Problem Set for the 1st Day of Summer Camp

Japanese Alumni Group

Contest Held on: 22 Sep 2006

This problem set contains three problems identified by A through C.

Problem B was newly created by the members of Japanese Alumni Group. You may use this problem in any form, entirely or in part, with or without modification, for any purposes, without prior or posterior consent to Japanese Alumni Group, provided that your use is made solely at your own risk.

Problem A was taken from ACM ICPC Northeast North America 2005 and Problem C from ACM ICPC Africa and Arab 2004. These problems were modified by the members of Japanese Alumni Group for clarification of the problem statement, adjustment of the difficulty level in solving the problems, attempt to hide the source of the problems from the contestants during the contest proper, and/or other purposes. The sources of problems were revealed to the contestants at the review session held after the contest proper. *The permission to use stated above is not applicable to these problems*.

Problem A
Monkey Business

Summer Camp (1st Day), Tokyo
22 Sep 2006

Input: A.txt

A technician has been training the monkeys used in the laboratory where she works to open and close doors. Next to the laboratory, there is a long hallway that contains a number of offices which initially have closed doors. The offices in the hallway are numbered starting at 1 and ending at N, where N is the number of offices in the hallway. Interestingly the number of monkeys in the laboratory is exactly equal to the number of doors in the hallway.

At night the technician lets the monkeys out of their cages one at a time for exercise. The first monkey that is let out of the cage runs down the hallway and opens every door (i.e., 1, 2, 3, 4, ...). The second monkey runs down the hallway and closes all of the even numbered doors (i.e., 2, 4, 6, 8, ...). The third monkey runs down the hallway and examines every third door (i.e., 3, 6, 9, 12, ...). The monkey will open any closed door that it examines, and close any open door that it examines. The fourth monkey examines every fourth door, and so forth. This process continues until all of the monkeys have been let out of their cages and allowed to run down the hallway. The last monkey will only examine the last door.

It would be nice to know which doors are left open at the end of the night. For example, given a hallway that contains five doors (and five monkeys), doors 1 and 4 will be open after all the monkeys have been let out of their cages. The technician has decided to write a grant to fund this work and has asked you for help. She requires you to write a program that, given as input the number of doors in a hallway, shows the open doors after all of the monkeys have been let out of cages. As said above, there are as many monkeys as there are doors, and all of the doors are initially closed.

Input

The input consists from a series of test cases. Each test case consists of a single positive integer which specifies the number of doors in the hallway. The input is terminated by a single zero. This is not part of the input.

Output

For each test case, you should output a list of numbers that identify the doors that are open after all monkeys have been let out of their cages. The numbers in the list should be output one per line in increasing order.

You should output a blank line between two test cases.

Sample Input

```
5
0
```

Output for the Sample Input

```
1
4
```

Problem B
The Closest Circle

Input: B.txt

Summer Camp (1st Day), Tokyo
22 Sep 2006

You are given N non-overlapping circles in xy-plane. The radius of each circle varies, but the radius of the largest circle is not double longer than that of the smallest.

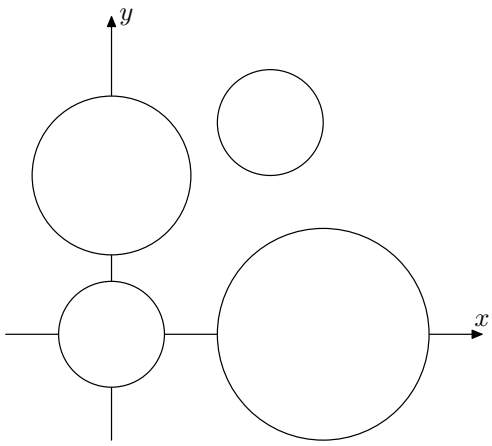

Figure 1: The Sample Input

The distance between two circles C_1 and C_2 is given by the usual formula

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} - r_1 - r_2$$

where (x_i, y_i) is the coordinates of the center of the circle C_i, and r_i is the radius of C_i, for $i = 1, 2$.

Your task is to write a program that finds the closest pair of circles and print their distance.

Input

The input consists of a series of test cases, followed by a single line only containing a single zero, which indicates the end of input.

Each test case begins with a line containing an integer N ($2 \leq N \leq 100000$), which indicates the number of circles in the test case. N lines describing the circles follow. Each of the N lines has three decimal numbers R, X, and Y. R represents the radius of the circle. X and Y represent the x- and y-coordinates of the center of the circle, respectively.

Output

For each test case, print the distance between the closest circles. You may print any number of digits after the decimal point, but the error must not exceed 0.00001.

Sample Input

```
4
1.0 0.0 0.0
1.5 0.0 3.0
2.0 4.0 0.0
1.0 3.0 4.0
0
```

Output for the Sample Input

```
0.5
```

Problem C
Sort that Queue

Input: C.txt

Summer Camp (1st Day), Tokyo
22 Sep 2006

A queue Q and two stacks A and B are given, and the following operations are defined for them:

`QA(`n`)` : dequeue an item from Q, pushing it on A; repeat n times
`QB(`n`)` : dequeue an item from Q, pushing it on B; repeat n times
`QQ(`n`)` : dequeue an item from Q, enqueue it again in Q; repeat n times
`AQ(`n`)` : pop an item from A, enqueue it in Q; repeat n times
`BQ(`n`)` : pop an item from B, enqueue it in Q; repeat n times
`AB(`n`)` : pop an item from A, push it on B; repeat n times
`BA(`n`)` : pop an item from B, push it on A; repeat n times

Note that each of the above is considered as a single operation, regardless of the value of n. Here, *dequeue* means retrieving the front item from a queue, and *enqueue* means adding a new item to the back of a queue. Both *push* and *pop* operations on a stack are done at the top of the stack.

Now, suppose that the queue is already populated with several numbers and that both of the stacks are empty, we would like to know the minimum number of operations needed to have the same numbers stored in the queue but sorted in an ascending order. For example, the queue (4 3 1 2 0) where 4 is at the front, can be sorted in three steps as follows: `QA(2)`, `QQ(2)`, then `AQ(2)`. The queue (5 4 1 3 2 0) can be sorted in four operations as follows: `QB(2)`, `QQ(1)`, `QB(2)`, `BQ(4)`.

Your task is to write a program that determines the minimum number of operations needed to sort the given queue.

Input

The input consists of a number of test cases. Each test case is specified on a single line. Each test case is made of $(N + 1)$ integers. The first integer specifies N which is the number of elements in the queue. A queue of N elements has the integers from 0 to $(N - 1)$ in some random order. The integers in the queue are specified from the front to the back. No queue has more than 10 elements.

The end of the test cases is identified with an input line that contains a single integer $N = 0$, which is not part of the test cases.

Output

For each test case, write the result on a separate line.

Sample Input

```
5 4 3 1 2 0
6 5 4 1 3 2 0
0
```

Output for the Sample Input

```
3
4
```