



# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

## Problem Set

# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

## Problem 1 – Red Balloon Stockbroker Grapevine

Stockbrokers are known to overreact to rumours. You have been contracted to develop a method of spreading disinformation amongst the stockbrokers to give your employer the tactical edge in the stock market. For maximum effect, you have to spread the rumours in the fastest possible way.

Unfortunately for you, stockbrokers only trust information coming from their 'trusted sources'. This means you have to take into account the structure of their contacts when starting a rumour. It takes a certain amount of time for a specific stockbroker to pass the rumour on to each of his colleagues. Your task will be to write a program that tells you which stockbroker to choose as your starting point for the rumour, as well as the time it will take for the rumour to spread throughout the stockbroker community. This duration is measured as the time needed for the last person to receive the information.

Your program will input data for different sets of stockbrokers. Each set starts with a line with the number of stockbrokers. Following this is a line for each stockbroker which contains the number of people who they have contact with, who these people are, and the time taken for them to pass the message to each person. The format of each stockbroker line is as follows: The line starts with the number of contacts ( $n$ ), followed by  $n$  pairs of integers, one pair for each contact. Each pair lists first a number referring to the contact (e.g. a '1' means person number one in the set), followed by the time in minutes taken to pass a message to that person. There are no special punctuation symbols or spacing rules. For each set of data, your program must output a single line containing the person who results in the fastest message transmission, and how long before the last person will receive any given message after you give it to this person, measured in integer minutes.

Each person is numbered 1 through to the number of stockbrokers. The time taken to pass the message on will be between 1 and 10 minutes (inclusive), and the number of contacts will range between 0 and one less than the number of stockbrokers. The number of stockbrokers will range from 1 to 100. The input file is terminated by a set of stockbrokers containing 0 (zero) people.

It is possible that your program will receive a network of connections that excludes some persons, i.e. some people may be unreachable. If your program detects such a broken network, simply output the message "disjoint". Note that the time taken to pass the message from person A to person B is not necessarily the same as the time taken to pass it from B to A, if such transmission is possible at all.

## SAMPLE INPUT

```
3
2 2 4 3 5
2 1 2 3 6
2 1 2 2 2
5
3 4 4 2 8 5 3
1 5 8
4 1 6 4 10 2 7 5 2
0
2 2 5 1 5
0
```

## SAMPLE OUTPUT

```
3 2
3 10
```

# Third Southern African Regional ACM Collegiate Programming Competition

**Sponsored by IBM**

## Problem 2 – Yellow Balloon Frame Stacking

Consider the following 5 picture frames placed on an 9 x 8 array.

..... EEEEEE..	..... .....	..... .....	..... ..BBBB..	.CCC.... .C.C....
E....E..	DDDDDD..	.....	..B..B..	.C.C....
E....E..	D....D..	.....	..B..B..	.CCC....
E....E..	D....D..	....AAAA	..B..B..	.....
E....E..	D....D..	....A..A	..BBBB..	.....
E....E..	DDDDDD..	....A..A	.....	.....
E....E..	.....	....AAAA	.....	.....
EEEEEE..	.....	.....	.....	.....
1	2	3	4	5

Now place them on top of one another starting with 1 at the bottom and ending up with 5 on top. If any part of a frame covers another it hides that part of the frame below. Viewing the stack of 5 frames we see the following.

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

In what order are the frames stacked from bottom to top? The answer is EDABC. Your problem is to determine the order in which the frames are stacked from bottom to top given a picture of the stacked frames. Here are the rules:

1. The width of the frame is always exactly 1 character and the sides are never shorter than 3 characters.
2. It is possible to see at least one part of each of the four sides of a frame. A corner shows two sides.
3. The frames will be lettered with capital letters, and no two frames will be assigned the same letter.

## INPUT DATA

Each input block contains the height,  $h$  ( $h \leq 30$ ) on the first line and the width  $w$  ( $w \leq 30$ ) on the second. A picture of the stacked frames is then given as  $h$  strings with  $w$  characters each.

Example input:

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

Your input may contain multiple blocks of the format described above, without any blank lines in between. All blocks in the input must be processed sequentially.

## OUTPUT DATA

Write the solution to the standard output. Give the letters of the frames in the order they were stacked from bottom to top. If there are multiple possibilities for an ordering, list all such possibilities in alphabetical order, each one on a separate line. There will always be at least one legal ordering for each input block. List the output for all blocks in the input sequentially, without any blank lines (not even between blocks).

