Задача A. Checkpoint

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

You are racing on a 2D lattice grid starting from the origin $(0, 0)$ towards a goal (M, N) where M and N are positive integers such that $0 < M \leq N$. There is a checkpoint that's neither on the origin nor on the goal with coordinates (m, n) such that $0 \leq m \leq M$ and $0 \leq n \leq N$. You must clear the checkpoint before you reach the goal. The shortest path takes $T = M + N$ steps. At each point, you can move to the four immediate neighbors at a fixed speed, but since you don't want to lose the race, you are only going to take either a step to the right or to the top.

Even though there are many ways to reach the goal while clearing the checkpoint, the race is completely pointless since it is relatively easy to figure out the shortest route. To make the race more interesting, we change the rules. Instead of racing to the same goal (M, N), the racers get to pick a goal (x, y) and place the checkpoint to their liking so that there are exactly S distinct shortest paths.

For example, given $S = 4$, consider the following two goal and checkpoint configurations Placing the checkpoint at $(1, 3)$ and the goal at $(2, 3)$. There are 4 ways to get from the origin to the checkpoint depending on when you move to the right. Once you are at the checkpoint, there is only one way to reach the goal with minimal number of steps. This gives a total of 4 distinct shortest paths, and takes $T = 2 + 3 = 5$ steps. However, you can do better. Placing the checkpoint at $(1, 1)$ and the goal at $(2, 2)$. There are two ways to get from the origin to the checkpoint depending on whether you move to the right first or later. Similarly, there are two ways to get to the goal, which gives a total of 4 distinct shortest paths. This time, you only need $T = 2 + 2 = 4$ steps. As a Hacker Cup racer, you want to figure out how to place the checkpoint and the goal so that you cannot possibly lose. Given S, find the smallest possible T, over all possible checkpoint and goal configurations, such that there are exactly S distinct shortest paths clearing the checkpoint.

Формат входного файла

As input for the race you will receive a text file containing an integer $5 \leq R \leq 20$, the number of races you will participate in. This will be followed by R lines, each describing a race by a single number $1 \leq S \leq 10000000$.

Формат выходного файла

Your submission should contain the smallest possible length of the shortest path, T for each corresponding race, one per line and in order.

Пример

stdin	stdout
5 4 5 12 14 1	Case #1: 4 Case #2: 6 Case #3: 6 Case #4: 9 Case #5: 2

Задача B. Recover the Sequence

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

Merge sort is one of the classic sorting algorithms. It divides the input array into two halves, recursively sorts each half, then merges the two sorted halves.

In this problem merge sort is used to sort an array of integers in ascending order. The exact behavior is given by the following pseudo-code:

```
function merge_sort(arr):
    n = arr.length()
    if n <= 1:
        return arr

    // arr is indexed 0 through n-1, inclusive
    mid = floor(n/2)

    first_half = merge_sort(arr[0..mid-1])
    second_half = merge_sort(arr[mid..n-1])
    return merge(first_half, second_half)

function merge(arr1, arr2):
    result = []
    while arr1.length() > 0 and arr2.length() > 0:
        if arr1[0] < arr2[0]:
            print '1' // for debugging
            result.append(arr1[0])
            arr1.remove_first()
        else:
            print '2' // for debugging
            result.append(arr2[0])
            arr2.remove_first()

    result.append(arr1)
    result.append(arr2)
    return result
```

A very important permutation of the integers 1 through N was lost to a hard drive failure. Luckily, that sequence had been sorted by the above algorithm and the debug sequence of $1s$ and $2s$ was recorded on a different disk. You will be given the length N of the original sequence, and the debug sequence. Recover the original sequence of integers.

In the first example, N is 2 and the debug sequence is 1. The original sequence was 12 or 21. The debug sequence tells us that the first number was smaller than the second so we know the sequence was 12. The checksum is 994.

In the second example, N is 2 and the debug sequence is 2. This time the original sequence is 21.

In the third example, N is 4 and the debug sequence is 12212. The original sequence is 2431.

Формат входного файла

The first line of the input file contains an integer $5 \leq T \leq 20$. This is followed by T test cases, each of which has two lines. The first line of each test case contains the length of the original sequence,

$2 \leq N \leq 10000$. The second line contains a string of $1s$ and $2s$, the debug sequence produced by merge sort while sorting the original sequence.

