FRISCO First Bytes High School Programming Contest
10-17-2015
Advanced Category

- **Time Allowed: two hours.**
- **Each team must only use one of the high school's computers.**
- **Answer the questions in any order.**
- **Use only Java 1.7, minGW g++, or MS Visual Studio 12 C/C++**
- **Your program source code must be contained in one source file.**
- **Do not use the "package" construct in your Java code.**
- **Your programs must read from a named file and output to System.out (cout for C/C++ programmers.)**
- **Do not access the web.**
- **Do not use any recording device containing code.**
- **Your solutions must be entirely yours, typed by you during the time allowed for the contest.**
- **As soon as you have solved a problem, submit ONLY the source file via your `PC^2` client.**

Scoring

Equal points will be awarded to each question, even though they may not be equally difficult.

In order to break ties, we will use penalty points. The penalty points for a question will be zero if the question is not answered correctly. If a correct submission occurs for a question at time T minutes, then T penalty points will be added, plus 20 penalty points for each incorrect submission for that question.

A. **Algebra**

Input File: A.txt
Run Time Limit: 10 sec

Given an equation containing unknowns and a list of possible integers to substitute for the unknown, determine if there is an assignment of values to the unknowns that makes the equation correct.

Input:
Each test case in the input file consists of two lines:

* The first line contains a sequence of natural numbers. The first one ($1 \leq n \leq 5$) is the number of unknowns that will occur in the expression. It is followed by a sequence of n integers $v_1 ... v_n$ ($0 \leq v_i \leq 50$), which are the values to be assigned to the unknowns. Finally, there is an integer m ($0 \leq m \leq 1000$) representing the desired result of the evaluation of the expression.

* The second line contains an arithmetic expression composed of lowercase letters ($a - z$), and binary operators ($+, -, *$). This expression will contain n unknowns, represented by n different lowercase letters, without repetitions.

  The expression will not contain any blanks and will always be syntactically correct, i.e. it will be an unknown or it will have the form (e_1 *op* e_2), where e_1 and e_2 are expressions and *op* is one of the three possible binary operators.

  The expression is to be evaluated using the operator precedence rules of regular algebra. For example, a+b*c-d*e, with a=5, b=6, c=7, d=2, and e=3, has value 5+6*7-2*3 = 5+42-6 which is 41.

The input will finish with a dummy test case of just one line containing 0 0, which must not be processed.

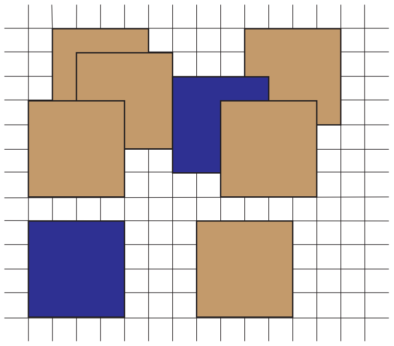
Output:
For each test case, print a single line with *YES* if there exists an assignment of the values $v_1 ... v_n$ to the unknowns such that the expression evaluates to m, and *NO* otherwise. Notice that each value v_i must be assigned to exactly one unknown.

Sample Input	Output for Sample Input
3 2 3 4 14	YES
a+b*c	NO
2 4 3 11	YES
a-b	
1 2 2	
a	
0 0	

B. **Solar Panel Slip**

Input File: B.txt
Run Time Limit: 10 sec

Sunny had solar panels installed on his roof. There were N of them, each 4x4 meters, But unfortunately the installers (and Sunny) did not allow for the strong winds in Texas. Some of the panels have slipped from their original position, but remain connected and oriented parallel to the roof edges. The wiring has also (remarkably) caused the corners of each panel to have integer coordinates. Some of the panels took a beating and stopped working completely. They are shaded dark in the figure below.



You are given the positions of all the solar panels and, for each, an indication of whether it still works. Calculate the total area of the working panels that can be seen from directly above the roof.

Input:
There may be up to 100 test cases. Each one follows the format below. Following the last test case is a line containing a single integer 0. Do not process this line. Each test case begins with the integer N ($1 \leq N \leq 100$) giving the total number of solar panels. N lines follow, each giving the x, y ($2 \leq x, y \leq 98$) integer coordinates of the center of the panel and the word *OK* if the panel works, or *FAULTY* if it doesn't work.

Panels are listed strictly in the order of their height from the roof's surface. Therefore, if panel i is partially, or totally occluded by panel j, panel i will be listed before j.

In spite of the repositioning of the panels, they did not bend. Each remains flat and parallel to the surface of the roof. Therefore cyclic overlaps cannot occur. For example, if we use the notation $x \rightarrow y$ to mean that x partially occludes y, then we cannot have $x \rightarrow a \rightarrow b \cdots \rightarrow x$ for any number of intervening panels between x and itself.

