Problem I. Iceberg Orders

Input file: `iceberg.in`
Output file: `iceberg.out`

You are working for Metagonia stock exchange. Recently traders in Metagonia had heard about Iceberg orders traded on London stock exchange and asked your employer to add such functionality as well. A *stock exchange* is an engine that receives *orders* and generates *trades*.

An *iceberg order* is a quintuple of integers (ID, T, P, V, TV). Each order has an identifier ID (unique among all orders), type T (which is equal to either $BUY = 1$ or $SELL = 2$), price P, total remaining volume V and tip volume TV. For each order, exchange additionally keeps track of its current volume CV and priority PR. There is also a global priority counter of the exchange GP. An *order book* of the exchange is a set of orders.

Trades that are generated by the exchange are quadruples of integers (BUY_ID, $SELL\_ID$, P, V). Each trade has BUY_ID and $SELL\_ID$ — identifiers of matching BUY and $SELL$ orders, trade price P, and trade volume V.

When an order is received by the exchange it is matched against orders currently on the order book. This is done as follows. Suppose an order a is received with $T_a = SELL$. Among all orders currently on the order book we look for an order b such that $T_b = BUY$ and $P_b \geq P_a$. We select such an order b with the largest price, and if there are several — one with the smallest priority. If there is such an order b, then a trade t is generated with $BUY\_ID_t = ID_b$ and $SELL\_ID_t = ID_a$ at trade price $P_t = P_b$ with trade volume $V_t = \min(V_a, CV_b)$. V_a, V_b, and CV_b are all decreased by trade volume. If $V_b = 0$ after this, then the order b is removed from the order book. If $CV_b = 0$ (but $V_b > 0$) then we set current volume of order b to $CV_b = \min(V_b, TV_b)$, set $PR_b = GP$, and increment GP. We continue these operations of selecting b and generating trades until either $V_a = 0$ or there are no more orders b on the order book which satisfy the condition. In the latter case, we add order a to the order book with $CV_a = \min(V_a, TV_a)$ and $PR_a = GP$, and then increment GP. When the process of matching the order a is finished with several trades between the same pair of orders a and b (and there can be lots of them!), they are all united into a single trade with the volume equal to the sum of individual trade volumes.

If $T_a = BUY$ we are looking for an order b with $T_b = SELL$ and $P_b \leq P_a$ and select such an order b with the smallest price and the smallest priority among those. The rest of the matching process is as described above, with trades having $BUY\_ID_t = ID_a$, $SELL\_ID_t = ID_b$, $P_t = P_b$, and $V_t = \min(V_a, CV_b)$.

Initially the order book is empty. You are presented with several orders that are received by the exchange one by one. You need to print generated trades and the order book state after all orders are processed.

Hint: The priority and GP are introduced in the problem statement only for the purpose of a formal description of the algorithm. The actual implementation does not have to keep track of priority. Typically, an exchange simply keeps a priority-ordered list of orders of each type at each price in its order book.

Input

The first line of the input contains the number of orders n ($1 \leq n \leq 50\,000$). Each of the following n lines represent an order. Each order is given by a space-separated quintuple $ID\ T\ P\ V\ TV$, where $1 \leq ID \leq 1\,000\,000$, $T = 1$ for BUY and $T = 2$ for $SELL$, $1 \leq P \leq 100\,000$ and $1 \leq TV \leq V \leq 1\,000\,000\,000$.

Output

For each order print all trades generated by processing this order, in ascending order of pairs (BUY_ID, $SELL\_ID$), each trade on its own line. Each trade shall be printed as a space-separated quadruple of integers $BUY\_ID\ SELL\_ID\ P\ V$. It is guaranteed that the total number of trades would not exceed 100 000. Print a blank line after all trades, followed by the order book. Each order that is still on the book shall be printed as a space-separated **sextuple** $ID\ T\ P\ V\ TV\ CV$, sorted first by P and then by PR.

Sample input and output

iceberg.in	iceberg.out
7 42 1 100 200 20 239 1 100 50 50 1111 1 101 30 15 1234 1 100 300 15 4321 2 99 125 25 5678 1 101 30 30 8765 2 101 100 20	42 4321 100 30 239 4321 100 50 1111 4321 101 30 1234 4321 100 15 5678 8765 101 30 42 1 100 170 20 10 1234 1 100 285 15 15 8765 2 101 70 20 20

In the sample input the first four orders have $T = BUY$. Assuming that at the beginning GP at the exchange was equal to 1, after receiving these orders the order book looks in the following way when ordered according to the matching rules from the problem statement for $T_b = BUY$ orders (first by the largest price, then by the smallest priority):

ID	T	P	V	TV	CV	PR
1111	1	101	30	15	15	3
42	1	100	200	20	20	1
239	1	100	50	50	50	2
1234	1	100	300	15	15	4

The fifth order (with $ID_a = 4321$) has $T_a = SELL$, $P_a = 99$, and $V_a = 125$ and is eligible for match with all of the above four orders given in the above table. First, it matches twice with the order 1111 with the highest price of 101 for a total trade volume of 30, removes it from the order book, bumps GP to 6 in the process, and decreases V_a to 95. Then, there are three other orders at price 100. One matching pass through them produces three trades for a total volume of 85 (volume 20 with order 42, volume 50 with order 239, removes it from the book, volume 15 with order 1234), bumps GP to 8, and decreases V_a to 10. The remaining orders in the book are shown below:

ID	T	P	V	TV	CV	PR
42	1	100	180	20	20	6
1234	1	100	275	15	15	7

One last match with the order 42 produces a trade with a volume of 10 (for a total volume of 30 for matches with order 42) and the order 4321 is done ($V_a = 0$). Four corresponding total trades for order 4321 are printed in the sample output. The remaining order book is:

ID	T	P	V	TV	CV	PR
42	1	100	180	20	10	6
1234	1	100	275	15	15	7

The sixth BUY order (with $ID = 5678$) is added to the order book (GP becomes 9):

ID	T	P	V	TV	CV	PR
5678	1	101	30	30	30	8
42	1	100	180	20	10	6
1234	1	100	275	15	15	7

The last, seventh order (with $ID = 8765$), can be matched only with the order 5678 due to price condition, generates a trade with volume of 30, order 5678 is removed from the order book, while order 8765 is added. Now the order book has both BUY and $SELL$ orders:

ID	T	P	V	TV	CV	PR
42	1	100	180	20	10	6
1234	1	100	275	15	15	7
8765	2	101	70	20	20	9