Qualification Round 2022 - Code Jam 2022

Punched Cards

PROBLEM   ANALYSIS

Problem

A secret team of programmers is plotting to disrupt the programming language landscape and bring punched cards back by introducing a new language called *Punched Card Python* that lets people code in Python using punched cards! Like good disrupters, they are going to launch a viral campaign to promote their new language before even having the design for a prototype. For the campaign, they want to draw punched cards of different sizes in ASCII art.

The ASCII art of a punched card they want to draw is similar to an $\mathbf{R} \times \mathbf{C}$ matrix without the top-left cell. That means, it has $(\mathbf{R} \cdot \mathbf{C}) - 1$ cells in total. Each cell is drawn in ASCII art as a period (`.`) surrounded by dashes (`-`) above and below, pipes (`|`) to the left and right, and plus signs (`+`) for each corner. Adjacent cells share the common characters in the border. Periods (`.`) are used to align the cells in the top row.

For example, the following is a punched card with $\mathbf{R} = 3$ rows and $\mathbf{C} = 4$ columns:

```
..+-+-+-+
..|.|.|.|
+-+-+-+-+
|.|.|.|.|
+-+-+-+-+
```

```
|.|.|.|.|
+-+-+-+-+
```

There are more examples with other sizes in the samples below. Given the integers $\mathbf{R}$ and $\mathbf{C}$ describing the size of a punched card, print the ASCII art drawing of it as described above.

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ lines follow, each describing a different test case with two integers $\mathbf{R}$ and $\mathbf{C}$: the number of rows and columns of the punched card that must be drawn.

Output

For each test case, output one line containing `Case #`x`:`, where x is the test case number (starting from 1). Then, output $(2 \cdot \mathbf{R}) + 1$ additional lines with the ASCII art drawing of a punched card with $\mathbf{R}$ rows and $\mathbf{C}$ columns.

Limits

Time limit: 5 seconds.
Memory limit: 1 GB.

Test Set 1 (Visible Verdict)

$1 \leq \mathbf{T} \leq 81.$
$2 \leq \mathbf{R} \leq 10.$
$2 \leq \mathbf{C} \leq 10.$

Sample

Sample Input

```
3
3 4
2 2
2 3
```

Sample Output

```
Case #1:
..+-+-+-+
..|.|.|.|
+-+-+-+-+
|.|.|.|.|
+-+-+-+-+
|.|.|.|.|
+-+-+-+-+
Case #2:
..+-+
..|.|
+-+-+
|.|.|
```

```
+-+-+
Case #3:
..+-+-+
..|.|.|
+-+-+-+
|.|.|.|
+-+-+-+
```

Sample Case #1 is the one described in the problem statement. Sample Cases #2 and #3 are additional examples. Notice that the output for each case contains exactly $\mathbf{R} \cdot \mathbf{C} + 3$ periods.

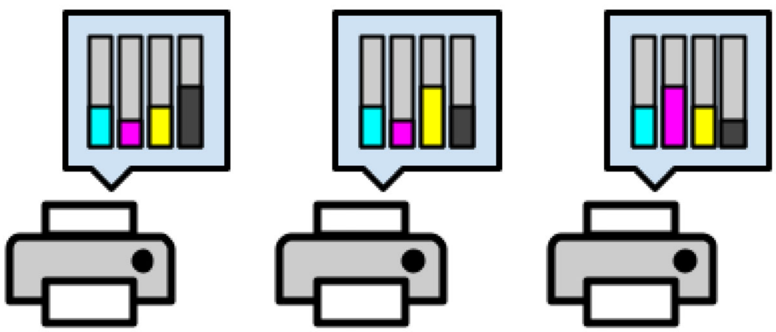
Qualification Round 2022 - Code Jam 2022

3D Printing

PROBLEM ANALYSIS

Problem

You are part of the executive committee of the Database Design Day festivities. You are in charge of promotions and want to print three D's to create a logo of the contest. You can choose any color you want to print them, but all three have to be printed in the same color.



You were given three printers and will use each one to print one of the D's. All printers use ink from 4 individual cartridges of different colors (cyan, magenta, yellow, and black) to form any color. For these printers, a color is uniquely defined by 4 non-negative integers c, m, y, and k, which indicate the number of ink units of cyan, magenta, yellow, and black ink (respectively) needed to make the color.

