South African
Computer Olympiad Training
DAY 2

Overview

Author	**Bruce**	**Shen**	**Ruan**	**Christaan**
Problem	**Frame**	**Compress**	**Coded**	**Water**
Program name	`frame.exe`	`compress.exe`	`coded.exe`	`water.exe`

Source name	`frame.pas`
	`frame.java`
	`frame.cpp`
		`compress.pas`
		`compress.java`
		`compress.cpp`
			`coded.pas`
			`coded.java`
			`coded.cpp`
				`water.pas`
				`water.java`
				`water.cpp`

Input file	`stdin`	`stdin`	`stdin`	`stdin`
Output files	`stdout`	`stdout`	`stdout`	`stdout`
Time limit	1 second	1 second	2 second	1 second
Num. of tests	5	10	10	5
Points per test	20	10	10	20
Total points	**100**	**100**	**100**	**100**

The maximum total score for Day2 is 300 points.

Frames

Author

Bruce Merry

Introduction

Farmer John has so many pictures of his cows that he doesn't have enough space to display them all without overlapping. He is planning how to lay them out by putting up only the frames (with the inside transparent). He has found a pleasing arrangement, but realises that he doesn't know in what order the frames were placed. If he takes the frames down to put the pictures in, he may never get the same arrangement!

Task

Farmer John realised that if he photographed the wall, he could figure out the ordering. Given a snapshot of the wall, compute a possible order in which the frames were put up.

The snapshot is represented as a rectangular grid of characters. A '.' represents an empty space, and a letter from 'A' to 'Z' (uppercase only) represents part of a frame. Each frame is assigned a unique letter from 'A' to 'Z'.

Frames are 1 cell wide and at least 3 cells in each direction. You are able to see at least part of each side of each frame (a corner counts as two sides).

Example

```
.CCC....
ECBCBB..
DCBCDB..
DCCC.B..
D.B.ABAA
D.BBBB.A
DDDDAD.A
E...AAAA
EEEEEE..
```

This is a possible grid. It results from placing E, then D, then A, then B and finally C.

Input format

The first line of the input file contains two space-separated integer, R and C. R is the number of rows in the grid and C is the number of columns. The following R lines each contain C characters (with no whitespace), representing the R rows of the grid.

Sample Input

```
9 8
.CCC....
ECBCBB..
DCBCDB..
DCCC.B..
D.B.ABAA
D.BBBB.A
DDDDAD.A
E...AAAA
EEEEEE..
```

Output format

The output file consists of a single line of letters, which are the labels of the frames in the order that they are placed, with no spaces separating them.

Only letters that appear in the input can appear precisely once in the output and no others. If more than one output is possible, any valid one will be accepted. There will always be at least one valid output.

Sample Output

```
EDABC
```

Constraints

1 <= R <= 100
1 <= C <= 100

Time limit

1 second.

Scoring

100% for a correct answer.
0% for an incorrect answer.

Compress

Author

Shen Tian

Introduction

Characters are ordinarily encoded as binary words of fixed length, such as 8-bit for ASCII or 16-bit for UNICODE. However, this does take up a fixed amount of space per character, and generally does not yield the smallest file sizes. A method of compressing text is to encode characters as variable bit length binary words. This improves the space utilisation compared to fixed length words in two ways: 1. it allows shorter words when the number of different characters is not a power of two. i.e. there are not unused combinations; 2. It allows often-used characters to be encoded using shorter words.

A problem encountered in this approach is that ambiguity may arise during decompressing/decoding if the binary representation of one character is a prefix of another. For example, if A = 1001010 and B = 1001, it is impossible to tell if 10010101010… starts with A or B.

Task

Given the frequency with which characters occur in a piece of text, find the optimal way to encode the text such that the compressed result is as short (in bits) as possible. Assume the encoding/decoding key itself is of negligible size. It must be made sure that no binary representation of one character is a prefix of another. There may be several optimal solutions. Give only one.

Example

Consider the string CABCABC. C occur thrice, while A and B both occur twice. An optimal way to encode it would be C = 0, A = 10, B = 11. Then the string becomes 01011010110.

Input format

Your program should read its input from stdin. The first line of input contains the single integer N, which is the number of different characters in a piece of text. Lines 2 to N+1 contains numbers F_1 to F_N in that order, where character i occurs F_i times in the text.

Sample Input

```
3
2
2
3
```

Output format

Your program should write its output to stdout. The output should give the encoding key. It should contain N lines, where line i contains the binary word (a string of 1’s and 0’s) with which to represent character i with.

Sample Output

```
10
11
0
```

Constraints

1<=N<=5000
1<=F_i<=400 for all i

Time limit

1 Second per case.

Scoring

Let X be the length of the text (in bits) compressed with an optimal key, and let Y be the length of the text compressed using your key. If X = Y, a score of 10 will be given. If X<>Y then a score of 9*e^(1.2*(X-Y)/X), rounded to the nearest integer, will be given. If your encoding key is ambiguous, a score of zero will be given. Similarly, if the output format is invalid (e.g. containing characters other than 1 and 0), a score of zero will be given.

