the motto "Each cheapest K -th commodity - free!". Discovering the rules of the promotion Uncle Scrooge understood: when goes to the cashier with chosen goods, the buyer gets a voucher. Let the voucher contains N commodities. Then the cheapest $\lfloor N/K \rfloor$ of them are free ($\lfloor x \rfloor$ is the floor of x). For example, if the voucher has 5 products for 200, 100, 1000, 400 and 100 money and $K = 2$, then $\lfloor 5 / 2 \rfloor = 2$ and the two products for 100 money each will be free and the client will have to pay a total of 1600 money.

You know that Uncle Scrooge is a miser and before going to the checkout misconstrued that can divide the selected items into groups and for each group to receive a separate voucher so that to spend less money. Being with him, as usual, Yellow Duck would like to help his uncle in purchases. Uncle Scrooge charged him to divide the goods into groups to take advantage of the promotion in an optimal way. Help them by writing a program that determines the minimum amount that can be paid for selected goods if divided into several groups.

Input: The program must be able to handle a few test cases. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case starts with line where two integers will be given – the number N of goods Uncle Scrooge wants to buy and the parameter K of the promotion "Each cheapest K -th commodity - free!". The next line will contain N integers, such as the i -th of them is the price A_i of the i -th commodity that will buy Uncle Scrooge.

Output: For each test case the program has to print on a separate line of the **standard output** the minimum amount Uncle Scrooge has to pay for the goods.

Restrictions: $1 \leq N \leq 100\,000$, $2 < K \leq 100$, $1 \leq A_i \leq 10\,000$.

Example:

Input	Output
1 5 2 200 100 1000 400 100	1300

Explanation: In the example Uncle Scrooge can divide purchases into two groups: one with goods for 1000 and 400 money and then good for 400 money will be free The other group will be of other goods and then one of the goods for 100 money will be free. So Uncle Scrooge will pay only 1300 money.



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача I. ПОРЯДЪК

Мими обича теорията на числата. Това е една от топ 10 на любимите ѝ дисциплини в математиката. Вчера, четейки предпочитания от нея учебник, отново среща едно от любимите си понятия – *мултипликативен порядък на число по модул n* (или просто *порядък*). Тя е възхитена от грациозността на това понятие и, след като ви припомни дефиницията му, иска да решите една задачка, свързана с него. И така: ако са дадени две взаимно прости числа $(a, n) = 1$, то показател на a по модул n наричаме най-малкото положително цяло число b , за което е изпълнено $a^b \equiv 1 \pmod{n}$. Т.е. остатъкът на a^b при деление на n е 1. Забележете, че това понятие е добре дефинирано – ако две числа a и n са взаимно прости, тогава винаги има поне едно b , за което $a^b \equiv 1 \pmod{n}$. Например, според теоремата на Ойлер, $a^{\varphi(n)} \equiv 1 \pmod{n}$, където $\varphi(n)$ е функцията на Ойлер за даденото число n – броят на естествените положителни числа по-малки от n и взаимно прости с него. За удобство ще считаме, че ако две числа a и n не са взаимно прости, то порядъкът на a по модул n също е дефиниран и е 0. Мими иска от вас да напишете програма, която по зададено число a , намира порядъците на a по модул всички числа в даден интервал.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Всеки тест се състои от един ред, на който ще са зададени три цели числа A , F и T – числото, чиито порядъци търсим, началото и края на интервала, от който се интересуваме, съответно.

Изход: За всеки пример програмата трябва да изведе на отделен ред на **стандартния изход** сумата на порядъците на A по модул всички числа в затворения интервал $[F, T]$.

Ограничения: $1 \leq A \leq 10^6$, $1 \leq F \leq 10^6$, $1 \leq T \leq 10^6$, $F \leq T$, $T - F \leq 10^5$.

