4th Balkan Olympiad in Informatics
Nicosia, Cyprus, 19-25 October 1997

Day 2 - Problem 1 (Waiting for a ticket)

The U2 rock group will give a concert in Nicosia during November 96. A queue of n<=200 U2-fans waiting to buy a ticket from a single cashier has been formed. Each person wants to buy only one ticket, whereas the cashier can sell at most two tickets to a person.

Suppose that the cashier spends ti time units to serve the i-th fan (where 1 <= i <= n). However, two consecutive fans in the queue (e.g. the j-th and the (j+1)-th one (1 <= j <= n-1) may agree so that one of them remains in the queue and the other leaves the queue, if the time rj required by the cashier to serve the j-th and the (j+1)-th fan is smaller than $t_j + t_{j+1}$. Thus, in order to minimize the cashier's time and prevent the U2-fans from being squeezed, each person in the queue tries to negotiate with his predecessor and his successor, and finally speed up the process.

Given positive integer values for n, t_i and r_j, minimize the total cashier's time. This is achieved by making pairs of consecutive persons in an optimal way. Note, however, that it is not necessary for a specific fan to be paired with a predecessor or a successor.

- **Input:**
  Your program should read the input file named INPUT.TXT. Input data consists of three lines:
  - The first line contains the integer n
  - The second line contains n integers the t_i values separated by spaces
  - The third line contains n-1 values standing for the rj values separated by spaces

- **Output:**
  The first line of the file named OUTPUT.TXT contains an integer which represents the total cashier's time. Each one of the next lines contains either 1 number or 2 numbers separated by the character +. More specifically, each line contains the number i, if the i-th fan is served alone, or i+(i+1) if these two fans are served as a pair.

- **Example:**
  - *Input:*
    7
    5 4 3 2 1 4 4
    7 3 4 2 2 4
  - *Output:*
    14
    1
    2+3
    4+5
    6+7

Time Limit: 15 sec
Maximum Score : 35

4th Balkan Olympiad in Informatics
Nicosia, Cyprus, 19-25 October 1997

Day 2 - Problem 2 (Satellite configuration)

We want to launch a set of satellites in the sky in orbit around earth. Two satellites may transmit data to each other in a direct way, or in indirect way (i.e. via one or more intermediate satellites), or finally may have no connection at all. We say that two satellites are neighbors if they can send data to each other in a direct way.

Given a number n of satellites and a set of n integers, find out:

a. if there exists a configuration of n satellites such that the given set of n integers correspond to the neighbors of the satellites
b. if the answer in part (a) is positive, then describe such a setting,
c. given the setting of part (b), conclude if we can send data from any satellite to any other one.

- **Input:**
  The input file INPUT.TXT consists of a line containing a sequence of integers separated by spaces. The first integer, a number n <= 20, gives the number of satellites in orbit. The following n integers represent the required number of neighbors for each one of the satellites. Note that the order of these n numbers (which represent the number of neighbors for each satellite) is not important.

- **Output:**
  The first line of the output file OUTPUT.TXT contains the answer to part (a): a YES or a NO. In case of a NO in part (a), then no more output is required. In case of a YES in part (a), then a number of n lines follow containing n integers 1 or 0 separated by spaces, forming an nxn matrix. If the (i,j)-th element of this matrix is 1, then the satellites i and j are neighbors, whereas if this element is 0 then these satellites are not neighbors. The (n+2)-th line of the output file corresponds to part (c) and is a YES or a NO.

- **Example 1:**

  - *Input:*
    6 4 3 1 4 2 0
  - *Output:*
    NO

- **Example 2:**

  - *Input:*
    7 4 3 1 5 4 2 1
  - *Output:*
    YES
    0 1 1 1 1 1 0
    1 0 1 0 0 0 0
    1 1 0 1 0 1 0
    1 0 1 0 0 1 0
    1 0 0 0 0 0 0
    1 0 1 1 0 0 1
    0 0 0 0 0 1 0
    YES

- **Example 3:**

  - *Input:*
    6 2 3 1 1 2 1
  - *Output:*
    YES
    0 1 0 1 0 0
    1 0 1 1 0 0
    0 1 0 0 0 0
    1 1 0 0 0 0
    0 0 0 0 0 1
    0 0 0 0 1 0
    NO

Time Limit : 20
Maximum Score : 35

4th Balkan Olympiad in Informatics
Nicosia, Cyprus, 19-25 October 1997

Day 2 - Problem 3 (Windows in a graphical user interface)

A common operation in graphical environments is the handling of overlapping windows. Only the exposed portions of a window are displayed, with special routines written to determine what portions of a window are exposed, and should be dealt with. Your task is to take a list of windows, do the appropriate manipulations, and, upon request, indicate the exposed area of a given window.

- **Input:**
  The program reads the input file INPUT.TXT as a series of one instruction per line in the following format:
  w(I,X,Y,x,y)
  t(I)
  b(I)
  e(I)
  s(I)
  In the above input 'w', 't', 'b', 'e', 's' are the commands manipulating the windows. The rest of the information represent the following data:

  - I is a windows id. The windows identifiers will be single characters in the range:
    abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789-+ As the identifiers are unique, it will not be possible to have two windows with the same id, open at the same moment. There will be no more than 64 windows opened at any one time.
  - X and Y are the upper left-hand coordinates of the window.
  - x and y are the lower right-hand coordinates of the window. It is assumed that in order to perform an operation on a given window, this specific window must have been created before with the 'w' command.

  Explanation of window-commands :

  - 'w': creates a window with the appropriate id, size and location. A newly created window is created on top of all existing windows.
  - 't': brings the indicated window on the top of all other windows.
  - 'e': erases the indicated window from the screen.
  - 'b': puts the indicated window below all other windows.
  - 's': shows the percentage of indicated window's exposed area, printed as a real number followed by a new line.

- **Output:**
  The output file of the program OUTPUT.TXT will simply be responses to the 's' commands. For each 's' command found and processed in the input file, one output line is generated that shows in a percentage the exposed area of the indicated window.

- **Example:**

  - *Input:*
    w(a,10,132,20,12)
    w(c,12,120,22,16)
    w(b,8,16,124,15)
    t(a)
    w(d,18,93,102,20)
    b(b)
    b(a)
    s(a)
    s(b)
    s(c)
    s(d)
    e(d)
    e(c)
    s(a)
    s(b)
  - *Output:*
    29.83%
    100.00%
    71.92%
    100.00%

99.17%
100.00%

- **Limits**
  The screen that all windows will be displayed on will not exceed 250x250. The thickness of windows boundaries is negligible (i.e. w(a,0,4,4,0) is a window of area 16).

- **Note:**
  Correct output values will be considered those with ρ0.02% deviation from the indicate value in the output (i.e. for 29.83% the percentages 29.81%, 29.82%, 29.84% and 29.85% will be consider correct).

Time limit per test : 5 seconds
Maximum Score : 30
