UNIVERSITY OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Manitoba High School Programming Contest 2015
22 May 2015 12:45 – 3:30

Contest Rules:

- Do not open this package until instructed to do so.
- All solutions must be entered completely during the contest. No electronic copies of pre-written code are permitted.
- You may submit as many solutions as you like to each problem; however, incorrect solutions will be assessed a time penalty.
- Contest score is based on the most problems successfully solved; ties are won by shortest total time taken, including any penalties.
- A correct submission must solve the given problem and produce correct output for the given test data within a reasonable time.
- Programming style will not be considered during judging.
- Any programming language resources and notes are allowed.
- No other Internet access is allowed during the contest.

Submission Requirements / Pre-submission Checklist:

- All input must be read from standard input (in Java, `System.in`). Only Problem 1 does not require input. **Do not open input files.**
- All output must be written to standard output (in Java, use `System.out.print` or `println`).
- Your output format must follow the problem requirements **exactly**.
- Submit the source code file (`.java`, `.c`, etc.–**NOT** `.class`, `.exe`, etc.).
- Java programs must be in a single file and not placed in a package (no `package` statements – `import` statements are of course OK).
- Java programs must be complete; **they require a `main` method**.

Problem 1 – PIN possibilities

While setting up a PIN for a bank account, you are told "only digits 1-8 are allowed and the PIN must have exactly 10 digits". How many PINs of length 10 are there? Write a program that prints the number of PINs of length 1 to 10 if there are 8 different possibilities for each character in the PIN. For length 1, there are 8 possibilities (any digit from 1-8) and for length 2 there are 64 possibilities (8 possibilities for the first digit and 8 possibilities for the second digit).

Input

This problem does not have any input. The program will begin, write the output, and then quit.

Output

Output ten integer outputs, one on each line, representing the number of possible PINs of length 1-10.

Output
8 64 *(...eight more lines of output not shown...)*

Problem 2 – XYZ

Given three numbers x, y and z, where x is a single digit number (1-9), is it true that the number x appears y times in x*y*z?

For instance, if x = 7, y = 9 and z =12345679, then x*y*z = 777777777, so the answer is yes, since the number 7 appears 9 times in the answer. For x=3, y=4 and z=278, the answer is no, since x*y*z = 3336, and the number 3 only appears 3 times while y=4.

Write a program that takes the three integers x, y and z and prints out the product of the three numbers and then answers whether x*y*z has y appearances of the number x.

Input

Each case is on its own line. The three integers on the line are in the order x, y, z. In between the numbers is a single space. The number x satisfies $1 \le x \le 9$, while y and z satisfy $1 \le y, z \le 2^{31}-1$ and $x*y*z \le 2^{31}-1$.

The final line of input is 0 0 0. Do not process this line of input.

Output

For each case, start the input with "CASE x: " where x is an integer starting at 1, then write the product x*y*z and either the word "yes" or "no", depending on whether x appears y times in x*y*z* or not.

Sample Input	Sample Output
1 2 3 7 9 12345679 3 4 278 0 0 0	CASE 1: 6 no CASE 2: 777777777 yes CASE 3: 3336 no

Judging Data for Problem 2

```
1 1 1
3 1 4
2 2 55
1 5 9
3 3 367
7 2 55
1 5 202221
3 4 8253
4 4 2759
4 2 11706
7 9 821480
6 6 462963
6 8 34722222
6 8 34722221
3 4 8253421
7 7 1585873
9 9 12345679
6 6 185185
8 8 13888842
9 6 1685185
0 0 0
```

Problem 3 – A la Mode

Given a collection of numerical values, the mode is the value that appears most often. For instance, the mode of {1, 3 ,1, 5, 5, 7, 0, -1, 1} is 1 because it appears three times in the data.

Write a program that reads in lists of integers and finds the mode. In all the data sets, there is a unique mode: the value that appears most appears more times than every other value in the collection.

Input

Each list of integers is given on its own line. The line contains one or more integers, each separated from the next integer by a single space.

