South African Computer Olympiad
Final round
Day 1

Overview

Author	**Marco Gallotta**	**Bruce Merry**	**Graham Poulter**
Problem	**honey**	**bursary**	**babylon**
Source	honey.pas honey.c honey.cpp honey.java	bursary.pas bursary.c bursary.cpp bursary.java	babylon.pas babylon.c babylon.cpp babylon.java
Input file	honey.in	bursary.in	babylon.in
Output file	honey.out	bursary.out	babylon.out
Time limit	2 seconds	1 second	5 seconds
Number of tests	10	10	10
Points per test	10	10	10
Total points	**100**	**100**	**100**

The maximum total score is 300 points.

Honeycomb

Author

Marco Gallotta

Introduction

Figure 1 shows a honeycomb of numbers. A route starts from some node in the uppermost row and ends in some node in the lowest row. From a node, the route can continue only diagonally down to the left or down to the right. When creating a route through the honeycomb, you are allowed to make at most one swap of two numbers on at most one horizontal row of the honeycomb. Swapping essentially means that in one chosen row you are allowed to place the greatest number of that row to any position on the same row.

Task

Your task is to write a program that calculates the highest sum of numbers on any route using the ability of swapping two numbers on a chosen row.

Example

For the honeycomb in the diagram above, the highest sum ($3+2+8+5+4=22$) is shaded in gray. Notice that the number 5 on the fourth row (from the top) is swapped to the 3rd position (from the left) on that row.

Input (honey.in)

The first line contains a single integer N, the side length of the honeycomb. The next $2N-1$ lines give the numbers (space separated) for each row of the honeycomb, from top to bottom and left to right.

Sample input

```
3
1 2 3
3 2 2 1
4 2 8 0 3
5 3 1 2
3 1 4
```

Output (honey.out)

The only line of the output file should contain a single integer, the highest sum, calculated as described above.

Sample output

```
22
```

Constraints

- $1 \leq N \leq 500$
- $0 \leq$ each node ≤ 1000

50% constraints

- $1 \leq N \leq 15$
- $0 \leq$ each node ≤ 200

Time limit

2 seconds.

Scoring

You will score 0% for an incorrect answer and 100% for a correct answer.

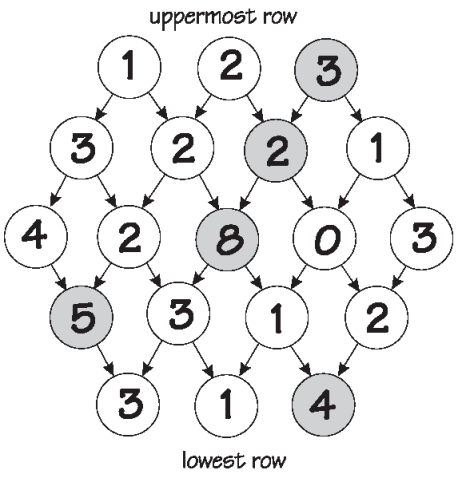

Figure 1: Honeycomb

South African Computer Olympiad
Final round
Day 1

Bursary

Author

Bruce Merry

Introduction

Fred the manic store-keeper has decided to go to university to study for a marketing degree, so that he can improve sales in his shop. He has been offered numerous bursaries, each of which limits how much he can receive from other bursaries. Each also limits the amount of time he can spend on part-time work (running the shop), and thus how much extra money he can make. Help him to decide which bursaries he should accept to obtain the maximum amount of money.

Task

Each bursary has the following properties: the value v_i, the maximum total value m_i and the maximum amount that Fred can make doing part-time work, t_i. These values are all expressed in whole numbers of rands.

The amount of money that Fred will receive is computed as follows. First, he adds up v_i for each bursary he accepts, to obtain a total value V. The maximum total M is the minimum of m_i for each bursary he accepts. Fred will actually receive the minimum of V and M from the bursaries. Finally, he will make an additional T from part-time work, where T is the minimum t_i for the bursaries he accepts.

