Data Structures, Spring 2013
IIIT Hyderabad
Programming Assignment 4

1. Online Median

Let us define :
F[1] = 1
F[i] = (a*M[i] + b*i + c)%1000000007 for i > 1

where M[i] is the median of the array {F[1],F[2],..,F[i-1]}.
The median of an array is the middle element of that array when it is sorted. If there are an even number of elements in the array, we choose the first of the middle two elements to be the median.

Given a,b,c and n, calculate the sum F[1] + F[2] + .. + F[n].

Input Format
The first line contains T the number of test cases. Each of the next T lines contain 4 integers : a,b,c and n.

Output Format
Output T lines, one for each test case, containing the required sum.

Sample Input
2
1 0 0 3
3 1 2 6

Sample Output
3
103

Constraints
1 <= T <= 100
0 <= a,b,c < 1000000007
1 <= n <= 200000

---

2. Shortest Path.

Given are 2 vertices, S and T, in a weighted graph. Find the shortest path between S and T.

Input
First line contains 2 integers N and M. N is the number of vertices and M is the number of edges.

The next M lines contain 3 integers each: U, V and W. This means that there is an edge between vertex u and vertex v with weight w.

Then follows 1 more line of input containing two integers S and T.

Output
Print the shortest distance between S and T. Print -1 if there is no path between vertex S and vertex T.

Constraints
1<= N<= 10000
0<= M<= [N(N-1)]/2
1<= U,V <=N
U !=V
W>0
1<=S,T<=N

---

3. Help the Soldier

Igor, a famous russian soldier, must go to war in Afghanistan (we are in late 80's). His superiors allowed him to buy himself his equipment. So, he must buy 6 items: helmet, bulletproof vest, trousers, boots, tunic and a firearm. This items are represented with numbers from 1 to 6. There are N(6 < N < 101) items of this 6 types. Each item is characterized by its price p[i] (in rublas)
and is quality q[i]. Igor has T (0 < T < 1001) rublas and he wants to maximize the total quality of his equipment. The total quality is the quality of the item with the lowest quality. Help him.

Input Format
On the first line there are two integers N and T. On the lines 2 . . . N+1 there are 3 integers, type[i] (from 1 to 6) p[i] and q[i]. (0 < p[i], q[i] < T)

Output Format
Output the total quality.

Sample Input
7 53
5 8 2
2 4 8
6 8 13
1 13 12
4 5 1
3 2 7
3 13 5
Sample Output:
1

---

4. Ordering Tasks

John has n tasks to do. Unfortunately, the tasks are not independent and the execution of one task is only possible if other tasks have already been executed.

Input
The input will consist of several instances of the problem. Each instance begins with a line containing two integers, **1 <= n <= 100** and **m**.**n** is the number of tasks (numbered from **1** to **n**) and **m** is the number of direct precedence relations between tasks. After this, there will be **m** lines with two integers **i** and **j**, representing the fact that task **i** must be executed before task **j**. An instance with **n = m = 0** will finish the input.

Output
For each instance, print a line with n integers representing the tasks in a possible order of execution.

Sample Input

```
5 4
1 2
2 3
1 3
1 5
0 0
```

Sample Output

```
1 4 2 5 3
```

---

5. Frogger

Freddy Frog is sitting on a stone in the middle of a lake. Suddenly he notices Fiona Frog who is sitting on another stone. He plans to visit her, but since the water is dirty and full of tourists' sunscreen, he wants to avoid swimming and instead reach her by jumping.
Unfortunately Fiona's stone is out of his jump range. Therefore Freddy considers to use other stones as intermediate stops and reach her by a sequence of several small jumps.
To execute a given sequence of jumps, a frog's jump range obviously must be at least as long as the longest jump occurring in the sequence.
The *frog distance* (humans also call it *minimax distance*) between two stones therefore is defined as the minimum necessary jump range over all possible paths between the two stones.

You are given the coordinates of Freddy's stone, Fiona's stone and all other stones in the lake. Your job is to compute the frog distance between Freddy's and Fiona's stone.