Формат выходного файла

To avoid having to upload the entire original sequence, output an integer checksum of the original sequence, calculated by the following algorithm:

```
function checksum(arr):
    result = 1
    for i=0 to arr.length()-1:
        result = (31 * result + arr[i]) mod 1000003
    return result
```

Пример

stdin	stdout
5 2 1 2 2 4 12212 5 1122211 10 12122121211112212121 2	Case #1: 994 Case #2: 1024 Case #3: 987041 Case #4: 570316 Case #5: 940812

Задача C. Squished Status

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

Some engineers got tired of dealing with all the different ways of encoding status messages, so they decided to invent their own. In their new scheme, an encoded status message consists of a sequence of integers representing the characters in the message, separated by spaces. Each integer is between 1 and M, inclusive. The integers do not have leading zeroes. Unfortunately they decided to compress the encoded status messages by removing all the spaces!

Your task is to figure out how many different encoded status messages a given compressed status message could have originally been. Because this number can be very large, you should return the answer modulo 4207849484 (0xfaceb00c in hex).

For example, if the compressed status message is "12"it might have originally been "1 2 or it might have originally been "12". The compressed status messages are between 1 and 1000 characters long, inclusive. Due to database corruption, a compressed status may contain sequences of digits that could not result from removing the spaces in an encoded status message.

Формат входного файла

The input begins with a single integer, $5 \leq N \leq 25$, the number of compressed status messages you must analyze. This will be followed by N compressed status messages, each consisting of an integer $2 \leq M \leq 255$, the highest character code for that database, then the compressed status message, which will be a string of digits each in the range '0' to '9', inclusive. All tokens in the input will be separated by some whitespace. $1 \leq$ length of encoded status ≤ 1000.

Формат выходного файла

For each of the test cases numbered in order from 1 to N, output "Case #i: "followed by a single integer containing the number of different encoded status messages that could be represented by the corresponding compressed sequence modulo 4207849484. If none are possible, output a 0.

Пример

stdin	stdout
5 12 12 255 219 30 1234321 2 101 70 8675309	Case #1: 2 Case #2: 4 Case #3: 6 Case #4: 0 Case #5: 2

Задача D. Monopoly

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

In a certain business sector there are currently N small companies, each having just a single employee. These employees are numbered 1 through N.

The business sector is about to be transformed into a monopoly. This will happen through a series of mergers, until there is only one company. A single merger involves two companies. In a merger, the president of one company becomes the direct report of the president of the other company, preserving the rest of the hierarchies of both companies.

You will be given the descriptions of all mergers. Depending on how they are performed (which of the two presidents involved becomes the president of the new company), the hierarchy can of the final company can take different shapes. We want the hierarchy of the final company to be as shallow as possible. The task is to find the smallest possible number of levels in the final hierarchy.

There is also a limit D on the number of direct reports any employee can have. Because of this limit, there may be only one way to accomplish a certain merger, or it might even be impossible. However, there will always be some way to accomplish all the mergers.

In the first example, we have $N = 3$ and $D = 2$. The first merger happens between the companies of employees 1 and 2. In the resulting company we can have employee 1 as the president with 2 as his report, or vice versa. Next this company merges with the company of employee 3. If we have employee 3 become the president, the hierarchy will be a chain 3-1-2 or 3-2-1. If 1 or 2 become the president, that president will have the other two employees as direct reports. This last hierarchy has two levels.

Формат входного файла

The first line contains the number of test cases $5 \leq T \leq 20$.

Each test case starts with a blank line. The next line contains two integers, $2 \leq N \leq 30000$ and $1 \leq D \leq 5000$.

Each of the following $N - 1$ lines describes a single merger, with two integers between 1 and N. These are the employees whose companies are merging. The two employees will never already be part of the same company.

The mergers must be performed in the order in which they are given.

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #i: "followed by a single integer, the smallest number of levels in the final hierarchy.

Пример

stdin	stdout
5 3 2 1 2 1 3 3 1 1 2 1 3 4 2 4 2 2 1 3 2 5 2 5 2 4 5 2 1 3 1 6 3 6 1 4 2 1 4 5 6 3 6	Case #1: 2 Case #2: 3 Case #3: 3 Case #4: 3 Case #5: 4

Задача E. Road removal

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

You are given a network with N cities and M bidirectional roads connecting these cities. The first K cities are important. You need to remove the minimum number of roads such that in the remaining network there are no cycles that contain important cities. A cycle is a sequence of at least three different cities such that each pair of neighboring cities are connected by a road and the first and the last city in the sequence are also connected by a road.

