20^{th} South African Regional International Collegiate Programming Contest

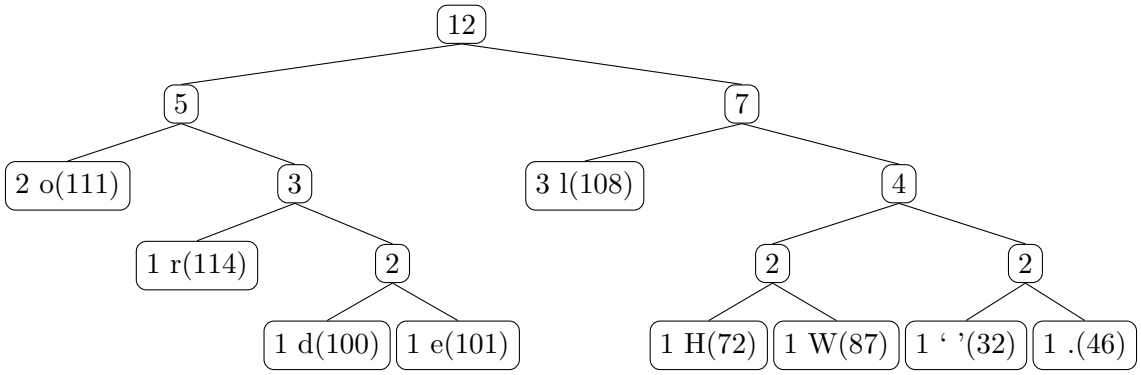
20 October 2018

Problem A — Green Balloon
Huff and Puff

Problem Description

You have discovered a piece of text encoded by an ancient computer system. After quite a bit of puffing, your research revealed that the mechanism is based on the Huffman encoding mechanism. Huffman encoding uses a variable-length encoding mechanism to encode more frequent characters using fewer bits than less frequent characters.

Huffman encoding is based on a binary tree such as the one shown below, which is used to encode the text "Hello World." The nodes at the bottom of the tree ("leaves") are labelled "F S(C)", where S is an ASCII character from the text, C is its ASCII code, and F is its frequency (the number of times it appears in the text). The non-leaf nodes are labelled only with a frequency. This frequency is defined as the sum of the frequencies of the two nodes below it ("children"). Every non-leaf node has exactly two children.

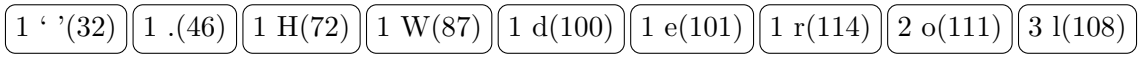


Note: ‘ ’ represents a space (without the quotes, which are present merely to indicate the presence of the space).

To use a Huffman tree to encode an ASCII character, follow the path from the top of the tree to the leaf node for that character, writing down a `0` when taking a left branch and a `1` when taking a right branch. Thus, for example, 'H' is encoded as 1100 (right, right, left, left) and 'e' as 0111. The text "Hello World." becomes `110001111010001110110100010100110111111`.

You have the encoded string of 0's and 1's from the ancient computer system, and you need to decode it. Unfortunately, your disaster-prone colleague accidentally deleted the Huffman tree used to generate the encoded string. Before he could do any more damage, you were able to find the frequency of each character, plus an algorithm that generates the Huffman tree from that information.

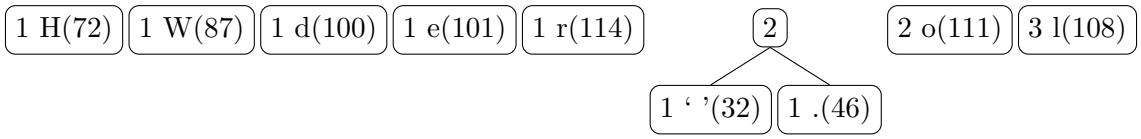
The algorithm you found works on an ordered list of trees. Initialise the list to contain a tree per character, each consisting of just a leaf node. Sort the list by increasing frequency, breaking ties by increasing ASCII code. In our "Hello World." example, the initial list would be:



Now repeat the following steps until the list contains only one tree, which will be the Huffman tree:

1. Remove the first two entries from the list, call them X and Y (in that order);
2. Create a new tree by creating a new non-leaf node, and making X and Y its left and right children respectively. Recall that the frequency of the new node is the sum of the frequencies of its children.
3. Insert the newly created tree into the list immediately after all nodes that have a strictly lower frequency than the new node.