Example Output:

```
EDABC
```

# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

## Problem 3 – Green Balloon Channel Allocation

When a radio station is broadcasting over a very large area, repeaters are used to retransmit the signal so that every receiver has a strong signal. However, the channels used by each repeater must be carefully chosen so that nearby repeaters do not interfere with one another. This condition is satisfied if adjacent repeaters use different channels. Since the radio frequency spectrum is a precious resource, the number of channels required by a given network of repeaters should be minimised. You have to write a program that reads in a description of a repeater network and determines the minimum number of channels required.

### INPUT

The input consists of a number of maps of repeater networks. Each map begins with a line containing the number of repeaters. This is between 1 and 26, and the repeaters are referred to by consecutive upper-case letters of the alphabet starting with A. For example, ten repeaters would have the names A,B,C,...,I and J. A network with zero repeaters indicates the end of input.

Following the number of repeaters is a list of adjacency relationships. Each line has the form:

```
A : BCDH
```

which indicates that the repeaters B, C, D and H are adjacent to the repeater A. The first line describes those adjacent to repeater A, the second those adjacent to B, and so on for all of the repeaters. If a repeater is not adjacent to any other, its line has the form

```
A :
```

The repeaters are listed in alphabetical order.

Note that the adjacency is a symmetric relationship; if A is adjacent to B, then B is necessarily adjacent to A. Also, since the repeaters lie in a plane, the graph formed by connecting adjacent repeaters does not have any line segments that cross.

## OUTPUT

For each map (except the final one with no repeaters), print a line containing the minimum number of channels needed so that no adjacent channels interfere. The sample output shows the format of this line. Take care that channels is in the singular form when only one channel is required.

## SAMPLE INPUT

```
2
A:
B:
4
A:BC
B:ACD
C:ABD
D:BC
4
A:BCD
B:ACD
C:ABD
D:ABC
0
```

## SAMPLE OUTPUT

```
1 channel needed.
3 channels needed.
4 channels needed.
```

# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

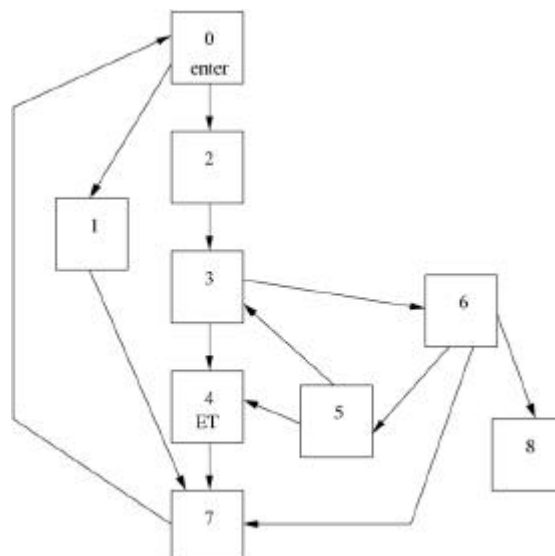
## Problem 4 – Blue Balloon Alien Security

You are in charge of security at a top-secret government research facility. Recently your government has captured a live extra-terrestrial (ET) life form, and is hosting an open day for fellow researchers. Of course, not all the guests can be trusted, so they are assigned different security clearance levels. Only guests with a level 5 rating will be allowed into the lab where the extra-terrestrial is being held; other than that, everyone is free to roam throughout the rest of the facility. Each room in the facility is connected via one-way airlocks, so that you can pass through the door in only one direction.

To protect your precious ET you will put in place enhanced security measures (in the form of armed guards) on the route leading to the room containing the ET, but not in the room itself – the guards do not have sufficient clearance to enter the room containing the ET. The guards will check the identity and the security rating of all guests trying to pass through the room in which they are stationed, so you would like to place the guards where they will cause the minimum amount of irritation to the guests who have no intention of visiting the ET. The room where the guards must be placed thus satisfies the following two conditions:

1. In order to get to the room containing the ET, the guests must pass through the room containing the guards;
2. There is no other room with this property that is closer to the room containing the ET – remember, the guards cannot be placed in the room containing the ET itself.

The diagram below illustrates one possible map of your facility:





Note that placing the guards in room 2 would satisfy the first condition, but room 3 is closer to the ET, so the guards must be placed in room 3.

All guests enter through room 0, the entrance to your facility. Your program accepts a sequence of lines containing integers. The first line consists of two integers: the number of rooms, and the room in which the ET is being held (out of his own free will, of course). The rest of the input is a sequence of lines consisting of only two integers, specifying where the airlock-doors are located. The first number on these lines specifies the source room, and the second the destination room. Remember: you can pass only from the source room to the destination room.