Input Specification

The input file will contain one or more test cases. The first line of each test case will contain the number of stones *n* (2<= n <=200). The next *n* lines each contain two integers x_i, y_i (0<= x_i , y_i <= 1000) representing the coordinates of stone #*i*. Stone #1 is Freddy's stone, stone #2 is Fiona's stone, the other *n*-2 stones are unoccupied. There's a blank line following each test case. Input is terminated by a value of zero (0) for *n*.

Output Specification

For each test case, print a line saying ``Scenario #*x*" and a line saying ``Frog Distance = *y*" where *x* is replaced by the test case number (they are numbered from 1) and *y* is replaced by the appropriate real number, printed to three decimals. Put a blank line after each test case, even after the last one.

Sample Input

```
2
0 0
3 4

3
17 4
19 4
18 5

0
```

Sample Output

```
Scenario #1
Frog Distance = 5.000

Scenario #2
Frog Distance = 1.414
```

---

6. Prime Path

The ministers of the cabinet were quite upset by the message from the Chief of Security stating that they would all have to change the four-digit room numbers on their offices.

- It is a matter of security to change such things every now and then, to keep the enemy in the dark.
- But look, I have chosen my number 1033 for good reasons. I am the Prime minister, you know!
- I know, so therefore your new number 8179 is also a prime. You will just have to paste four new digits over the four old ones on your office door.
- No,it’s not that simple. Suppose that I change the first digit to an 8, then the number will read 8033 which is not a prime!

- I see, being the prime minister you cannot stand having a non-prime number on your door even for a few seconds.
- Correct! So I must invent a scheme for going from 1033 to 8179 by a path of prime numbers where only one digit is changed from one prime to the next prime. Now, the minister of finance, who had been eavesdropping, intervened.
- No unnecessary expenditure, please! I happen to know that the price of a digit is one pound.
- Hmm, in that case I need a computer program to minimize the cost. You don't know some very cheap software gurus, do you?
- In fact, I do. You see, there is this programming contest going on

Help the prime minister to find the cheapest prime path between any two given four-digit primes! The first digit must be nonzero, of course. Here is a solution in the case above.
1033
1733
3733
3739
3779
8779
8179
The cost of this solution is 6 pounds. Note that the digit 1 which got pasted over in step 2 can not be reused in the last step a new 1 must be purchased.

Input Format
One line with a positive number: the number of test cases (at most 100). Then for each test case, one line with two numbers separated by a blank. Both numbers are four-digit primes (without leading zeros).

Output Format
One line for each case, either with a number stating the minimal cost or containing the word Impossible.

Sample Input
3
1033 8179
1373 8017
1033 1033

Sample Output
6
7
0

---

7. Promotion

A large Bytelandian supermarket chain has asked you to write a program for the simulating costs of a promotion being prepared. The promotion has to follow the following rules:

- A customer who wants to participate in the promotion, writes on the receipt, paid by himself, his personal details and throws it into a special ballot box.
- At the end of every day of the promotion, two bills are taken out from the ballot box:
  – first, the receipt amounting to the largest sum is chosen,
  – then the receipt amounting to the smallest sum is chosen;

  The customer who has paid the largest sum gets a money prize equal to the difference between the sum on his bill and the sum on the bill amounting to the smallest sum.
- To avoid multiple prizes for one purchase, both bills selected according to the above rules are not returned to the ballot box, but all remaining bills still participate in the promotion.

The turnover of the supermarket is very big, thus an assumption can be made, that at the end of every day, before taking out receipts amounting to the largest and the smallest sum, there are at least 2 receipts in the ballot box.
Your task is to compute (on the basis of information about prices on receipts thrown into the ballot box on each day of promotion) what the total cost of prizes during the whole promotion will be.
Write a program, which: reads from the standard input a list of prices on receipts thrown into the ballot box on each day of the promotion, computes the total cost of prizes paid in consecutive days of promotion, then writes the result to the standard output.