In our “Hello World.” example, the list will look like this after the first iteration:



Input

The input will consist of an arbitrary number of test cases, but no more than 100.

The first line of each test case contains an integer C, where $2 \leq C$, indicating the number of characters to be used in the record.

The following C lines each contain two integers O, where $32 \leq O \leq 126$, and F, where $1 \leq F \leq 10\,000$, indicating that the character with ASCII code O appears F times in the text. The characters are listed in strictly increasing order of O.

This is followed by a single line containing `0`'s and `1`'s, which is the encoded form of the text. It is guaranteed that the decoded text contains at most 10 000 characters and is consistent with the provided frequency information.

Input is terminated by a `-1` on a single line.

Output

For each input test case, output the decoded text, followed by a newline.

Sample input

```
9
32 1
46 1
72 1
87 1
100 1
101 1
108 3
111 2
114 1
1100011110100011101101000101001101111
15
32 3
44 1
46 1
83 1
97 1
99 1
101 2
104 1
105 1
107 1
110 1
111 1
114 1
116 4
121 1
000010010000101011101011011111101111011100011111011111100001000110010111001011
-1
```

Sample output

```
Hello World.
See, not that tricky.
```

Time limit

2 second

20^{th} South African Regional International Collegiate Programming Contest

20 October 2018

Problem B — White Balloon
Recursive Function

Problem Description

In combinatorial analysis and other branches of mathematics, as well as in competitive programming competitions, recursive functions are often useful. When calculating the result of these kinds of functions, you must be very careful not to make calculation errors. The value of a recursive function usually grows exponentially. It is therefore very difficult to validate the calculations by hand. For these reasons your help is required to calculate several results of the following recursive function:

$$F(n) = \begin{cases} 0, & n < 0 \\ 1, & n = 0 \\ a \cdot F(n-1) + b \cdot F(n-2) + c \cdot F(n-3) + 2^d, & n > 0 \end{cases}$$

Input

The input consists of an arbitrary number of test cases, but no more than 10.

Each test case consists of a number of lines.

The first line of each test case contains four integers that represent the values of the coefficients a, b, c and d, where $0 \leq a, b, c, d \leq 10^9$, of the recursive function $F(n)$ as defined above.

The second line contains a single integer Q, where $1 \leq Q \leq 500$, representing the number of evaluations that are required.

The next Q lines each contain a single integer value n, where $-10^9 \leq n \leq 10^9$.

Input is terminated by `-1` on its own line.

Output

For each value of n you must output a single line with the value of $F(n)$.

IMPORTANT: As these values can become exceptionally large, all answers must be printed modulo $10^9 + 7$.

Sample input

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

Sample output

```
1
3
6
12
23
```

Time limit

120 seconds

20^{th} South African Regional International Collegiate Programming Contest

20 October 2018

Problem C — Orange Balloon fertiliser

Problem Description

Farmer Joe has been doing a lot of research into fertilisers and the exact ratios of Nitrogen (N), Phosphorus (P) and Potassium (K) needed for each crop he is growing.

Unfortunately, Farmer Joe is unable to buy fertiliser in the exact ratios he needs, but he is able to buy the standard ratios of fertiliser.

In order to get to the ratios he requires using the standard fertiliser ratios he is able to order, Farmer Joe has asked for your assistance. Given fertilisers with specific quantities of Nitrogen, Phosphorus and Potassium in them, you need to determine if you are able to get to the exact ratio Farmer Joe requires for his crop by mixing different counts of each bag of standard fertiliser (bags must be used in their entirety). The only limitation, is that his supplier is only able to supply him with a maximum of 50 bags of each standard fertiliser.

Input

The input consists of an arbitrary number of test cases, but not more than 10.

The first line of each test case is a single integer x, with $2 \leq x \leq 4$, indicating the number of standard fertilisers to consider.

