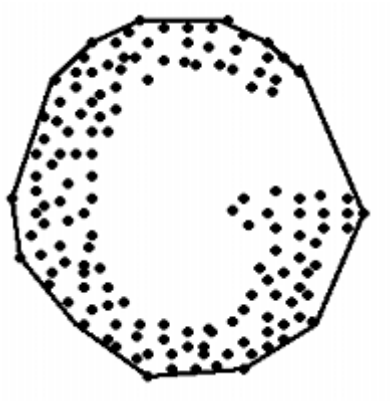
Day 2 Problem 1

Maple Syrup Roundup

Source file: maple.c *or* maple.cpp *or* maple.pas
Input file: maple.in
Output file: maple.out



An Elmira area maple syrup producer was selected as the winner of this year's CCC (Canadian Confectionary Competition) and the judge wants to place a blue ribbon around the sugar bush. To do this, she finds the most northerly tree (if there is more than one most northerly tree, any one will do) and stands at the position of that tree, facing due East. She then turns to the right until she is facing another tree, and walks to that tree in a straight line, measuring the distance. Once again she turns right until she faces a tree, and walks to it. At each step she chooses the tree that involves turning the least angle to the right, continuing until the starting tree is reached. The total distance travelled is the length of ribbon required. Your task is to calculate this length.

The input to the program will consist of a line containing an integer *m* followed by *m* data sets. Each data set consists of a line containing an integer $1 < n < 100$, the number of trees in the bush, and this is followed by *n* lines each with an ordered pair (x,y) of integers which give the location of a tree on the a cartesian grid. You may assume that the y axis points North while the x axis points East.

For each test case, the output from the program is the length of ribbon, to 2 decimal places, that can enclose every tree.

Sample Input

```
2
3
-1 1
1 1
1 -1
5
1 0
2 2
2 3
3 1
-1 2
```

Output for Sample Input

```
6.83
10.46
```

Day 2 Problem 2

Sum of Products

Source file: sum.c *or* sum.cpp *or* sum.pas
Input file: sum.in
Output file: sum.out

Mathematicians, as opposed to computer scientists, write expressions using single lower case letters as variables. Addition is represented by a plus sign and multiplication is represented by placing the variables or expressions adjacent with no symbol in between. Multiplication is done before addition unless the addition is parenthesized.

For this problem, we consider mathematical expressions consisting of only variables, addition, multiplication, and parentheses. (There are no other symbols, like spaces, division, subtraction, or numerals in the input or output.) For example,

```
a+b+c
xyz
xyz+ab+cd
(a+b)(c+d)e
```

Your task is to read in a number of expressions in this notation and to rearrange each, using the laws of algebra, to equivalent expressions with no parentheses. For example,

```
(a+b)(c+d)e
```

can be re-expressed as

```
ace+ade+bce+bde
```

If there are several solutions, any one will do. The order of variables within terms does not matter, nor does the order of terms within the expression. You do not need to collect like terms; indeed, this is impossible as there are to be no numerals in the output.

Input and Output Specification

The first line of input contains an integer *n*. *n* lines of input follow, each containing a valid expression as described above. No input line exceeds 100 characters. Your output should consist of *n* lines, each giving a parenthesis-free expression equivalent to the corresponding line of input.

Sample Input

```
5
a+b+c
(a+b)(c+d)e
c+cb+a+c
((a+b)+(c+d))e
(a+a)(a+a)
```

Possible Output for Sample Input

a+b+c
ace+ade+bce+bde
c+cb+a+c
ae+be+ce+de
aa+aa+aa+aa

Day 2 Problem 3

Fast Food

Source file: food.c *or* food.cpp *or* food.pas
Input file: food.in
Output file: food.out

The fast food chain, McBurger, has recently consolidated all activities to restaurants along the Trans Canada Highway. (Following the completion of fixed links joining Newfoundland and Vancouver Island to the mainland.) They have also decided to build several warehouses along the highway, each located at a restaurant and supplying those restaurants nearby. Naturally, these warehouses should be placed so as to minimize the distance between the warehouses and the restaurants they service. Your task is to write a program that determines the optimal positions for the warehouses.

To make the problem more precise, chairman McBoss has issued the following specification: You are given the positions of *n* restaurants across the country as *n* integers *d1 < d2 < ... < dn*. These integers are the distances, in kilometres, from the company's new headquarters in St. John's, at the extreme Eastern end of the highway. You are also given the number *k <= n*, the number of warehouses to be constructed. You are to place the warehouses at the locations of *k* of the *n* restaurants so as to minimize the maximum distance from any restaurant to its nearest warehouse.

Input Specification

The input file contains several test data sets. Each data set begins with the two integers *n <= 200* and *k*. Following this will be *n* integers *d1, d2, ... dn*. The number 0 will follow the last data set. Each integer will be on a separate line of input.

Output Specification

For each input data set, output three lines. The first line must contain k integers giving the locations of the warehouses, in kilometres from head office, in ascending order. If several sets of warehouse locations yield the same maximal distance, any one will do. The second line must contain the maximum distance from any restaurant to the nearest warehouse (this is the quantity that your program is to minimize). The third line must be blank.

Sample Input

```
6
3
5
6
12
19
20
27
0
```

Possible Output for Sample Input

6 20 27
6