Input Format
The first line of the input contains one positive integer n ($1 \leq n \leq 5000$), which is the duration of promotion in days. Each of the next n lines consists of a sequence of non-negative integers separated by single spaces. Numbers in the (i+1)-th line of the file represent prices on receipts thrown into the ballot box on the i-th day of promotion. The first integer in the line is k, $0 \leq k \leq 105$, the number of receipts on the day, and the next k numbers are positive integers standing for the sums on receipts; none of these numbers is larger than 106.

Output Format
The output should contain exactly one integer, equal to the total cost of prizes paid during the whole promotion.

Sample Input
```
5
3 1 2 3
2 1 1
4 10 5 5 1
0
1 2
```

Sample Output
```
19
```

8. **Graph 2 coloring**

You are given a graph G. Color the graph using 2 colors such that each vertex is given a color and no 2 vertices connected by an edge have the same color. If such a coloring is possible, then print the lexicographically smallest coloring.

Input Format
First line contains 2 integers N and M. N is the number of vertices and M is the number of edges.
The next M lines contain 2 integers each : U and V. This means that there is an edge between vertex U and vertex V. The vertices are numbered from 0 to N - 1.

Output Format
If there is no possible coloring, print “No”.
If there is a coloring, print “Yes” on the first line. The next line should contain the lexicographically smallest possible coloring.

Sample Input
5 6
0 1
1 2
2 3
3 0
0 4
0 5

Sample Output
Yes
0 1 0 1 1 1

Sample Input
5 7
0 1
1 2
2 3
3 0
0 4
0 5
4 5

Sample Output
No

---

9. Set Friends/Enemies

A war is being lead between two countries, A and B. As a loyal citizen of C, you decide to help your country's espionage by attending the peace-talks taking place these days (incognito, of course). There are n people at the talks (not including you), but you do not know which person belongs to which country. You can see people talking to each other, and through observing their behaviour during their occasional one-to-one conversations, you can guess if they are friends or enemies. In fact what your country would need to know is whether certain pairs of people are from the same country, or they are enemies. You may receive such questions from C's government even during the peace-talks, and you have to give replies on the basis of your observations so far. Fortunately nobody talks to you, as nobody pays attention to your humble appearance.

Abstract

Now, more formally, consider a black box with the following operations:

setFriends(x, y) shows that x and y are from the same country
setEnemies(x, y) shows that x and y are from different countries
areFriends(x, y) returns true if you are sure that x and y are friends
areEnemies(x, y) returns true if you are sure that x and y are enemies

The first two operations should signal an error if they contradict with your former knowledge. The two relations 'friends' (denoted by ~) and 'enemies' (denoted by *) have the following properties:

~ is an equivalence relation, i.e.

1. If x ~ y and y ~ z then x ~ z (The friends of my friends are my friends as well.)
2. If x ~ y then y ~ x (Friendship is mutual.)
3. x ~ x (Everyone is a friend of himself.)

* is symmetric and irreflexive

4. If x * y then y * x (Hatred is mutual.)
5. Not x * x (Nobody is an enemy of himself.)

Also

6. If x * y and y * z then x ~ z (A common enemy makes two people friends.)
7. If x ~ y and y * z then x * z (An enemy of a friend is an enemy.)

Operations setFriends(x, y) and setEnemies(x, y) must preserve these properties.

Input

The first line contains a single integer, n, the number of people.

Each of the following lines contains a triple of integers, c x y, where c is the code of the operation:

c = 1, setFriends
c = 2, setEnemies
c = 3, areFriends
c = 4, areEnemies

and x and y are its parameters, which are integers in the range [0, n), identifying two (different) people. The last line contains 0 0 0.

All integers in the input file are separated by at least one space or line break.

Output

For every ‘areFriends’ and ‘areEnemies’ operation write 0 (meaning no) or 1 (meaning yes) to the output. Also for every ‘setFriends’ or ‘setEnemies’ operation which contradicts with previous knowledge, output a –1 to the output ; note that such an operation should produce no other effect and execution should continue. A successful ‘setFriends’ or ‘setEnemies’ gives no output.