In the first example, we have $N = 5$ cities that are connected by $M = 7$ roads and the cities 0 and 1 are important. We can remove two roads connecting (0, 1) and (1, 2) and the remaining network will not contain cycles with important cities. Note that in the remaining network there is a cycle that contains only non-important cities, and that there are also multiple ways to remove two roads and satisfy all conditions. One cannot remove only one road and destroy all cycles that contain important cities.

Формат входного файла

The first line contains the number of test cases $1 \leq T \leq 20$.

Each case begins with a line containing integers $1 \leq N \leq 10000, 1 \leq M \leq 50000$ and $1 \leq K \leq N$, which represent the number of cities, the number of roads and the number of important cities, respectively. The cities are numbered from 0 to $N - 1$, and the important cities are numbered from 0 to $K - 1$. The following M lines contain two integers $a[i]$ and $b[i], 0 \leq i < M$, that represent two different cities connected by a road.

It is guaranteed that $0 \leq a[i], b[i] < N$ and $a[i] \neq b[i]$. There will be at most one road between two cities.

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #i: "followed by a single integer, the minimum number of roads that need to be removed such that there are no cycles that contain an important city.

Пример

stdin	stdout
You can download it here http://pastie.org/5917165	Case #1: 2 Case #2: 4 Case #3: 72 Case #4: 33 Case #5: 361

Задача F. Sequence Slicing

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

Let S be a sequence of N natural numbers. We can define an infinite sequence MS in the following way: $MS[k] = S[kmodN] + N * floor(k/N)$. Where k is a zero based index.

For example if the sequence S is $2, 1, 3$ then MS would be $2, 1, 3, 5, 4, 6, 8, 7, 9, 11, 10, 12...$

Now consider a subsequence of MS generated by picking two random indices a, b from the range $[0..R]$ inclusive, and taking all the elements between them, that is: $MS[min(a, b)..max(a, b)]$.

If we use the same MS as in the example above and $a = 2, b = 5$ then our subsequence would be $3, 5, 4, 6$.

Your task is to calculate the probability that the selected subsequence has at least K distinct elements. a and b are selected independently and with a uniform distribution. The result should be printed as a fraction. See the "Output"section for clarification.

In the first example there are 36 different subsequences to consider. 6 of them have only a single number, and the remaining 30 have at least 2 different numbers, so the answer is $5/6$.

The second example is similar, but now the sequence looks like $2, 1, 5, 5, 4, 8$. There are 8 subsequences with less than 2 distinct numbers: the six single number subsequences plus $(a = 2, b = 3)$ and $(a = 3, b = 2)$ which both result in $5, 5$. That gives a probability of $(36 - 8)/36 = 7/9$.

The third example uses the same sequence as the second example, but now we want to have subsequences with at least 4 different numbers. All pairs of indices that have this property are: $(0, 4), (0, 5), (1, 5), (4, 0), (5, 0)$, and $(5, 1)$. Six out of thirty six results in a probability of $1/6$.

Формат входного файла

The first line of the input file contains an integer $1 \le T \le 20$. This is followed by T test cases, each of which has two lines.

The first line of each test case contains three integers separated by spaces, $1 \le N \le 2000$, $1 \le K \le 1000000000$, and $1 \le R \le 1000000000$, $K \le R$. The second line contains N space separated integers, $S[0]$ through $S[N - 1]$, $1 \le S[i] \le 100000$.

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #i: "followed by the probability that the selected subsequence of MS has at least K distinct elements. The probability should be expressed as a fraction p/q, where p and q represent the numerator and denominator respectively and are relatively prime (that is they share no common positive divisors except 1).

If the probability is 0 or 1 output $0/1$ or $1/1$ respectively.