To summarise:

$$\begin{aligned} V &= \sum v_i \\ M &= \min\{m_i\} \\ T &= \min\{t_i\} \\ \text{total} &= \min\{V, M\} + T. \end{aligned}$$

Fred must accept at least one bursary. This ensures that T is always well defined.

Example

Consider the bursaries listed in table 1.

If Fred takes only one bursary, he will make at most R70 ($50+20$ or $60+10$). If he takes all three, he makes a total of R75 ($65+10$). But if he takes the first and third, he will make R85 ($65+20$).

i	v_i	m_i	t_i
1	50	80	20
2	60	70	10
3	30	65	20

Table 1: Example bursaries

Input (bursary.in)

The first line of input contains the N, the number of bursaries on offer. The following N lines each describe a bursary. Each line contains the three integers v_i, m_i and t_i, separated by single spaces.

Sample input

```
3
50 80 20
60 70 10
30 65 20
```

Output (bursary.out)

The first line of output must contain K, the number of bursaries that Fred should accept to obtain the maximum amount of money. The following K lines list the numbers of the bursaries that Fred will accept, one per line. The bursaries are numbered from 1 to N, in the order they appear in the input file. The bursaries may be listed in any order.

If there is more than one way in which Fred can obtain the maximum, you are only required to output one.

Sample output

```
2
1
3
```

Constraints

- $1 \le N \le 2000$
- For each bursary i:
  - $1 \le v_i \le m_i \le 100000$
  - $0 \le t_i \le 100000$

50% constraints

- $1 \le N \le 20$
- Other constraints are as above.

South African Computer Olympiad
Final round
Day 1

Time limit

1 second.

Scoring

If your output file is invalid (for example, you accept a bursary twice, or have negative numbers) you will score 0. If you obtain the maximum amount of money, you will score 100%. Otherwise, you will score $\frac{P}{Q} \times 50\%$ rounded down, where P is your total and Q is the optimal total.

Tower of Babylon

Author

Graham Poulter

Introduction

The Babylonians are famous for the Hanging Gardens and the Tower of Babylon. According to legend, the tower was meant to reach the sky, but the project failed because of a confusion of language imposed from somewhat higher. For the 2638th anniversary a model of the tower is being rebuilt from large precast blocks.

Task

n different types of blocks are available, each in unlimited supply. Each type is characterised by its three dimensions, x, y and z.

The blocks are to be stacked one upon eachother so that the resulting tower is as high as possible. Of course the blocks can be rotated as desired before stacking. However, for reasons of stability, a block can only be stacked upon another if both of its baselines are shorter, as illustrated in Figure 2.

Example

Suppose there is only one block type, with dimensions (5,4,3). You can place the first block any way you wish.

Supposing you place the first block with base dimensions of 5×4 and height 3, then you can place the second block using base dimension 4×3 and height 5.

However, the second block could not have had a base of 3×4, 5×3 or 3×5, because then one of the edges would be as long or longer as corresponding one below it.

Input (babylon.in)

The number of types of blocks n is located in the first line of the input file. On the subsequent n lines the dimensions x, y and z of each type of block are given.

Sample input

```
5
31 41 59
26 53 58
97 93 23
84 62 64
33 83 27
```

Output (babylon.out)

Your program should output the height h of the highest possible tower.

Sample output

```
342
```

Constraints

- $1 \leq n \leq 30$
- $1 \leq x, y, z \leq 10000$
- $1 \leq h \leq 1000000$

50% constraints

- $1 \leq n \leq 10$

Time limit

5 seconds.

Scoring

You receive 10 points for a correct answer, and 0 points for an incorrect answer.

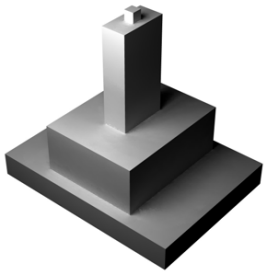

Figure 2: Tower of Babylon