The total amount of ink needed to print a single D is exactly 10^6 units. For example, printing a D in pure yellow would use 10^6 units of yellow ink and 0 from all others. Printing a D in the Code Jam red uses 0 units of cyan ink, 500000 units of magenta ink, 450000 units of yellow ink, and 50000 units of black ink.

To print a color, a printer must have at least the required amount of ink for each of its 4 color cartridges. Given the number of units of ink each printer has in each cartridge, output any color, defined as 4 non-negative integers that add up to 10^6, such that all three printers have enough ink to print it.

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ test cases follow. Each test case consists of 3 lines. The i-th line of a test case contains 4 integers $\mathbf{C_i}$, $\mathbf{M_i}$, $\mathbf{Y_i}$, and $\mathbf{K_i}$, representing the number of ink units in the i-th printer's cartridge for the colors cyan, magenta, yellow, and black, respectively.

Output

For each test case, output one line containing `Case #`x`:` r, where x is the test case number (starting from 1) and r is `IMPOSSIBLE` if there is no color that can be printed by all 3 printers. Otherwise, r must be equal to "$c\ m\ y\ k$" where c, m, y, and k are non-negative integers that add up to 10^6 and $c \le \mathbf{C_i}$, $m \le \mathbf{M_i}$, $y \le \mathbf{Y_i}$, and $k \le \mathbf{K_i}$, for all i.

If there are multiple solutions, you may output any one of them. (See "What if a test case has multiple correct solutions?" in the Competing section of the FAQ.) This information about multiple solutions will not be explicitly stated in the remainder of the 2022 contest.

Limits

Time limit: 5 seconds.
Memory limit: 1 GB.

Test Set 1 (Visible Verdict)

$1 \le \mathbf{T} \le 100$.
$0 \le \mathbf{C_i} \le 10^6$, for all i.
$0 \le \mathbf{M_i} \le 10^6$, for all i.
$0 \le \mathbf{Y_i} \le 10^6$, for all i.
$0 \le \mathbf{K_i} \le 10^6$, for all i.

Sample

Sample Input

```
3
300000 200000 300000 500000
300000 200000 500000 300000
300000 500000 300000 200000
1000000 1000000 0 0
0 1000000 1000000 1000000
999999 999999 999999 999999
768763 148041 178147 984173
699508 515362 534729 714381
949704 625054 946212 951187
```

Sample Output

```
Case #1: 300000 200000 300000 200000
Case #2: IMPOSSIBLE
Case #3: 400001 100002 100003 399994
```

Sample Case #1 is the image provided above. The proposed color is using up all of the ink in the cyan, magenta, and yellow cartridges of the first printer and all of the ink in the black cartridge of the last printer. This means that no additional unit of ink could be used from any of the 4 ink colors, so the given sample output is the only possible output for this case.

In Sample Case #2, magenta is the only color that both the first and second printers have, so our only chance would be to use 10^6 units of magenta. Unfortunately, the third printer does not have quite enough, making this case impossible.

In Sample Case #3, other correct outputs are: "`400000 100000 100000 400000`", "`300000 0 0 700000`", and "`350000 140000 160000 350000`", among lots of others. Notice that "`300000 140000 160000 700000`" would not be a valid answer because, even though there is enough ink in all printers to do that, the total number of ink units must be exactly 10^6.

Qualification Round 2022 - Code Jam 2022

d1000000

PROBLEM ANALYSIS

Problem

While the most typical type of dice have 6 sides, each of which shows a different integer 1 through 6, there are many games that use other types. In particular, a dk is a die with k sides, each of which shows a different integer 1 through k. A $d6$ is a typical die, a $d4$ has four sides, and a $d1000000$ has one million sides.

In this problem, we start with a collection of $\mathbf{N}$ dice. The i-th die is a $d\mathbf{S_i}$, that is, it has $\mathbf{S_i}$ sides showing integers 1 through $\mathbf{S_i}$. A straight of length ℓ starting at x is the list of integers $x, x+1, \ldots, x+(\ell-1)$. We want to choose some of the dice (possibly all) and pick one number from each to form a straight. What is the longest straight we can form in this way?

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ test cases follow. Each test case is described in two lines. The first line of a test case contains a single integer $\mathbf{N}$, the number of dice in the game. The second line contains $\mathbf{N}$ integers $\mathbf{S_1}, \mathbf{S_2}, \ldots, \mathbf{S_N}$, each representing the number of sides of a different die.

