UVa Online Judge

11423 Cache Simulator

Cache is a special type of memory providing very small latency for accessing. The basic idea of using cache is to keep recently used data close to processors. So, modern processors use small cache along with large main memory. Program performance depends on the cache performance. If data is found in cache (Hit), execution can proceed normally. On the other hand if data is not found in cache, execution stalls until that data is fetched from main memory. Computer architects always looks for miss rate of cache (# of misses per 100 accesses).

When a miss occurs, the new data is kept in the cache and some existing data is removed from cache. There are several policies to select which one to evict. A common technique used in any computer systems is LRU (Least Recently Used). According to this scheme, the data that has not been used for longest time is evicted. Let us explain how this scheme works

Assume a cache with 4 entries. So, if the processor accesses data 1, 2, 3, 4, 5, 2, 3, 99, 2, 5, 4 the following things will occur

Cache	Hit/Miss	Data evicted
	Miss (1)	
1	Miss (2)	
1 2	Miss (3)	
1 2 3	Miss (4)	
1 2 3 4	Miss (5)	1
2 3 4 5	Hit (2)	
2 3 4 5	Hit (3)	
2 3 4 5	Miss (99)	4
2 3 5 99	Hit (2)	
2 3 5 99	Hit (5)	
2 3 5 99	Miss (4)	3
2 4 5 99		

Here for 11 data references, we have 7 misses. In this problem you have to find the number of misses for different cache sizes. You will be given data references in several ways-

a) `ADDR` x: Processor will access data x

b) `RANGE` b y n: Processor will access all data references in the form $b + y * k$, where k is 0 to $n - 1$

Your will also be given commands `STAT` to print the number of misses for each cache (excluding the misses occurred before last `STAT` command). The above example can be abstracted in the following way

```
RANGE 1 1 5
RANGE 2 1 2
ADDR   99
STAT              ---------------------------------- 6
ADDR   2
RANGE 5 -1 2
STAT              ---------------------------------- 1
```

Input

Your program starts with a positive integer N ($1 \leq N \leq 30$), which is the number of caches. The next line contains size of each cache (in increasing order). Cache size will be at least 2 but not exceeding 2^{20}. The next few lines will contain any of '`RANGE`', '`ADDR`', or '`STAT`'. The last line will contain '`STAT`'. Total number of data references (all data accessed by processor) will not exceed 10^7. For the sample input, total number of data references is 30 (5+2+1+1+2+10+5+4). Also, the processor uses 24-bit address, no data reference will exceed $2^{24} - 1$ and will be non-negative. The value of b, y, n (in '`RANGE`'), and x (in '`ADDR`') will be consistent with all restrictions. Input is terminated by a line containing '`END`'. The input file has around 20000 lines of inputs.

Output

For each `STAT` command just print the numbers of misses (as described above) in a line.

Sample Input

```
2
4 8
RANGE 1 1 5
RANGE 2 1 2
ADDR   99
STAT
ADDR   2
RANGE 5 -1 2
STAT
RANGE 0 10000 10
RANGE 0 20000 5
RANGE 0 30000 4
STAT
END
```

Sample Output

```
6 6
1 0
18 13
```