All integers in the output file must be separated by at least one space or line break.

Constraints

n < 10000, the number of operations is unconstrained.

Sample Input

```
10
1 0 1
1 1 2
2 0 5
3 0 2
 3 8 9
4 1 5
4 1 2
4 8 9
1 8 9
1 5 2
3 5 2
0 0 0
```

Sample Output

```
1
0
1
0
0
-1
0
```

---

10. Potato Chip Factory

A potato chip factory has a conveyor belt on which potatoes keep moving. Potatos keeping coming from one end, and they come tagged in increasing order starting from 1. That is the ith potato is tagged with the integer i. A robot looking at them, diverts rotten potatoes to a dump yard. The healthy ones pass and are dumped to a storehouse, piled up randomly.

Another conveyor belt takes healthy ones one at a time from the pile for processing, and a tag reader on the belt keeps reading their tags. The output is hence a jumbled

subsequence of the numbers. You want to answer the following queries from the Quality Assurance manager, using the current state of the binary tree:

1. Given a tag number i, is the corresponding potato healthy?
2. Given a healthy potato tag number i, how many potatoes before it are rotten?
3. Which is jth healthy potato?

Note that these queries are w.r.t. the currently known list of healthy potatoes and answers to them will change as more healthy potatoes are known; i.e. a potato that is healthy but has not yet been read by the reader is considered rotten while answering these queries.

Input\Output Format

Input will be a list of these commands:

1. SetHealthy x
   Potato with tag x is healthy,

2. CheckHealthy x
   Check if potato with tag x is healthy, print “YES” or “NO”.

3. RottenBefore x
   Print the number of rotten potatoes before the potato with tag x.

4. Healthy j
   Print the jth healthy potato.
   If j is more than currently known number of healthy potatoes, print “NONE”

Note: All queries should take log(n) time, where n is the currently known number of healthy potatoes.

Sample Input

```
CheckHealthy 1
SetHealthy 1
SetHealthy 5
SetHealthy 3
SetHealthy 6
CheckHealthy 1
CheckHealthy 3
Healthy 3
Healthy 6
RottenBefore 3
RottenBefore 6
RottenBefore 9
Quit
```

Sample Output

NO
YES
YES
5
NONE
1
2
4

Q11. Huffman Coding

Every wondered how compression algorithms like bzip2, zip, pdf, etc.. work? They use a variety of (lossless) compression algorithms, one of the most important stage being a method called Huffman coding.

A normal character occupies 1 byte of space. For example the string “Welcometofelicity” occupies 20 bytes or 160 bits, if we use ASCII coding. But since there are only 10 distinct characters, we can use 4 bits to encode, reducing to 80 bits.

The basic idea of Huffman coding is to use a different map from a character to its binary representation, and allow variable bit-length encoding of characters, further reducing the size. The compression is achieved by using less number of bits to represent a character that occurs more number of times. The algorithm to find such an optimal encoding works by constructing a binary tree as follows:

Step1: Get frequencies of characters from input string. Construct an array F of trees, where the trees are all originally containing only one node. A node is a pair: (char, freq)

F = [(W, 1), (e, 3), (l, 2), (c, 2), (o, 2), (m, 1), (t, 2), (f, 1), (i, 2), (y, 1)]

Step2: We sort this array of trees in non-decreasing order of frequency (of the root node):