Thereafter, x number of lines will follow providing detail of the amount of Nitrogen, Phosphorus and Potassium in each standard fertilisers Farmer Joe can get, represented as three integers n_i p_i k_i, where $0 \leq n_i, p_i, k_i \leq 15$, and $n_i + p_i + k_i > 0$.

After these x lines, one final line, represented as three integers n p k, where $0 \leq n, p, k \leq 500$ and $n + p + k > 0$, describes the desired fertiliser ratio. (The final weight and quantity of mixed product isn't relevant, only the ratio).

The end of input is indicated by a line containing only `-1`.

Output

For each sample, simply output *yes* on a line if the desired ratio is possible given the available standard fertilisers or *no* on a line if the desired ratio isn't possible. Note, the output is case sensitive.

Sample input

```
2
3 2 1
7 1 3
2 1 2
3
3 2 1
7 1 3
1 2 5
2 1 2
-1
```

Sample output

```
no
yes
```

Time limit

2 seconds

20^{th} South African Regional
International Collegiate Programming Contest

20 October 2018

Problem D — Red Balloon
Power2 Game

Problem Description

Alice and Bob have developed a simple game to both pass the time and practice their arithmetic skills.

The game works as follows:

- Alice and Bob decide upon a number.
- Alice always takes the first turn, after which they alternate.
- The player whose turn it is performs one of two actions, depending on the current value of number at that time.
  - If the number is a power of two, the number is divided by two.
  - If the number is not a power of two, the largest power of two less than the number is subtracted from the number.
- This process repeats until there is a winner.
- The winner of the game is whoever reduces the number to 1.

Your task is to determine, for a given number, who the winner of the game is.

Input

Input consists of an arbitrary number of test cases, but no more than 10 000. Each test case consists of a single number N, where $1 < N < 2^{63}$, the start number as selected by Alice and Bob. Use "`long long`" in C/C++ and "`long`" in Java.

Input is terminated by `-1` on it's own line.

Output

For each test case, output the name of the winner, either "Alice" or "Bob".

Sample input

```
10
4251
4
12413211
-1
```

Sample output

```
Bob
Alice
Bob
Alice
```

Time limit

1 second

20^{th} South African Regional International Collegiate Programming Contest

20 October 2018

Problem E — Pink Balloon
Labyrinth Chess

Problem Description

Bob is a big fan of chess and every moment of the day he looks for an opportunity to play a game. Even in his dreams, Bob always seems to think about this ancient and valuable entertainment. It is from a recurring chess dream that Bob has been having lately, that he has asked you to help him.

In his dreams, Bob is a chess piece on a chessboard with dimensions $n \times m$. To be able to wake up he must get to row R, column C. On this board, some positions may be blocked by other immovable chess pieces and therefore can't be occupied by Bob after making a move. Bob has the choice of being either a knight or a king. He must choose wisely to maximise his chances of awakening from his terrible dream.

A knight can move to a square that is two squares away horizontally and one square vertically, or two squares vertically and one square horizontally. A king can only move one square at a time, but can move in any direction (horizontally, vertically or diagonally). Can you help Bob make his choice?

Input

The input consists of an arbitrary number of test cases, but no more than 150.

The first line of each case contains four integers n, m, R and C, where $0 < n, m \leq 100$, $1 \leq R \leq n$ and $1 \leq C \leq m$. n representing the total number of rows on the board, m the total number of columns on the board, R the row and C the column corresponding to the position which Bob needs to reach.

The following n lines each a string consisting of m characters. Each character is one of the following:

1. `B` – represents Bob's current location. Each test case will have exactly one `B`.
2. `.` – represents an unoccupied square.
3. `x` – represents an occupied square.

Input is terminated by `-1` on its own line.

Output

For each test case you must output a line with the choice that maximises Bob's chances of awakening. If, with both of the choices (knight and king), Bob manages to wake up, then output a result of `BOTH`. If neither choice allows him to reach the position (R, C), then output `NONE`. If it is only possible by doing knight movements, then output `KNIGHT`. If it is only possible by doing king movements, then output `KING`.