Пример:

Вход	Изход
2	19
7 8 11	610202925
123012 3 100001	



Task I. ORDER

Mimmy likes the Number theory. This is one of the top 10 of her favorite subjects in mathematics. Yesterday, reading her favorite book, she saw again one of her favorite concepts – the **multiplicative order of a number modulo n** (or simply **order**). She is delighted by the gracefulness of this concept and after recalling its definition wants you to solve a task, connected with it. So, for given two mutually prime natural numbers $(a, n) = 1$, the order of a modulo n is the smallest positive integer b such that $a^b \equiv 1 \pmod{n}$, i.e. the remainder of a^b modulo n is 1. Note that this concept is well defined – if two numbers a and n are co-prime then always exists at least one b for which $a^b \equiv 1 \pmod{n}$. For example, according to the Euler's theorem, $a^{\varphi(n)} \equiv 1 \pmod{n}$, where $\varphi(n)$ is the function of Euler for the given number n – the number of natural positive integers less than n and co-prime to it. Furthermore, we will consider that if two numbers a and n are not mutually prime, the order of a modulo n is also defined and equal to 0. Mimmy asks you to write a program that, given the number a , to find the sum of all orders of a modulo all numbers in a given interval.

Input: The program must be able to handle a few examples, in one run. The first line of the **standard input** will contain the number T of the test cases. Each test consists of a line with three integers A , F and T – the number whose orders we look for, the beginning and the end of the interval we are interested, respectively.

Output: For each test case, the program has to print on a separate line of the **standard output** the sum of all orders of A modulo the numbers in the closed interval $[F, T]$.

Restrictions: $1 \leq A \leq 10^6$, $1 \leq F \leq 10^6$, $1 \leq T \leq 10^6$, $F \leq T$, $T - F \leq 10^5$.

Example:

Input	Output
2	19
7 8 11	610202925
123012 3 100001	

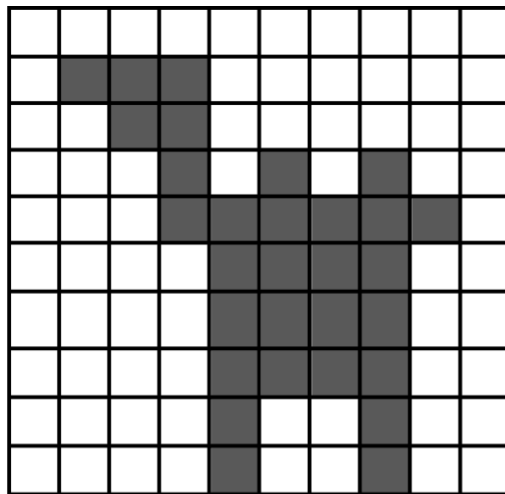


ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача J. ИЗОБРАЖЕНИЕ

Черно-бели изображения се представят като растер с размери $N \times N$ (виж фигурата). Напишете програма, която намира максималното лице (измерено в брой пиксели) на правоъгълник от изображението, съставен само от черни пиксели.

Вход: Програмата трябва да може да обработва няколко примера при едно изпълнение. На първия ред на **стандартния вход** ще бъде зададен броят T на тестовите примери. Всеки тестов пример започва с ред, на който ще е зададен размерът N на растера. Следват N реда с по един битов низ с дължина N на всеки от тях, представляващи самото изображение. Белите пиксели са кодирани с нули, а черните – с единици.



Изход: За всеки тестов пример програмата трябва да изведе на отделен ред на **стандартния изход** намереното лице на максимален черен правоъгълник.

Ограничения: $5 \leq N \leq 2048$.