Пример

stdin	stdout
5 3 2 5 2 1 3 3 2 5 2 1 5 3 4 5 2 1 5 3 5 5 2 2 5 10 1000 1000000 1 2 3 4 5 6 7 8 9 10	Case #1: 5/6 Case #2: 7/9 Case #3: 1/6 Case #4: 0/1 Case #5: 998005995006/1000002000001

Задача G. Divisor Function Optimization

Имя входного файла:	stdin
Имя выходного файла:	stdout
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

Let $d(N)$ be the number of positive divisors of positive integer N. Consider the infinite sequence $x(n) = d(n)^a/n^b, n = 1, 2, 3, \ldots$ where a and b are fixed positive integers. It can be shown that this sequence tends to zero. Hence it attains its maximum. Denote it by p/q where p and q are co-prime positive integers. Your task is for given a and b find p and q modulo $M = 10^9 + 7$. But to keep input and output small you will be given tuples $(b1; b2; a1; a2; c)$ and need to calculate the sum of $(p mod M)$ for all pairs $(a; b)$ such that $b1 \leq b \leq b2, a1 \leq a \leq a2$ and $a \leq c * b$, and the same sum for q-values.

Формат входного файла

The first line contains a positive integer $1 \leq T \leq 20$, the number of test cases. T test cases follow. The only line of each test case contains five space separated positive integers $b1, b2, a1, a2$ and c. $1 \leq b1 \leq b2 \leq 10000$, $1 \leq a1 \leq a2 \leq 250000$, $1 \leq c \leq 25$. In each testcase the total number of pairs $(a; b)$ for which the answer should be calculated does not exceed 100000.

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #i: "followed by a space separated pair of integers: the sum of $(p mod M)$ for all pairs $(a; b)$ mentioned above and the sum of $(q mod M)$ for all such pairs. Note that you need to find the sum of residues not the residue of sum (see testcase 3 as a reference).

Пример

stdin	stdout
5 1 1 1 3 10 1 2 1 88 3 1 10 1 50 5 1 1 1 15 15 30 33 1 29 25	Case #1: 1540 37 Case #2: 2377233 1491 Case #3: 75232917907 54682660016 Case #4: 5956707232 7441063226 Case #5: 116 116

Задача H. Trapezoids

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

Consider two horizontal lines and a set of N trapezoids. A trapezoid $T[i]$ between these lines has two vertices situated on the upper line and the other two vertices on the lower line. We will denote by $a[i], b[i], c[i]$ and $d[i]$ the upper left, upper right, lower left and respectively lower right vertices of the trapezoid $T[i]$. Assume that no two trapezoids share a common vertex (meaning that all $a[i]$ and $b[i]$ coordinates on the upper line are different, and the same holds for the bottom line and coordinates $c[i]$ and $d[i]$). A trapezoid set is called disconnected if one can separate the trapezoids in two or more groups such that no two trapezoids from different groups intersect. Determine the smallest number of trapezoids to remove, such that the remaining trapezoids form a disconnected set. If the solution does not exist, output -1.

In the first example, one can remove trapezoids 5 and 6 leaving two disconnected sets with trapezoids (1, 2, 3, 4) and (7, 8).

In the second example, the coordinates of ten trapezoids are

```
1000000 4000000 3000000 5000000
7000001 8000001 2000001 6000001
2000002 7000002 2000002 5000002
5000003 7000003 2000003 6000003
4 6000004 2000004 3000004
4000005 5000005 3000005 8000005
-999994 6 6 2000006
4000007 6000007 1000007 7000007
-999992 8 -999992 2000008
5000009 6000009 9 7000009
```

Формат входного файла

The first line of the input file contains an integer T, and this is followed by T test cases. Each test case is given in the compressed format.

The first line of each test case contains the number of trapezoids N, an integer K, and integer parameters X, A, B, M, p, q. In the next K lines are given integer numbers $aa[i], bb[i], cc[i], dd[i]$. The following code is used for generating the next random number using linear congruential generator with the starting value X and parameters A, B and modulo M:

```
long long prior = X;
long long next() {
  prior = (A * prior + B) % M;
  return prior;
}
```

The following code is used for extending the auxiliary sequences aa, bb, cc and dd:

```
for (int i = K; i < N; i++) {
    aa[i] = aa[i - K] + next() % (2 * p) - p;
    bb[i] = aa[i] + 1 + next() % (2 * (bb[i % K] - aa[i % K]));
    cc[i] = cc[i - K] + next() % (2 * q) - q;
    dd[i] = cc[i] + 1 + next() % (2 * (dd[i % K] - cc[i % K]));
}
```

