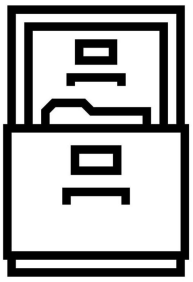
Data File Storage



You are working for a cloud data-storage company called Stork. The company owns some number of servers n and these servers are ordered from oldest to newest. Over time, requests come in to store new data files, to clear space for new data, or to query the amount of data across ranges of servers. More formally, here are the requirements of the problem:

1. There are n servers, ordered from index 0 (oldest) to index $n-1$ (newest). Each server has a storage capacity C.
2. When a new storage request comes, the size of the file f is provided as well as the index i of the requested server. The file should be stored on server i. If the data cannot be fully stored on server i (due to the server being raised to capacity), then store the most (integral) amount of data you can and move the rest to server $i+1$ (continuing to newer and newer servers as you go). However, Stork does not want any storage requests to involve more than five servers. So, try to store the data on servers i, then $i+1$, through $i+4$ (or server $n-1$ if that comes earlier) and then stop (even if there is still more data left to store).
3. Deletions are strange, Stork wants you to notice when the total storage of the oldest servers has gotten quite large and clear space from the newest server among them (confusing, we know). To model this, when a deletion request comes in, you will be given an amount of data d to delete and a threshold value t. You should find the oldest server i such that the sum of the data stored in servers 0 through i is at or above the threhold t. Remove d data from this server i. If there is less than d data on this server than remove as much as you can and stop. If the threshold t is larger than the sum of the data stored in all servers, you should ignore the deletion request and not remove any data.
4. At any time, Stork may send a query of the form l, r. These queries should return the total amount of data stored on server l through server r inclusive at that time.

Implement a program that given a stream of file storage requests, deletion requests, and query requests, prints the correct answer to each query as defined by the requirements above.

Input

The first line of input will contain the number of servers $n \leq 10^5$, the capacity $C \leq 10000$ and the number of commands $P \leq 50000$. The next line will contain exactly n space separated integers: the amount of data already on each of the n servers before adding any more. The following P lines will each contain a command in one of the following three formats:

- the letter F followed by the size $f \in \mathbb{N}$ and index of the server $0 \leq i \leq n-1$. For example, F 1245 10 means to store a file of size 1245 on server 10 (and overflow to server 11, etc. if necessary). If you run out of space to insert the data, insert as much as possible up through server $n-1$ or $i+4$ (whichever is smaller) and stop.

- the letter D followed by the amount of data $d \in \mathbb{N} \leq C$ to delete and the threshold value $t \leq 10^6$. For example, D 50 300 means that we need to delete 50 units of data. We use the number $t = 300$ to find the index of the oldest server such that the sum of data stored from server 0 through i is 300 or more. Remove up to 50 units of data from this server and stop.
- Q (for query) and the integers l and r such that $l, r \in \mathbb{N}, 0 \leq l \leq r \leq n-1$. For example, Q 4 10 is requesting the total data storage over servers 4 through 10 (inclusive).

Output

For each query request in the input, print the left and right indices of the query along with the total file storage across servers l through r at the time of the query. Use the following format: (l,r): f (see examples below).

Sample Input

```
10 10 12
5 5 5 5 5 5 5 5 5 5
F 20 5
F 4 2
Q 2 3
Q 6 8
F 100 0
Q 1 5
D 8 30
Q 2 2
D 15 20
Q 1 1
Q 1 2
Q 0 2
```

Sample Output

```
(2,3):  14
(6,8):  30
(1,5):  50
(2,2):  2
(1,1):  0
(1,2):  2
(0,2):  12
```