Пример:

Вход	Изход
1 10 0000000000 0111000000 0011000000 0001010100 0001111110 0000111100 0000111100 0000111100 0000100100 0000100100	16



Task J. IMAGE

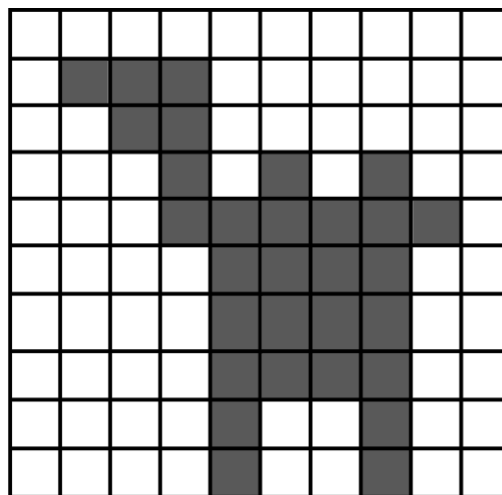
Black and white images are presented in a raster sized $N \times N$ pixels (see the figure). Write a program to find the maximum surface (measured with the number of pixels) of a rectangle from the image composed of only black pixels.

Input: The program must be able to handle a few test cases, in one run. The first line of the **standard input** will contain one integer T – the number of test cases. Each test case will start with a line which contains the size N of the raster. Then N lines follow with a bit string of length N on each of them representing the image. White pixels are encoded with zeros and black – with ones.

Output: For each test case the program should print on separate line of the **standard output** the surface of a maximum black rectangle.

Restrictions: $5 \leq N \leq 2048$.

Example:



Input	Output
1 10 0000000000 0111000000 0011000000 0001010100 0001111110 0000111100 0000111100 0000111100 0000100100 0000100100	16



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача К. ОВЦЕ

В едно стадо имало N овце. Една трета от тях оагнили по две агнета, половината – по едно агне, а останалите овце били ялови. Овчарят продал за Гергьовден половината от агнетата, а другата половина оставил в стадото – ако броят на агнетата се оказал нечетно число, овчарят канел приятели и изядяли едно агне на празника, за да може да се получат две равни половини. Напишете програма да определи колко овце и агнета са останали в стадото след Гергьовден?

Вход: Програмата трябва да може да обработва при едно изпълнение няколко тестови примера. Броят T на тестове ще бъде зададен на първия ред на **стандартния вход**. За всеки тестов пример на отделен ред е зададено цялото положително N , което се дели на 3 без остатък.

Изход: За всеки тестов пример програмата трябва да изведе на отделен ред на **стандартния изход** броя на овцете и агнетата в стадото след Гергьовден.

Ограничение: $5 < N < 1000$.

Пример:

Вход	Изход
2	95
60	237
150	

Обяснение: Във втория тестов пример $150/3 = 50$ овце са имали по 2 агнета и $150/2 = 75$ – по едно агне, общо $50 \cdot 2 + 75 = 175$ агнета. От тях едно било изядено от овчаря и приятелите му, $174/2 = 87$ са продадени, а 87 остават в стадото. Значи след Гергьовден в стадото има $150 + 87 = 237$ овце и агнета.



Task K. SHEEPS

A herd consists of N sheep. One third of them born two lambs each, a half - one lamb each, and others were barren. The shepherd sold a half of lambs for St. George day and a half left in the herd. If the new borne lambs was odd number, the shepherd invited friends and they ate one lamb for the holyday. How many sheep and lambs were in the herd after St. George day?

Input: The program must be able to process several test cases in one run. On the first line of the **standard input** the number T of the test cases will be given. For each test case, on a separate line, the number N of sheep in the herd will be given, where N is divisible by three.

Output: For each test on a separate line of the **standard output** the program has to print the number of sheep and lambs in the herd after St. George day.

Restrictions: $5 < N < 1000$.

Example:

Input	Output
2	95
60	237
150	

Explanation: In the second test case of the example each of $150/3 = 50$ sheep born 2 lambs and each of $150/2 = 75$ - one lamb, a total of $50 * 2 + 75 = 175$ lambs was born. One of them was eaten, $174/2 = 87$ was sold, and the other 87 remain in the herd. So, after St. George day the herd consists of $150 + 87 = 237$ sheep and lambs.