Output

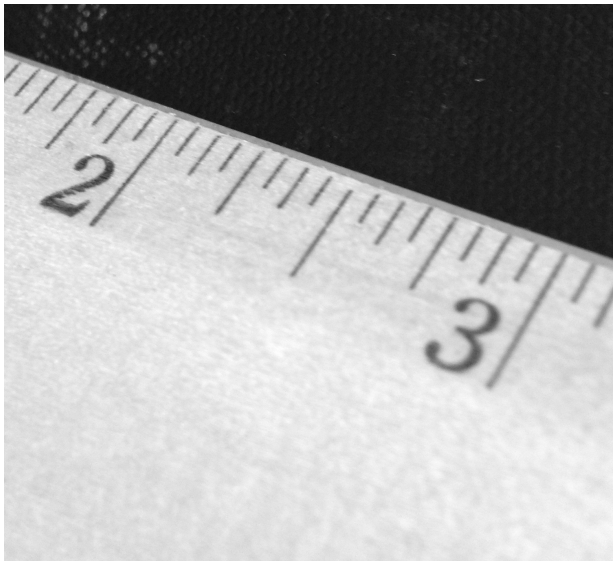
For each input, first output "CASE x: ", where x is an integer starting at 1. Then output the mode.

Sample Input	Sample Output
4 2 3 1 4 1 3 1 5 5 7 0 -1 1	CASE 1: 4 CASE 2: 1

Judging Data for Problem 3

```
10
10 10
10 10 10
10 2 10
10 100000 10
2 100000 2 10
2 2 2 3
10 2 3 3
10 10 10 10
10 10 10 2 2
2 2 2 10 2
2 2 2 2 2
4 2 3 20 20 5
10 3 10 5 5 7 0 -10 10
10 2 3 4 5 6 7 8 8
10 2 3 4 4 5 5 7 7 7
10 2 3 3 4 4 5 5 6 6 7 7 7
10 10 10 2 2 3 3 4 4 4 5 5 6 6 7 7
7 7 10 2 10 2 10 2 2 3 3 3 4 -10 2 5
```

Problem 4 – Rulers are meant to be broken



Have you ever looked at the scale of a ruler? I mean *really* looked at it? Most rulers have scales where the major divisions have the longest lines (inches in the picture) and smaller divisions (like half inches) have smaller lines. The smallest divisions (1/16th of an inch in the picture) have the smallest lines.

Write a program that draws divisions of a ruler for one unit using vertical bars (the character |, located above the return key on your keyboard). The smallest division would be represented by one vertical bar, the next smallest by two vertical bars, and so on up to the largest divisions, which happen at the start and For instance, for divisions up to 1/16th, the scale would look like this for one unit:

```
|||||||||||||||||
|   |   |   |   |   |   |   |   |
|       |       |       |       |
|               |               |
|                               |
```

Input

The input consists of several integers, each on their own line. Each integer represents one input. An input of n represents that the smallest division should be $1/2^n$. For example, if n=4, then the smallest division should be 1/16th, since $2^4 = 16$. The integers will be at least one and at most 20.

After the final input, the integer zero will appear on its own line. Do not process this input

Output

Begin the output for each line with a header line "=CASE X=" where X is the case number beginning with 1 for the first case. Include the two equals signs. Then output the ruler scale for one unit of measurement as described above.

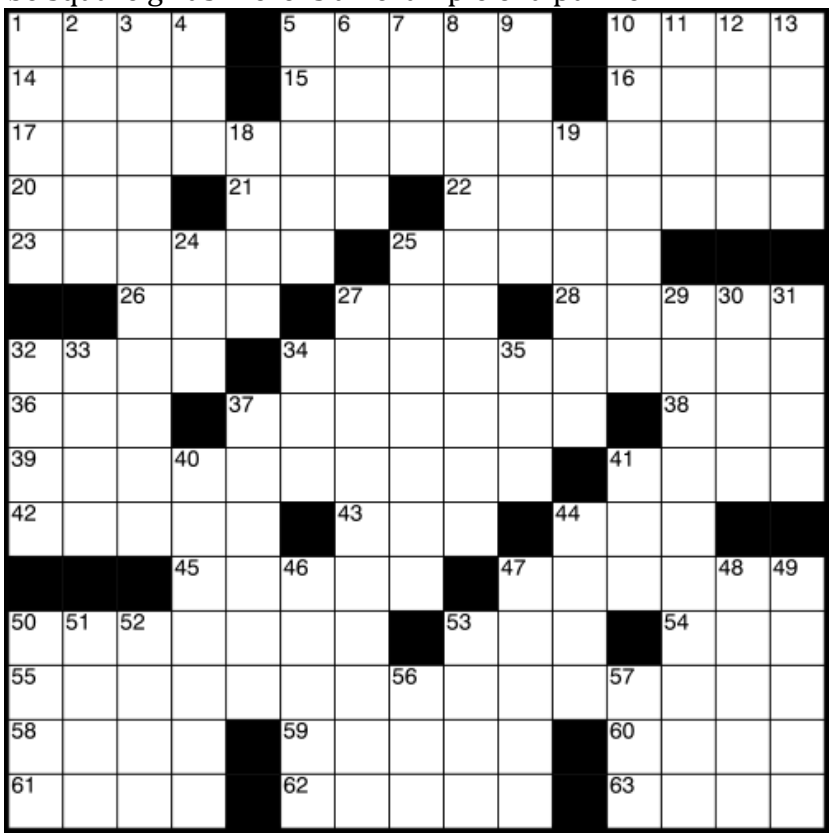
Sample Input	Sample Output
2 4 0	=CASE 1= ||||| | | | | | =CASE 2= |||||||||||||||| | | | | | | | | | | | | | | | | | | | |