Output

For each test case, output one line containing `Case #x: y`, where x is the test case number (starting from 1) and y is the maximum number of input dice that can be put in a straight.

Limits

Memory limit: 1 GB.
$1 \le \mathbf{T} \le 100$.

Test Set 1 (Visible Verdict)

Time limit: 5 seconds.
$1 \le \mathbf{N} \le 10$.
$4 \le \mathbf{S_i} \le 20$, for all i.

Test Set 2 (Visible Verdict)

Time limit: 15 seconds.
$1 \le \mathbf{N} \le 10^5$.
$4 \le \mathbf{S_i} \le 10^6$, for all i.

Sample

Sample Input

```
4
4
6 10 12 8
6
5 4 5 4 4 4
10
10 10 7 6 7 4 4 5 7 4
1
10
```

Sample Output

```
Case #1: 4
Case #2: 5
Case #3: 9
Case #4: 1
```

In Sample Case #1, there are multiple ways to form a straight using all 4 dice. One possible way is shown in the image above.

In Sample Case #2, since none of the dice can show an integer greater than 5, there is no way to have a straight with more than 5 dice. There are multiple ways to form a straight with exactly 5 dice. For example, pick the integers 4 and 5 for both $d5$'s and then integers $1, 2,$ and 3 for three of the $d4$'s to form $1, 2, 3, 4, 5$.

In Sample Case #3, it is possible to form the straight $1, 2, 3, 4, 5, 6, 7, 8, 9$ by discarding one $d4$ and using the $d4$'s, $d5$, and $d6$ to get 1 through 4; the $d7$'s to get 5 through 7; and the $d10$'s to get 8 and 9. There is no way to form a straight of length 10, so this is the best that can be done.

In Sample Case #4, we can only form a straight of length 1, but we can do so by picking any integer for the $d10$ we are given.

Qualification Round 2022 - Code Jam 2022

Chain Reactions

PROBLEM ANALYSIS

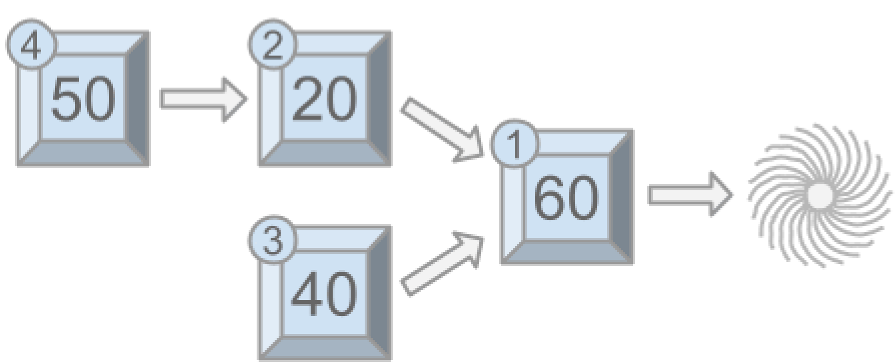
Problem

Wile lives alone in the desert, so he entertains himself by building complicated machines that run on chain reactions. Each machine consists of $\mathbf{N}$ modules indexed $1, 2, \ldots, \mathbf{N}$. Each module may point at one other module with a lower index. If not, it points at the abyss.

Modules that are not pointed at by any others are called *initiators*. Wile can manually trigger initiators. When a module is triggered, it triggers the module it is pointing at (if any) which in turn may trigger a third module (if it points at one), and so on, until the chain would hit the abyss or an already triggered module. This is called a *chain reaction*.