The final coordinates of the trapezoids are given by:

```
for (int i = 0; i < N; i++) {
    a[i] = aa[i] * 1000000 + i;
    b[i] = bb[i] * 1000000 + i;
    c[i] = cc[i] * 1000000 + i;
    d[i] = dd[i] * 1000000 + i;
}
```

Note that above code generates trapezoids that satisfy all conditions of the problem, and '

$1 \leq T \leq 20$ $1 \leq N \leq 300000$ $1 \leq K \leq N$ $0 \leq X, A, B \leq 2000000000$ $-2000000000 \leq aa[i], bb[i], cc[i], dd[i] \leq 2000000000$ $1 \leq p, q, M \leq 2000000000$

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #i: "followed by a single integer, the minimum number of trapezoids that need to be removed such that the remaining trapezoids form a disconnected set.

Пример

stdin	stdout
5 8 8 1 1 1 1 1 1 1 3 1 3 2 5 6 10 4 8 5 8 6 9 2 4 7 11 11 13 10 13 7 9 12 16 14 15 14 15 12 16 10 2 2 1 1 7 2 2 1 4 3 5 7 8 2 6 100000 2 1 1 1 2 1 1 -2 -1 -1000001 -1000000 -1000001 -1000000 -2 -1 200 3 5 1 2 1117 11 13 1 10 1 10 2 11 2 11 3 12 3 12 10000 1 0 0 1 2 1 1 1 10000 1 100000	Case #1: 2 Case #2: 6 Case #3: 50002 Case #4: 61 Case #5: -1

Задача I. Unfriending

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	100 seconds
Ограничение по памяти:	1024 Mb

It's time to clean up your friends list after friending too many people on Facebook over the winter. The set of your friends' user ids is $F = x_0, x_1, ..., x_{n-1}$. Each friend from F is on zero or more friend lists that you use to organize your friends. There are M friend lists, $c_0, c_1, ..., c_{m-1}$. You can unfriend (remove from F) some of your friends, but because you want to stay in touch with people, you can only unfriend one person from each friend list. You may also choose not to unfriend anyone from friend list. Any friend that isn't on a friend list cannot be unfriended.

Your goal is to find the maximum possible distance between the two friends whose user ids are the closest after you are done unfriending people. The distance between user id x and user id y is defined as $abs(x - y)$.

Формат входного файла

The first line contains a positive integer T, the number of test cases. T test cases follow.

The first line of each test case contains the number of friends N, and number of friend lists M, separated by a space. The second line contains $x0$ and integer parameters a, b, and p, separated by spaces. You must use these to generate the remaining numbers $x_1, \ldots, x_{n-1}$ according to this formula: $x_i = (x_{i-1} * a + b) \bmod p$

The next M lines of each test case define the friend lists for that test case. Each line consists of the following integers, separated by spaces: $size, y_0, a, b$. The size is the number of friends in the friend list. y_0 is the index in F of the first friend in the list. You must generate the remaining friends in the friend list $y_1, \ldots, y_{n-1}$ according to the following formula: $y_i = (y_{i-1} * a + b) \bmod n$

If a friend list contains more than one of a given index, consider it to only contain that index once.

$1 \leq T \leq 20$, $2 \leq n \leq 50000$, $0 \leq m \leq 1500$, $0 \leq$ sum of sizes of all friend lists ≤ 1000000

Numbers used in generators: $0 < a, p < 230$, $0 \leq b < 230$

Формат выходного файла

For each of the test cases numbered in order from 1 to T, output "Case #followed by the case number, followed by ": followed by the maximum possible distance between the two closest user ids that you are still friends with for that case.

Пример

stdin	stdout
5 9 2 48071530 715583197 479567108 1050406 2 6 743132951 85415827 2 3 376850600 968665455 11 3 787260730 352556659 382266931 1057730 2 9 734333632 693274928 4 2 109921566 4305285 2 6 677574083 984173544 10 3 395307869 893188066 853669149 1056195 4 7 225271526 816658669 4 0 131869161 339696459 4 7 709635520 662395613 12 3 796525811 590453825 261103276 1050505 3 9 594759110 714784928 3 5 431449006 621243801 3 10 638845242 279357772 12 2 609821186 802201942 33450613 1048819 4 3 988115593 116841583 4 0 327812052 590896542	Case #1: 7392 Case #2: 5130 Case #3: 15318 Case #4: 63175 Case #5: 5814