Judging Data for Problem 4

```
2
8
5
6
7
10
4
9
```

Problem 5 – Alien Crossword Clues

Crossword puzzles are grids of white and black spaces where the contents are filled with letters so that horizontal and vertical sections correspond to words indicated by clues. The clues are numbered and written separately from the puzzle. All of the puzzles will be square grids. Here is an example of a puzzle:



Notice the small numbers in the corner of the cells only appear when a word in either the horizontal (left-to-right) or vertical (top-to-bottom) direction should start in that cell. For instance the number 1 indicates that a word of length four should start in the upper-left-hand corner of the puzzle and move to the right, as well as that a word of length five starting in the same corner and moving down. The number 4 (in the cell six spots from upper-left-hand corner in the top row) indicates that a word should start in both the horizontal and vertical directions.

The numbers start in the upper-left hand corner and proceed in row-major order (i.e., along the first row, then subsequent rows), always from left to right. If no clue starts at a position, then there is no number.

You have been hired by a crossword book publisher. The publisher has recently been awarded a lucrative contract to publish crosswords in several alien languages. The puzzles have been created, but you are responsible for printing out the numbers of the across and down clues in the puzzle. For example, in the above puzzle, the

down clues would be listed in the order 1,2,3,4,5… while the across clues would be listed in the order 1,5,10,14,15,16,… Note that clues can be both across and down clues.

A crossword puzzle will be written as a series of blanks (indicated by an underscore _), black squares (indicated by #) and clue numbers (arbitrary single characters from a-z,A-Z,0-9,and special characters !@$%^&*()[]{}).

Input

The first line of the input gives the number of puzzles to consider. Each puzzle is first described by a number n on a line, which describes the size of the puzzle, then n lines, each with n characters per line. The set of valid characters on a line are given above.

Output

For each puzzle, there will be three lines of output. On the first line, write "Case m:" where m is the case number, starting with one. Then on the next line, write the across clue numbers in order on a single line, starting with "ACROSS: ". On the final line of output for that case, write the down clue numbers in order on a single line, starting with "DOWN: ". Each clue number should be separated from each other on the same line by a single space.

Sample Input	Sample Output
2 5 01#23 4_#5_ #67_# 8___9 A_#B_ 2 12 3#	Case 1: ACROSS: 0 2 4 5 6 8 A B DOWN: 0 1 2 3 7 8 9 Case 2: ACROSS: 1 3 DOWN: 1 2

Judging Data for Problem 5

```
15
1
0
2
01
2_
2
0#
#1
2
0#
12
3
012
3__
4__
3
01#
2_3
4__
3
#01
2__
3__
4
0123
4___
5___
6___
4
0#1#
23_4
5___
6__#
4
012#
3_#4
5#6_
#7__
5
01#23
4_#5_
#67_#
8___9
A_#B_
```

(input continued on next page)

```
10
0123##4567
8___#9____
A___B_____
C#D_____#E
FG#H___#I_
J_K#L_#M__
N__O##P___
Q___RS____
T__________
U__________
10
#01234567#
8________9
A__________
B__________
C__________
D__________
E__________
F__________
G__________
#H_______#
15
01234#5678#9ABC
D____#E___#F___
G____H____#I___
##J____##KL____
MN_#O__PQ_______
R__S__#T____###
U____#V____#WXY
Z___#a____#b___
c__#d____#e____
###f____#g_____
hij_____k__#l__
m_____##n__o_##
p___#qrs_____tu
v___#w____#x___
y___#z____#!___
```

(input continued on next page)

```
15
01234#5678#9ABC
D____#E___#F___
G____H____#I___
##J____##KL____
MN_#O__PQ_______
R__S__#T____###
U____#V____#WXY
Z___#a____#b___
c__#d____#e____
###f____#g_____
hij_____k__#l__
m_____##n__o_##
p___#qrs_____tu
v___#w____#x_##
y___#z____#@_!$
```

Problem 6 – Freemium

Freemium games are *the worst*. That's not part of the problem, it's just an opinion. For those not familiar with them, freemium games are games where the game is free, but making progress through the game usually requires (real) money and often a lot of time, too.