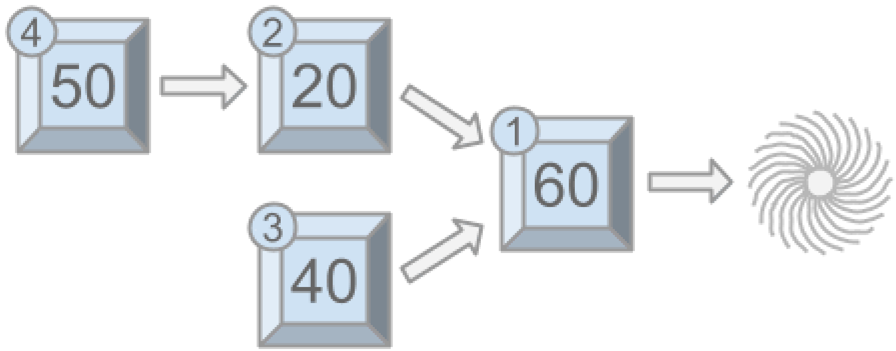
Each of the $\mathbf{N}$ modules has a fun factor $\mathbf{F_i}$. The fun Wile gets from a chain reaction is the largest fun factor of all modules that triggered in that chain reaction. Wile is going to trigger each initiator module once, in some order. The overall fun Wile gets from the session is the sum of the fun he gets from each chain reaction.

For example, suppose Wile has 4 modules with fun factors $\mathbf{F_1} = 60, \mathbf{F_2} = 20, \mathbf{F_3} = 40$, and $\mathbf{F_4} = 50$ and module 1 points at the abyss, modules 2 and 3 at module 1, and module 4 at module 2. There are two initiators (3 and 4) that Wile must trigger, in some order.



As seen above, if Wile manually triggers module 4 first, modules 4, 2, and 1 will get triggered in the same chain reaction, for a fun of $\max(50, 20, 60) = 60$. Then, when Wile triggers module 3, module

3 will get triggered alone (module 1 cannot get triggered again), for a fun of 40, and an overall fun for the session of $60 + 40 = 100$.



However, if Wile manually triggers module 3 first, modules 3 and 1 will get triggered in the same chain reaction, for a fun of $\max(40, 60) = 60$. Then, when Wile triggers module 4, modules 4 and 2 will get triggered in the same chain reaction, for a fun of $\max(50, 20) = 50$, and an overall fun for the session of $60 + 50 = 110$.

Given the fun factors and the setup of the modules, compute the maximum fun Wile can get if he triggers the initiators in the best possible order.

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ test cases follow, each described using 3 lines. Each test case starts with a line with a single integer $\mathbf{N}$, the number of modules Wile has. The second line contains $\mathbf{N}$ integers $\mathbf{F_1}, \mathbf{F_2}, \ldots, \mathbf{F_N}$ where $\mathbf{F_i}$ is the fun factor of the i-th module. The third line contains $\mathbf{N}$ integers $\mathbf{P_1}, \mathbf{P_2}, \ldots \mathbf{P_N}$. If $\mathbf{P_i} = 0$, that means module i points at the abyss. Otherwise, module i points at module $\mathbf{P_i}$.

Output

For each test case, output one line containing `Case #`x`:` y, where x is the test case number (starting from 1) and y is the maximum fun Wile can have by manually triggering the initiators in the best possible order.

Limits

Memory limit: 1 GB.
$1 \le \mathbf{T} \le 100$.
$1 \le \mathbf{F_i} \le 10^9$.
$0 \le \mathbf{P_i} \le i - 1$, for all i.

Test Set 1 (Visible Verdict)

Time limit: 5 seconds.
$1 \leq \mathbf{N} \leq 10$.

Test Set 2 (Visible Verdict)

Time limit: 5 seconds.
$1 \leq \mathbf{N} \leq 1000$.

Test Set 3 (Hidden Verdict)

Time limit: 10 seconds.
$1 \leq \mathbf{N} \leq 100000$.

Sample

Sample Input

```
3
4
60 20 40 50
0 1 1 2
5
3 2 1 4 5
0 1 1 1 0
8
100 100 100 90 80 100 90 100
0 1 2 1 2 3 1 3
```

Sample Output

```
Case #1: 110
Case #2: 14
Case #3: 490
```

Sample Case #1 is the one explained in the problem statement.

In Sample Case #2, there are 4 initiators (modules 2 through 5), so there are 4 chain reactions. Activating them in order $3, 5, 4, 2$ yields chains of fun $3, 5, 4, 2$ for an overall fun of 14. Notice that we are summing the four highest fun numbers in the input, so there is no way to get more than that.

In Sample Case #3, an optimal activation order of the 5 initiators is $4, 5, 7, 6, 8$.

Qualification Round 2022 - Code Jam 2022

Twisty Little Passages

PROBLEM ANALYSIS

Problem

You are investigating a cave. The cave has $\mathbf{N}$ rooms. There are underground passages that bidirectionally connect some pairs of rooms. Each room has at least one passage connected to it. No passage goes from a room to itself, and no two rooms are connected by more than one passage.