The output of your program consists only of a single line:

```
Put guards in room N.
```

where N is the room you've decided to place the guards.

### **SAMPLE INPUT**

This input sequence specifies the map shown above.

```
9 4
0 2
2 3
3 4
5 3
5 4
3 6
6 5
6 7
6 8
4 7
0 1
1 7
7 0
```

### **SAMPLE OUTPUT**

```
Put guards in room 3.
```

# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

## Problem 5 – Pink Balloon Octal Fractions

Fractions in octal (base 8) notation can be expressed exactly in decimal notation. For example, 0.75 in octal is 0.963125 ( $7/8 + 5/64$ ) in decimal. All octal numbers of  $n$  digits to the right of the octal point can be expressed in no more than  $3n$  decimal digits to the right of the decimal point.

Write a program to convert octal numerals between 0 and 1, inclusive, into equivalent decimal numerals. The input to your program will consist of octal numbers, one per line, to be converted. Each input number has the form  $0.d_1d_2d_3 \dots d_k$ , where the  $d_i$  are octal digits (0..7). There is no limit on  $k$ . Your output will consist of a sequence of lines of the form

$$0.d_1d_2d_3 \dots d_k [8] = 0.D_1D_2D_3 \dots D_m [10]$$

where the left side is the input (in octal), and the right hand side the decimal (base 10) equivalent. There must be no trailing zeros, *i.e.*  $D_m$  is not equal to 0.

### SAMPLE INPUT

```
0.75
0.0001
0.01234567
```

### SAMPLE OUTPUT

```
0.75 [8] = 0.953125 [10]
0.0001 [8] = 0.000244140625 [10]
0.01234567 [8] = 0.020408093929290771484375 [10]
```

# Third Southern African Regional ACM Collegiate Programming Competition

Sponsored by IBM

## Problem 6 – White Balloon Cracking the Code

You have been contracted by a terrorist group to crack encrypted transmissions. The only information that the terrorists could give you regarding the encrypted message is that a fixed key-length XOR-encryption algorithm was used to encode it. After a brief search on the 'Net, you find the following definition of XOR-encryption:

Assuming that we have a character  $u[i]$  from the unencrypted input stream, and a key  $k$  of length  $l$  with characters  $k[j]$ ,  $0 \leq j < l$ , then the encrypted value  $e[i]$  is obtained as follows:

$$e[i] = u[i] \text{ XOR } k[i \text{ MOD } l]$$

where MOD is the remainder after integer division (the % operator in C, C++ and Java), and XOR is the bitwise XOR operator applied to an 8-bit character (the ^ operator in C, C++ and Java). XOR encryption is a symmetric encryption scheme, so that the message is decoded by encrypting the encrypted message (a second time) with the same key, so that

$$u[i] = e[i] \text{ XOR } k[i \text{ MOD } l]$$

You are given an encrypted input stream of fewer than 30 000 characters. Your program must output the decrypted stream. The stream was encrypted using XOR encryption with a fixed length key of fewer than 30 characters. Each character in the key is unique (appears only once), and is selected from the set a..z merged with 0..9.

Your task is to determine the correct key length, and decrypt the encrypted input stream. Your terrorist friends provided you with one last vital piece of information: "The decrypted message will be in English."

It is recommended that you write an XOR encryption program first to aid you in testing your solution.

## SAMPLE INPUT

The output of the XOR encryption algorithm is not normally printable, since it may contain ASCII codes greater than 127. Therefore, the sample-encrypted message below is shown in numerical ASCII values (in decimal) – the actual input file contains the ASCII symbols. If the message "the quick brown fox jumps over the lazy dog" is encrypted using the key "12" (the literal characters "1" and "2", concatenated), the following (binary) file results.

```
69 90 84 18 64 71 88 81 90 18 83 64 94 69 95 18
87 93 73 18 91 71 92 66 66 18 94 68 84 64 17 70
89 87 17 94 80 72 72 18 85 93 86
```

If these values are converted to ASCII symbols and stored in a file (the file length should be exactly 43 bytes), it can be used as input to your program. It is recommended that you first write the encryption algorithm.

## SAMPLE OUTPUT

Your program should determine the key length to be 2 (you should *not* output this value), and decrypt the message to yield:

```
the quick brown fox jumps over the lazy dog
```

**A sample input text file will be provided**