Richi Pichichi S.A. P92025_en

Richi Pichichi S.A. is a company that sells and purchases shares: receives, in real time, petitions to *sell* and to *purchase* of shares, attached with the price that the customers are prepared to do the operation. If, at any time, Richi Pichichi S.A. discovers that has a customer that wants to sell n_s shares at a sale price p_s, and other customer that wants to purchase n_p shares at a purchase price p_p greater or equal than the sale price, then does the operation: exactly $n = \min(n_s, n_p)$ shares change of hands at the price of $n \cdot (p_c + p_v)/2$, rounding towards zero (besides of a minimal, economic comission, that does not matter). After an operation of purchase/sale, one of the two orders dissapears from the Richi Pichichi S.A. database, while the purchase or sale petition of the other one is decreased in n units.
If it was possible to match various purchasers with sellers, Richi Pichichi S.A. will *always* start matching the purchaser that is prepared to purchase at the greatest price with the seller that accepts to sell at the least price. In a event of a tie, the order that is most time in the Richi Pichichi S.A. database will have preference. An order that has been partially satisfied does not lose age.
You are asked to, given a sequence of purchase and sale orders, indicate the operations that Richi Pichichi S.A. will do.

Input

Each test data contains a sequence of purchase and sale orders, each one of them occupying a line. An order is given by a character `P` (purchase order) or `S` (sale order), a number a that identifies the type of action (from 1 to 1000), the price p and the quantity of shares n, separated by spaces.

Output

Each time that is possible to do an operation, your program must print the number of line in what the purchase order was given, the number of line in what the sale order was given, the type of action, the number of shares sold and the total cost of the operation. Follow the format of the instances.

Sample input 1

```
S 666 100 1
P 666 101 5
S 666 97 1
S 666 99 1
S 666 96 10
P 666 99 1
P 666 98 1
P 666 96 1
P 666 94 10
S 666 96 10
P 666 100 50
```

Sample output 1

```
1 #666 = 100 (1->2)
1 #666 = 99 (3->2)
1 #666 = 100 (4->2)
2 #666 = 197 (5->2)
1 #666 = 97 (5->6)
1 #666 = 97 (5->7)
1 #666 = 96 (5->8)
5 #666 = 490 (5->11)
10 #666 = 980 (10->11)
```

Sample input 2

```
P 333 1001 1
```

```
P 333 1000 1
P 333 1000 1
P 333 1001 1
```

```
P 333 1000 1
P 333 1001 1
S 333 1000 10
P 333 1000 1
P 333 1001 1
P 333 1000 1
P 333 1001 1
```

Sample output 2

```
1 #333 = 1000 (7->1)
1 #333 = 1000 (7->4)
1 #333 = 1000 (7->6)
1 #333 = 1000 (7->2)
1 #333 = 1000 (7->3)
1 #333 = 1000 (7->5)
1 #333 = 1000 (7->8)
1 #333 = 1000 (7->9)
1 #333 = 1000 (7->10)
1 #333 = 1000 (7->11)
```

Author: Omer Giménez

Problem information

Author: Omer Giménez
Translator: Carlos Molina

Generation: 2026-01-25T11:56:33.500Z