Sample input

```
3 4 2 4
.Bx.
..x.
..x.
5 4 1 1
....
.xxx
.x.x
x...
..B.
-1
```

Sample output

```
KNIGHT
KING
```

Time limit

1 second

20^{th} South African Regional International Collegiate Programming Contest

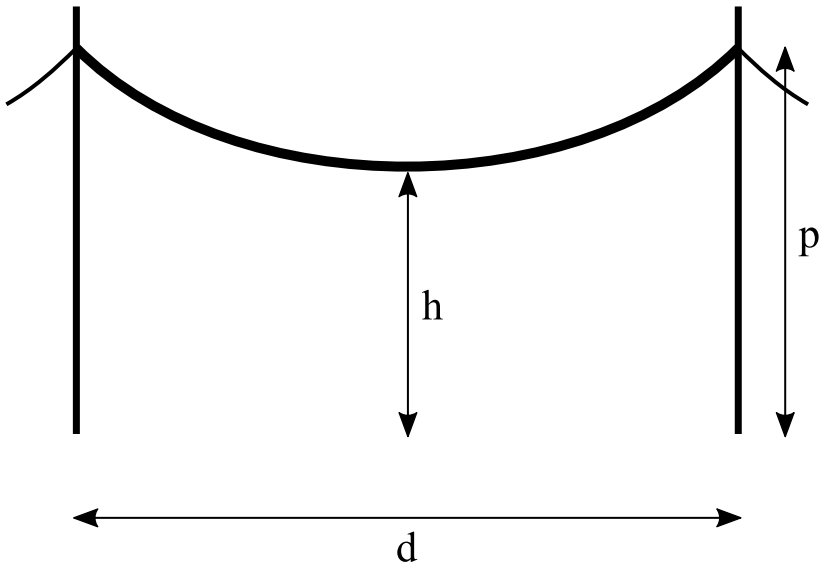
20 October 2018

Problem F — Blue Balloon
The Cable Guy

Problem Description

Given the age of some of the telephone cables hanging from poles across the continent and the prevalance of them being stolen by "entrepreneurs", you have been asked to help replace all of the cables with newer cables that are less likely to be stolen (they don't use copper).

While determining how much cable you will need you realised that, although you can measure the distance between the poles and the height of the poles, you don't know exactly how much cable is needed because the cable sags a bit between the poles. Here's a diagram:



Fortunately you've paid attention in the maths classes, and you remember that cables hang in a very particular shape (assuming that the cable won't stretch under its own weight). The functional form of this shape is called a catenary, and defines the shape of the cable as:

$$f(s) = a \cdot cosh\left(\frac{s}{a}\right)$$

where $cosh(x)$ is defined as:

$$cosh(x) = 0.5(e^x + x^{-x})$$

Note that $f(s)$ defines the "height" of the cable, with its lowest point at $f(0) = a$.

The shape in which the cable hangs is thus completely described by the catenary function $f(s)$ above, once you have computed the value of a. You are given the three parameters necessary to find a, namely h (height of cable above ground at its lowest point), p (height of the pole) and d (distance between the two poles).

Having found a, you must then compute the arc length of $f(s)$, i.e. the length of the cable.

The arc length of the catenary function for the range $[0, s]$ for all $s > 0$ is defined as:

$$l(s) = a \cdot sinh\left(\frac{s}{a}\right)$$

Input

Your input will consist of an arbitrary number of records, but no more than 10.

Each record consists of a single line containing the three integers p, d and h, where $1 < p \leq 20$, $1 \leq d \leq 50$ and $1 \leq h < p$.

Input is terminated with a line containing -1.

Output

For each input record, you must output the length of the cable described by the input record. Your answer must be rounded to three decimal places.

Sample input

```
10 20 8
-1
```

Sample output

```
20.524
```

Time limit

1 second

20^{th} South African Regional International Collegiate Programming Contest

20 October 2018

Problem G — Yellow Balloon
AfriBucks

Problem Description

The African Union has launched their own cryptocurrency, AfriBucks, to foster economic development on the continent. Some politicians saw an opportunity to enrich themselves by submitting invalid transactions to the blockchain. Due to the poor implementation of AfriBucks, these invalid transactions were accepted and added to the blockchain. You were brought in to determine at which point in time the corruption started in order to assist with the investigation.