In one freemium game, let's call it "Cash of Scams," players are required to spend in-game gold (purchased for real-world money) to build buildings in the game. The buildings also require time to build. However, in order to build some buildings, other buildings must also be present, which themselves need time and gold to be built. In this problem, we're concerned with the time to build buildings only.

Given a description of all buildings and their build times, write a program that finds the total amount time required to build a given building, called the "target" building. To calculate the total time, you can assume that as many buildings as possible can be built at the same time, and the total time is the longest time that would be required to build all buildings, including the target. Each target building will have at most two buildings that are necessary to be built before it (called the dependent buildings).

For example, consider a target building 4 and the following information:

1. To construct 4, you need to build buildings 2 and 3 and need 10 units of time.
2. To construct 3, you need twenty units of time
3. To construct 2, you need to build building 1 and 5 units of time
4. To construct 1, you need ten units of time.

With this information, we can conclude that it would take 30 units of time to build building 4: 10 units of time to build 4 + 20 units of time to build 3. During the 20 units of time to build 3, you can also build 1 and 2 as well, since they require 10+5 = 15 units of time.

On the other hand, if building 2 took 11 units of time and building 1 took 10 units of time, then the total build time would be 10+11+10=31 units.

Input

The input consists of several cases. Each case begins with an integer n on its own line, which represents the number of buildings that are involved in the construction of the target building. After the first line, there are **n** lines, each giving the information on one building. The information consists of an ID number (each building is numbered with an integer, and the building with ID number **n** is the target building), followed by a build time (an integer) and ending with a list of zero to two dependent buildings that must be constructed before this building. These buildings are listed by their ID numbers.

Buildings in a test case are listed with the target building last. For each building X in a test case, all buildings that are dependent on X are listed before X in the data.

The final test case is followed by the number 0 on its own line. Do not process this as an input case.

Output

For each case, output the total time required to build the building from scratch. Format your output as "Case #X: T" where X is a case number (starting at 1) and T is the total time.

Sample Input	Sample Output
4 1 10 2 5 1 3 20 4 10 2 3 4 1 10 2 11 1 3 20 4 10 2 3 0	Case #1: 30 Case #2: 31

Judging Data for Problem 6

```
1
1 100
2
1 100
2 100 1
3
1 100
2 100 1
3 100 2
3
1 100
2 101
3 100 1 2
4
1 10
2 5 1
3 20
4 10 2 3
4
1 10
2 11 1
3 20
4 10 2 3
5
1 20
2 10
3 10 1 2
4 10
5 10 3 4
7
1 10
2 20
3 30
4 40
5 10 1 2
6 10 3 4
7 10 5 6
7
1 40
2 20
3 30
4 10
5 10 1 2
6 10 3 4
7 10 5 6
```

(input continued on next page)

```
6
1 10
2 10
3 10 1
4 10 2 3
5 10
6 10 4 5
101
1 10
2 10 1
3 10 2
(lines of the form i 10 i-1 for i=4..98 deleted)
99 10 98
100 10 99
101 10 100
203
1 10
2 10 1
3 10 2
(lines of the form i 10 i-1 for i=4..98 deleted)
99 10 98
100 10 99
101 10 100
102 10
103 10 102
104 10 103
(lines of the form i 10 i-1 for i=105..199 deleted)
200 10 199
201 10 200
202 10 201
203 10 101 202
```

(input continued on next page)

```
63
1 26
2 8
3 47
4 69
5 72
6 55
7 87
8 74
9 57
10 99
11 78
12 38
13 41
14 0
15 23
16 13
17 100
18 39
19 92
20 44
21 78
22 29
23 91
24 1
25 63
26 101
27 7
28 36
29 59
30 40
31 64
32 78
33 4 1 2
34 56 3 4
35 7 5 6
36 88 7 8
37 80 9 10
38 87 11 12
39 83 13 14
40 47 15 16
41 30 17 18
42 62 19 20
43 72 21 22
44 82 23 24
45 27 25 26
46 15 27 28
47 9 29 30
48 70 31 32
49 65 33 34
50 81 35 36
51 0 37 38
52 83 39 40
53 9 41 42
54 88 43 44
55 45 45 46
56 3 47 48
57 28 49 50
58 52 51 52
59 28 53 54
60 45 55 56
61 48 57 58
62 87 59 60
63 19 61 62
0
```