ACM-ICPC Live Archive

6863 Multi-Machine Scheduling of Two Applications

Little Ellie has two applications she needs to execute as quickly as possible. Application i ($i = 1, 2$) consists of $ns(i)$ identical steps, numbered from 1 to $ns(i)$. Ellie owns a cluster consisting of M machines (numbered from 1 to M) on which she can execute the steps of the 2 applications. The machines are not all identical, which means that the durations of executing one step on each machine may be different. To be more precise, it takes $T(i, j)$ seconds to execute one step of application i on machine j. Because the steps of each application are very CPUintensive, one machine can execute only one step of one application at one time (i.e. if multiple steps of the two applications are scheduled for execution on the same machine, the time intervals during which they are executed must be disjoint or only intersect at their endpoints). Moreover, the execution of any step of any application on any machine cannot be paused and then resumed later (on the same machine or on another machine). This means that once the execution of a step started on any machine, Ellie can either let the execution complete successfully or cancel it at any time before its completion and restart it later from the beginning (on the same machine or on any other machine).

Because of data dependencies, the steps of the same application must be executed sequentially (i.e. the execution of step j of application i can start only after the execution of step $j - 1$ of that application has finished — either exactly at the same time when the execution of step $j - 1$ finished or at any later time). The execution of step j can be scheduled either on the same machine on which step $j - 1$ was executed or on any other machine. There are no dependencies between two steps of different applications.

Assuming that Ellie can start executing the steps of the two applications at time moment 0, she would like to minimize the time moment $TEND$ (measured in seconds) when the last step of any application completes its execution. Please help her by telling her the minimum possible value of $TEND$.

Input

The first line of input contains the number of test cases T. The T test cases are described next. Each test case consists of 3 lines. The first line contains three integer numbers: $ns(1)$, $ns(2)$ and M. The second line contains M integers, representing, in order: $T(1, 1)$, $T(1, 2)$, ..., $T(1, M)$. The third line contains M integers, representing, in order: $T(2, 1)$, $T(2, 2)$, ..., $T(2, M)$.

Output

For each test case (in the order given in the input), output the minimum possible value of $TEND$ on a separate line.

Constraints:

- $1 \leq ns(i) \leq 1000000$ (10^6)
- $1 \leq M \leq 10$
- $1 \leq T(i, j) \leq 1000$

Explanation:

Example case 1: Since there is only one machine, all the steps of both applications are executed on the single available machine. The time needed for all of them to finish executing is $1000000 * 1 + 1000000 * 2 = 3000000$.

Example case 2: All the steps of application 1 are executed on machine 1 and all the steps of application 2 are executed on machine 5. The time needed for all the steps to finish is $\max 999999 * 1, 999998 * 1 = 999999$.

Example case 3: All the steps of application 1 are executed on machine 1 and all the steps of application 2 are executed on machine 2. The time needed for all the steps to finish is $\max 765432 * 1, 765 * 1000 = 765432$.

Example case 4: All the steps of application 1 are executed on machine 1 and the first 765 steps of application 2 are executed on machine 2. After the first 765 steps of application 2 finish executing at the time moment 765000, Ellie waits until time moment 765432, when machine 1 becomes available. She then schedules the last step of application 2 on machine 1 and its execution finishes at the time moment 765433.

Example case 5: All the steps of application 1 are executed on machine 2 and all the steps of application 2 are executed on machine 1. The time needed for all the steps to finish is $\max 3 * 2, 5 * 1 = 6$.

Sample Input

```
6
1000000 1000000 1
1
2
999999 999998 5
1 2 3 4 5
5 4 3 2 1
765432 765 2
1 2
1 1000
765432 766 2
1 2
1 1000
3 5 10
1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10
10 10 5
101 102 103 104 105
101 102 104 105 103
```

Sample Output

```
3000000
999999
765432
765433
6
1016
```