The blockchain has M consecutive blocks, each block consists of N entries. A root hash is calculated for each block using its N entries which requires a total of $2N - 2$ hash calculations per block. For each block, the **first** of the N entries is the root hash of the previous block. The remaining $N - 1$ entries are the hashes of the transactions to be included in the block. Since the first block in the chain does not have a predecessor, it should use the genesis hash `0000000000` as the first of its N entries. The transactions of each block must be hashed before using them in the root hash calculation. The concept for the first three blocks is illustrated in the figure below.

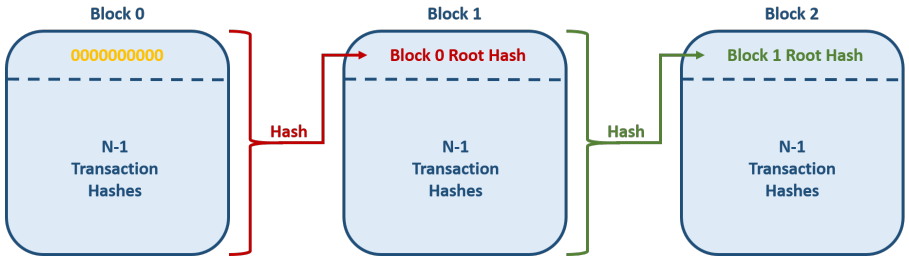


Figure 1: Illustration of blockchain hashing

Due to a limited budget, the developers of AfriBucks used the very primitive hashing algorithm `DJB2`. A `DJB2` hash is calculated using any arbitrary input string X with

$$D(X) = \bar{D}(X, len(X))$$

where $len(X)$ is the length of the input string X in number of characters and $\bar{D}$ is defined as:

$$\bar{D}(X, i) = \begin{cases} 5381 & for \quad i = 0 \\ \bar{D}(X, i-1) \times 33 \oplus X[i-1] & for \quad 0 < i \leq len(X) \end{cases}$$

where $\oplus$ is the XOR operator. The characters of X are XORed using their ASCII integer representation. The hash is stored as a 32-bit unsigned integer to allow for overflow. Although the hash is internally calculated as an integer, it should be returned as a 10-character string padded with leading 0's. The following shows an example of a `DJB2` hash calculated from the given transaction input string.

$$D(\text{“1552273520\_0.714125\_avUrTq\_gAaxtc”}) = \text{“0037240479”}$$

Given a sequence of transactions $[T(0), T(1), ..., T(MN - M - 1)]$, the root hash $H(j)$ of block j is calculated as:

$$H(j) = \bar{H}(j, 2N - 2)$$

where $\bar{H}$ is defined as:

$$\bar{H}(j,k) = \begin{cases} D(\bar{H}(j, 2k - 2N) +\!\!+ \bar{H}(j, 2k - 2N + 1)) & for \quad N \leq k \leq 2N - 2 \\ D(T(jN - j + k - 1)) & for \quad 0 < k < N \\ \bar{H}(j - 1, 2N - 2) & for \quad j > 0, k = 0 \\ 0000000000 & for \quad j = 0, k = 0 \end{cases}$$

and $+\!\!+$ is a simple string concatenation.

Your mission is to find out when the corruption started, that is, finding the index of the **first** block in the chain that contains a maliciously manipulated transaction. Memory limitations are not a concern here, but time constraints are of utmost importance.

Input

The input consists of an arbitrary number of test cases, but not more than 5.

The first line is N, the number of entries in a block. N is always a power of two with $2 \leq N \leq 1024$.

The second line is M, where $1 \leq M \leq 15\ 000$, the total number of blocks in the chain.

The following M lines contain the **expected** root hashes of the blocks (in the correct order). The remaining $M(N-1)$ lines represent the transactions as previously explained (also in the correct order). Transactions do not contain any white spaces and no further processing is required on them, except calculating their respective hashes.

Output

The output is the index of the **first** corrupted block in the chain. If no malicious transactions were detected, the output should be -1.