Output:
For each test case, give the test case number starting at 1, and give the total area in square meters of the working solar panels.

Sample Input	**Output for Sample Input**
2	Case 1: 7
6 6 OK	Case 2: 0
7 7 FAULTY	Case 3: 79
2	
6 6 OK	
6 6 FAULTY	
8	
4 11 OK	
5 10 OK	
3 8 OK	
3 3 FAULTY	
12 11 OK	
9 9 FAULTY	
10 8 OK	
10 3 OK	
0	

C. **Crazy Search**

Input File: C.txt
Run Time Limit: 3 sec

Problem
Many people like to solve hard puzzles some of which may lead them to madness. Well, it's almost Halloween!

Your task is to write a program that, given a String S, and the size N of a valid substring, determines the number of different substrings of size N that appear in S. A substring of S is a contiguous subsequence of the characters in S

As an example, consider N=3 and S = "daababac". The different substrings of size 3 that can be found in this text are: "daa"; "aab"; "aba"; "bab"; "bac". Therefore, the answer is 5.

Input
The input begins with the number of test cases to follow. Each test case takes up one line containing the number N, then a space, and then the string S. S will only contain lower-case alphabetical characters. You may assume that the number of substrings will not exceed 16 Million.

Output
For each test case the program should output a line beginning with the String "Case number X: " followed by an integer corresponding to the number of different substrings of size N found in the given text. X begins at 1 and progresses upward by one for each test case.

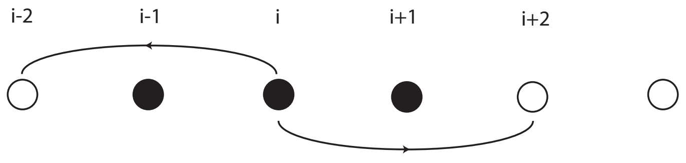
Sample Input	**Output for Sample Input**
2	Case 1: 5
3 daababac	Case 2: 7
2 yabadabadoo	

D. **Linear Solitaire**

Input File: D.txt
Run Time Limit: 10 sec

Given a peg board made up of a long sequence of equally spaced holes and some pegs initially placed in holes, can you reduce the board to a single peg by repeatedly applying the following move:

A peg at position i with a neighboring peg at position $i+1$ and a gap at position $i+2$ can be reduced to a single peg at position $i+2$ by making the peg at position i jump over and take its immediate neighbor. The rule can equally be applied by making a peg jump over and take its immediate neighbor to the left if that neighbor has a gap to its left. These two situations are illustrated in the diagram below.



Two possible moves for the peg at position i

Input:
There will be multiple board descriptions, each occupying a single line of text. A board description will be a binary sequence such as 0110100 where 0 means the absence of a peg and 1 means the presence of a peg. The board above would be described by the string 011100 and the two moves indicated would result in boards 100100 and 010010.

A board description will not be longer than 20 symbols.
Input is terminated by end of file.

Output:
For each board description output the single line of text containing the case number, as shown below, 'yes' if the board can be reduced to a single peg, and 'no' if it cannot.

Sample Input	Expected Output
1	Case 1: yes
0	Case 2: no
1111	Case 3: no
1110111	Case 4: no
111110	Case 5: no
0110	Case 6: yes
010111100	Case 7: yes
0110010110	Case 8: yes

E. **Code Distance**

Input File: E.txt
Run Time Limit: 10 sec

There are many distances in our world. The distance to the moon is 384.4 million meters and the distance to Alpha Centauri is 4.367 light years. There are also distance measures between bit fields. For example, the Hamming Distance between two n bit binary code words is the number of bits that must be changed in one of them to make it equal to the other.

Two Strings of characters have a Levenschtein distance D between them if a minimum of D of the following operations are necessary to transform one of the Strings into the other.

- Add one character
- Remove one character
- Substitute one character for another

The two Strings "Hello World" and "Hell Word" have a distance of 2. And the two strings, "abcghijkpq" and "bcdehijprt" have a distance of 7.

Given a series of pairs of Strings, calculate the distance between each pair.

Input:
There are multiple test cases in the input, each occupying two lines of text. For n test cases there will be $2n+1$ lines of text, $1 < n < 100$, the last being a line containing the word "End". Do not process this line. Each String will contain between 1 and 50 characters inclusive. All characters will be chosen from the alpha-numeric set plus the space character:
$\{A,B,C,...,Z\} \cup \{a,b,c,...,z\} \cup \{1,2,3,...,9\} \cup \{' \; '\}$

Output:
For each of the n pairs of Strings in the input, print the case number as shown below and the distance between those two Strings.

Sample Input	**Output for Sample Input**
Hello World Hell Word Magic moments All Magic Tu eres Hermosa Tu estas Hermosa abcghjkpq bcdehijprt End	Case 1: 2 Case 2: 12 Case 3: 3 Case 4: 7