ЮГОЗАПАДЕН УНИВЕРСИТЕТ „СВ. НЕОФИТ РИЛСКИ“ БЛАГОЕВГРАД
XXVII РЕПУБЛИКАНСКА СТУДЕНТСКА ОЛИМПИАДА ПО
ПРОГРАМИРАНЕ, 8-10 май, 2015 г.

Задача L. ДЕЛИТЕЛИ

Да се напише програма, която намира:

- броя на простите числа в интервала $[a, b]$;
- за всяко число от същия интервал, което не е просто – неговия най-малък прост делител.

Вход: Програмата трябва да може да обработва при едно изпълнение няколко тестови примера, които са зададени в **двоичен (binary) файл**, който ще бъде зададен като **стандартен вход**. Всеки тестов пример се състои от двете 32-битови положителни цели числа, които са краища на интервала – първо във файла е записан левият край, а после – десният.

Изход: За всеки тестов пример програмата трябва да изведе на **стандартния изход**, първо, ред с текста `test case #<t>`: (спазвайте стриктно форматирането от примера), където `<t>` е номерът на текущия тестов пример, а след това ред с текста `<p> primes in [<a>, <b>]` ., където `<a>` и `<b>` са двата края на интервала, а `<p>` е броят на простите числа в него. Всеки от следващите редове в изхода за тестовия пример е във вида `<q> <k>` и означава, че простото число `<q>` е най-малък прост делител на `<k>` непрости числа в интервала. Двойките трябва да бъдат сортирани във възходящ ред на `<q>`. Между резултатите от всеки два поредни тестови примера трябва да има по един празен ред.

Ограничения: $2 \leq a \leq 10^7$, $2 \leq b \leq 10^7$.

Пример:

Вход	Изход
2 10 17 17 9 9 20 200	test case #1: 4 primes in [2, 10]. 2 4 3 1 test case #2: 1 primes in [17, 17]. test case #3: 0 primes in [9, 9]. 3 1 test case #4: 38 primes in [20, 200]. 2 91 3 30 5 12 7 6 11 3 13 1

Забележка: По понятни причини входните данни в примера са показани в текстов, а не в двоичен вид, както ще бъдат подадени на стандартния вход.



Task L. DIVISORS

Write a program to find:

- the number of primes in the interval $[a, b]$;
- the smallest prime divisor for each number from the same interval, which is not prime.

Input: The program must be able to process in one run several test cases that are set in **binary file** which will be given as **standard input**. Each test case consists of two 32-bit positive integers that are the ends of the interval – the left end is saved first in the file, and then – the right end.

Output: For each test case, the program must write to the **standard output**, first, a line with the text `test case #<t>:` (strictly follow the format of the example), where $\langle t \rangle$ is the number of the current test case; on the second line – the text `<p> primes in [ $\langle a \rangle$ ,  $\langle b \rangle$ ].`, where $\langle a \rangle$ and $\langle b \rangle$ are the two ends of the interval, and $\langle p \rangle$ – the number of prime numbers in it. Each of the following lines in the output for the test case should be in the form `<q> <k>` and means that the prime number $\langle q \rangle$ is the smallest prime divisor of $\langle k \rangle$ not prime numbers in the interval. Couples must be sorted in ascending order of $\langle q \rangle$. Between each two consecutive test cases a blank line should be printed.

Restrictions: $2 \leq a \leq 10^7$, $2 \leq b \leq 10^7$.

Example:

Input	Output
2 10 17 17 9 9 20 200	test case #1: 4 primes in [2, 10]. 2 4 3 1 test case #2: 1 primes in [17, 17]. test case #3: 0 primes in [9, 9]. 3 1 test case #4: 38 primes in [20, 200]. 2 91 3 30 5 12 7 6 11 3 13 1

Note: For obvious reasons, the input data in the example is shown in text form, not binary, as will be submitted on standard input.