Sample input

```
4
5
1084397125
0600462444
3307222183
0097440334
0302234539
1539172279_0.04975_hw0jkD_ROHlhW
1539172857_0.865625_Oa7s27_4eB2gR
1539173600_0.8605_2cpZ3F_hrcMbe
1539174205_0.06925_zgWHNl_kFIYgP
1539174355_0.410375_WxhgUT_SyN4Hk
1539174899_1.078375_YuoKed_sVWC8f
1539175122_0.051375_a7Snrs_ou20j8
1539175956_0.325375_rkTfbq_PIykc7
1539176250_0.201375_CW0YVf_u5CSbr
1539176291_1.043625_I3KYCF_Fs0DGD
1539176796_0.010125_qpmLxY_z6HaXk
1539177257_1.123625_E9RdV4_DuRL9w
1539177447_1.144375_neGgHk_R5Sdvw
1539178114_0.20775_HQP6BD_0zar0k
1539178427_0.902625_ekRl4q_aaFgSW
4
4
0672414848
0392895212
1799483585
3590281358
1546051770_0.78775_F2JEdc_sYPUTR
1546052766_0.070625_ah6tSG_5Db21C
1546053355_0.4185_hy6DZ5_GL5ddF
1546054006_0.201875_NC0v7S_X05HoK
1546054082_0.933375_78B5Ck_3tZ1Me
1546054190_0.591_aHArED_MatBEU
1546054502_0.39125_OC8rjH_XPdnT2
1546055229_0.10975_2acczw_ZLVgOs
1546055847_1.183_BA9PGH_iNpIW3
1546056307_0.23575_xS4LyN_LMi11f
1546056388_0.346_hBL9DZ_q6QFZB
1546057070_0.22775_R4UhVH_h9UPsW
-1
```

Sample output

```
3
-1
```

Time limit

2 seconds

20^{th} South African Regional International Collegiate Programming Contest

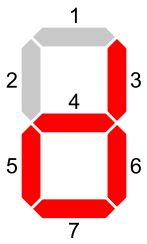
20 October 2018

Problem H — Purple Balloon Segments

Problem Description

7-segment displays can display a range of digits and letters, and each segment is addressed by a given number, which is used by the controller to turn that segment on or off.

For example, to display the letter d, segments 3, 4, 5, 6 and 7 need to be illuminated (on) and segments 1 and 2 must not (off).



Because there are only 7 segments, not all letters can be displayed and some letters need to be tweaked slightly to be able to display them.

You have been tasked to determine the longest possible word that can be displayed on a 7-segment display to aid with determining how many 7-segment display modules will be needed.

Input

The input consists of an arbitrary number of test cases, but not more than 10.

The first line of each test case contains a single integer n, where $0 < n \leq 60\ 000$, representing the number of words to test. The following n lines each contain one word. A word consists of at least 1 character and no more than 30, and will consist only of characters from the sets $0 - 9$ and $a - z$. Following the word list there are 36 lines, one for each of the 10 digits $(0 - 9)$ and 26 lowercase characters $(a - z)$ in the English alphabet. Each of these lines contains the digit or character, followed by a list of segments that needs to be on to display that digit or character, followed by a `-` at the end of the line. If that digit or character cannot be displayed, then the line will only have the digit or character and a `-`.

The end of input is indicated by a line containing only `-1`.

Output

For each test case, simply output a line with the length of the longest word, along with the word itself, that is possible for the given set of allowed characters. In the case where more than one word is the longest, the first longest word in the list must be returned. In the case where no word can be displayed, only output a length of 0.

Sample input

```
9
the
quick
brown
fox
jumps
over
the
lazy
dog
0 1 2 3 5 6 7 -
1 3 6 -
2 1 3 4 5 7 -
3 1 3 4 6 7 -
4 2 3 4 6 -
5 1 2 4 6 7 -
6 1 2 4 5 6 7 -
7 1 3 6 -
8 1 2 3 4 5 6 7 -
9 1 2 3 4 6 7 -
a 1 2 3 5 4 6 -
b 2 4 5 6 7 -
c 4 5 7 -
d 3 4 5 6 7 -
e 1 2 4 5 7 -
f 1 2 4 5 -
g 1 2 5 6 7 -
h 2 4 5 6 -
i 6 -
j 3 6 7 -
k -
l -
m -
n 4 5 6 -
o 4 5 6 7 -
p 1 2 3 4 5 -
q 1 2 3 4 6 -
r 4 5 -
s -
t 2 4 5 7 -
u 5 6 7 -
v -
w -
x -
y 2 3 4 6 7 -
z -
-1
```

Sample output

```
3 the
```

Time limit

1 second