CS 213 (M) - Interesting Problem

In this problem, you have to simulate memory allocation in a system. There are n programs and the system has a memory of size M, consisting of locations with address 0 to M-1.
A program may request to be allocated memory of size m, and you have to allocate a set of m consecutive locations that are currently not allocated to any program. If they are available, you should find the starting address of such a set of m locations, if not, the request should be kept pending.
A program may also request deallocation of memory that was previously allocated to it. The starting address of the memory to be deallocated will be given. You have to check that some amount of memory with that starting address was allocated to the program previously, and deallocate it. If the address is incorrect, you should indicate an error. A program may also terminate, in which case all memory allocated to it is deallocated.

Whenever memory is freed, you have to check if any pending requests for memory can now be allocated, and if so do the allocation in the order in which the requests were made. Note that when a program terminates, all memory allocated to it must be freed, before considering pending requests.

Different strategies are used for deciding the locations to allocate. The simplest one is called the first-fit strategy, in which the smallest address such that there exist required number of consecutive unallocated locations starting from it, is allocated. Another is called the best fit strategy, in which the starting address of a block of consecutive unallocated locations of minimum size >= required number is allocated.

For this question, all you need to do is implement the first fit strategy.

Input Format
The first line of input specifies n, the number of programs, and M the number of memory locations. Assume 1 <= n <= 1000 and 1 <= M <= 10^9.
The subsequent lines contain a description of the requests to the system.
Each request is of one of the following types:

A PID size
D PID address
T PID
H

size and address are numbers, having the following meaning
size: number of additional memory locations assigned to process PID
address: starting memory location of memory segment assigned to

A pid size: allocate ‘size’ consecutive locations to program number pid.
D pid address: deallocate memory allocated to program number pid at address.
T pid: Terminate program pid and deallocate all memory allocated to it.
H: halt the system.

The sequence of requests will always terminate with H.

Output Format
Each request has a corresponding output.
For every request of type A, print out the address of the starting location that is allocated. If the request is pending when the system halts, print -1.
For every D request, print out a 0 if it is a valid deallocation, else print out a 1.
For every T request, if the program had at least one request allocated, the answer for the terminate command must be 0, otherwise 1.
Each output should be printed on a separate line, in the order in which the requests are made. Note that the order in which allocation is done may be different.
Also, The total number of requests would be at most 10^6.

In this problem, correctness is important and not efficiency.
Simple data structures using lists/vectors are sufficient, although it is possible to have more efficient ones.

Sample Input

```
2 10
A 0 5
A 1 4
A 0 4
D 0 0
A 1 2
H
```

Sample Output

```
0
5
0
0
-1
```

Note that using a different strategy it is possible to allocate all requests. The locations assigned are [0,4], [6,9] [0,3] [4,5]. There is no efficient algorithm known to decide whether it is possible to satisfy all requests.