Coded

Author

Ruan Havenstein

Introduction

The Mad Hatter (from Alice in Wonderland) has devised a devious new way to entertain his teatime guests. He gives them an encrypted string and a transition table that shows them how to transform each letter. What he doesn't tell them is how many times they should transform the string until it is the original string again.

Task

Mr. Hatter needs your help in constructing his transition table so that the guests take as long as possible to transform the string back to its original self.
As input you will receive the string to be encoded. The Mad Hatter will then use his table to encode it once (his guests don't know this, so they will blindly transform the string until it looks right.)

Example

Suppose the input (case sensitive) was
"COMPUTER"
then the Mad Hatter would construct the following transformation table: CEMOPRTU

Replace this	*With this*
C	U
E	O
M	P
O	C
P	R
R	M
T	E
U	T

(it should be noted that the table is a permutation, in other words, each column only contains each distinct letter once)

In other words the Mad Hatter would(in this example) give his guest the following string:
"UCPRTEOM"
which would then take 14 steps to transform correctly:
"TURMEOCP"
"ETMPOCUR"
"OEPRCUTM"
"CORMUTEP"
"UCMPTEOR"
"TUPREOCM"
"ETRMOCUP"
"OEMPCUTR"
"COPRUTEM"
"UCRMTEOP"
"TUMPEOCR"
"ETPROCUM"
"OERMCUTP" and finally
"COMPUTER"

Input format

As input, a single line with a string (with no spaces is given). The string is case sensitive and could have characters with ASCII codes 33 up to 126 inclusive (none of these are white space characters).

Sample Input

```
COMPUTER
```

Output format

Your output should describe a possible(because there is more than one "best" solution) transition table that maximises the time the Mad Hatter's guest takes to transform the string. Each line should contain two characters, not seperated by a space. In other words the output will be as many lines long as there are distinct letters in the string. Please do not output letters that would not have an influence on the string transformations, as this would score 0 marks. You may only transform to letters that are present in the original input string. (E.g. in this example you can't transfrom something to, say, "Z" as it's not in the input)

Sample Output

```
CU
EO
MP
OC
PR
RM
TE
UT
```

Constraints

1• string length• 250

Time limit

2 seconds.

Scoring

10 marks for a correct solution.
0 marks otherwise.

Water Pipes

Author

Christaan Scheepers

Introduction

The city of Kwaga-Kwaga has recently had a lot of trouble with flooding in the center city area. After a thorough investigation they discovered that the storm water pipes are blocked. Unfortunately they only have a limited quantity of explosives to use to open up some of the pipes. A few private contractors have already created plans that identify which pipes need to be blown open.

Task

Your task is to help the city to identify the optimal use of their explosives by calculating the maximum water flow capacity through the storm water pipe system after the pipes have been blown open. You will be provided with the number of open pipes and the number of distribution points that will be used. You need to calculate the maximum flow capacity from distribution point 1 to distribution point M (M will always be the last distribution point). Figure 1 shows a proposed solution for which the total water flow capacity from distribution point 1 to 5 is 7 units.

A distribution point can take input from any number of pipes and outputs to any number of pipes.

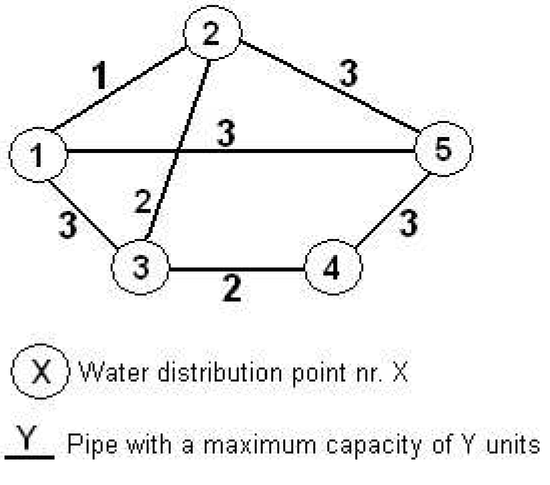
2
1
3
3
1
5
3
2
3
3
4
2
X Water distribution point nr. X
Y Pipe with a maximum capacity of Y units

Figure 1

Input format

The first line of input will consist of two integers N the number of pipes and M the number of distribution points separated by a single space. The following N lines will describe the pipes using three integers A, B the two distribution points linked by the pipe and F the total water flow capacity through that pipe e.g. 1 2 3 is a pipe that links distribution point 1 and 2 and has a maximum flow capacity of 3 units. There may be multiple pipes connecting the same two distribution points.

Sample Input

```
6 5
1 2 1
1 5 3
1 3 2
2 5 3
3 4 2
4 5 3
```

Output format

The output must contain one integer value which describes the maximum water flow capacity from the first to the M'th distribution point.

Sample Output

```
6
```

Constraints

$2 < N < 500$
$2 < M < 500$
$0 <=$ Maximum pipe capacity $<= 3000$
$0 <$ Total water flow capacity < 32000

Time limit

1 second.

Scoring

- The correct total water flow capacity will score 100%
- A total water flow capacity that is higher than half of the correct total water flow capacity but less than the correct total water flow capacity will score as follows:

Score % = 70 * (Your total flow) / (Correct total flow)

- Incorrectly formatted output will score 0%
- Everything else will score 0%