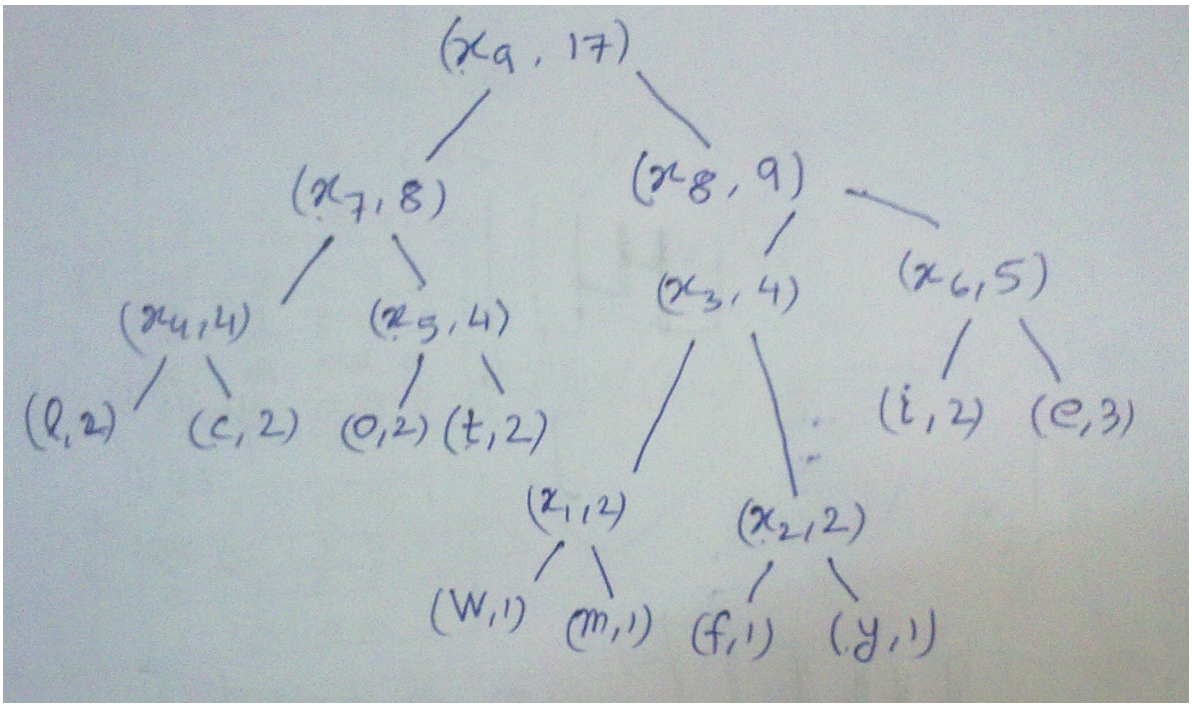
F = [(W, 1), (m, 1), (f, 1), (y, 1), (l, 2), (c, 2), (o, 2), (t, 2), (i, 2), (e, 3)]

Step3: If there is only one element in F, goto Step5, else create a new tree with root as (c, f) where c is an auxiliary character, not in the input string, and f is the sum of frequencies of the first two nodes. Make the first two nodes as the left and right child of the root node and replace them, with this new tree:

```
F = [ (x1, 2), (f, 1), (y, 1), (l, 2), (c, 2), (o, 2), (t, 2), (i, 2), (e, 3) ]
          /  \
   (W, 1)   (m, 1)
```

Step4: GoTo Step2

Step5: F now contains a single binary tree:



Step6: We now assign codes to each character based on the path from the root to that character. It will be a sequence of 0s & 1s, based on whether we move to the left or right of a node during the traversal:

l:000 c:001 o:010 t:011 w:1000 m:1001 f:1010 y:1011 i:110 e:111

Total bits required to encode input string: 4+4+4+4+6+6+6+6+6+9 = 55
Compression ratio (%) = CompressedSize/UncompressedSize = (55/80)*100 = 68.75 %

Your task in to now optimally compress the input using Huffman coding.

Input Format

Input will be some text, containing a sequence of type-able ASCII characters. Read till EOF. Let the number of distinct characters in input be n. (You have to compress everything including spaces, newlines, tabs, and any character in the input).

Output Format

Output n+1 lines, the first n lines containing a character, followed by a space and its prefix code in binary. The last line should contain the compression ratio (%), exactly up to 2 places after decimal. If there are multiple solutions, any one will do.

Sample Input

Welcometofelicity

Sample Output

l 000
c 001
o 010
t 011
w 1000
m 1001
f 1010
y 1011
l 110
e 111
68.75

All The Best :)