When in a room, you can identify what room you are in and see how many passages it connects to, but you cannot distinguish the passages. You want to estimate the number of passages that exist in the cave. You are allowed to do up to $\mathbf{K}$ operations. An operation is either:

- be magically teleported to a room of your choice, or
- walk through a random passage connected to the room you are in, taking you to the room at the other end of that passage.

When you decide to walk through a passage, you are unable to choose which one, because they are all alike. A passage is chosen for you uniformly at random.

You begin the investigation in an arbitrary room. Estimate the number of passages between rooms in the cave with at most $\mathbf{K}$ operations.

If E is your estimate and P is the actual number of passages, your solution is considered correct for a test case if and only if $P \cdot 2/3 \leq E \leq P \cdot 4/3$.

To pass a test set, your solution must be correct for at least 90% of the test cases in that set.

Input and output

This is an interactive problem. You should make sure you have read the information in the Interactive Problems section of our FAQ.

Initially, your program should read a single line containing an integer, $\mathbf{T}$, the number of test cases. Then, $\mathbf{T}$ test cases must be processed.

For each test case, your program must first read a line containing two integers $\mathbf{N}$ and $\mathbf{K}$: the number of rooms in the cave, and the maximum number of room operations you are allowed. Rooms are numbered between 1 and $\mathbf{N}$. The cave is determined at the beginning of the test case – it won't be changed while you explore it. Then, your program must process up to $\mathbf{K} + 1$ exchanges.

The i-th exchange starts with you reading a line containing two integers $\mathbf{R_i}$ and $\mathbf{P_i}$, representing the number of the room you are currently in and the number of passages it connects to. Then, you must output a single line containing one of the following:

- A single uppercase `W`: this means you want to walk through a random passage.
- A single uppercase `T` and an integer S: this means you want to teleport to room S.
- A single uppercase `E` and an integer E: this means you want to finish exploring and estimate that the cave contains E passages.

After an estimation operation, the judge will immediately start the next test case if there is one, regardless of the correctness of your estimation. If there is no next test case, the judge will wait for you to finish without any further output.

If the judge receives an invalidly formatted line from your program at any moment, or if your $(\mathbf{K} + 1)$-th exchange for a test case is not an estimation operation, the judge will print a single number -1 and will not print any further output. If your program continues to wait for the judge after receiving a -1, your program will time out, resulting in a Time Limit Exceeded error. Notice that it is your responsibility to have your program exit in time to receive a Wrong Answer judgment instead of a Time Limit Exceeded error. As usual, if the memory limit is exceeded, or your program gets a runtime error, you will receive the appropriate judgment.

Limits

Time limit: 120 seconds.
Memory limit: 1 GB.

Test Set 1 (Visible Verdict)

$1 \leq \mathbf{T} \leq 100$.
$2 \leq \mathbf{N} \leq 10^5$.
$K = 8000$.
Each room has at least one passage connected to it.

Testing Tool

You can use this testing tool to test locally or on our platform. To test locally, you will need to run the tool in parallel with your code; you can use our interactive runner for that. For more information, read the instructions in comments in that file, and also check out the Interactive Problems section of the FAQ.

FAQ.

Instructions for the testing tool are included in comments within the tool. We encourage you to add your own test cases. Please be advised that although the testing tool is intended to simulate the judging system, it is **NOT** the real judging system and might behave differently. If your code passes the testing tool but fails the real judge, please check the Coding section of the FAQ to make sure that you are using the same compiler as us.

Download testing tool

Sample Interaction

Judge	Solution
Number of cases	
`1`	
Case 1	
`5 3`	
Judge gives $\mathbf{N} = 5, \mathbf{K} = 3$.	
`4 1`	
We start at room 4 *which has* 1 *passage.*	
	`T 5`
	Teleport to room 5.
`5 2`	
It has two passages.	
	`W`

Walk through a passage.

```
4 1
```

We arrived at room 4 again.

```
T 1
```

Teleport to room 1.

```
1 3
```

It has three passages.

```
E 5
```

Guess 5 passages.
(It can be shown that the actual number of passages is either 4 or 5. The two possible graphs for this test case are shown below.)