2020 SBU ACM Programming Selection Contest (Sponsors: IACS & GSO)

(Stony Brook University, Nov 18, 2020)

Welcome to the contest! We would like to remind you of several things:

- This contest is a local ACM-style contest. Each participant must compete alone (this is not a team contest).
- The contest lasts for 3 hours (7:30PM - 10:30PM).
- C / C++ / Java / Python would be supported. GCC 7.3.0, G++ 7.3.0, Java 8, PyPy 2.7 and PyPy 3.5 are used.
- All submissions read from stdin and output to stdout.
- You can write and test your code directly on the HackerRank platform. If you have any question, don?t hesitate to ask. Again, enjoy the contest!

Problem setters:

A. SBUWW Potion Making (Haochen Chen/Allen Kim)

B. Optimal Riders (Shawn Mathew)

C. Monster Numbers (Zafar Ahmad)

D. The New Sisyphean Task (Allen Kim)

E. Crocodile Logistics (Taras Kolomatski)

F. Resilient Connections (Jiarui Zhang)

G. Solitaire Chess (Yimin Zhu)

SBUWW Potion Making

In the Stonybrook University of Witchcraft and WIzardry, we have a potion making machine that is operating throughout the day to help keep the local stockpile full. During this time, potions are produced at varying rates due to varying quality of ingredients being passed into the machine.

Unfortunately, the school's logging systems are not as advanced as their magic, and only the rate in which potions are created per hour as well as the number of hours elapsed are logged. The school would like to keep track of how many potions are created every day given these two metrics.

Help them design a new spell to calculate the number of potions created.

Input Format

On the first line, we are given an integer n equal to the number of logs.

On the following n lines, we have two integers, r and t, representing the rate at which the potions were made per hour and the time in hours the machine was running for respectively.

Constraints

- $1 \leq n \leq 10$
- $1 \leq s \leq 90$
- $1 \leq t \leq 12$. The values for t are always in strictly increasing order

Output Format

A single integer representing the total number of potions made

Sample Input 0

/

```
3
20 2
30 6
10 7
```

Sample Output 0

```
170
```

Explanation 0

We have 20 potions per hour from hours 0 to 2 (2 hours total), 30 potions per hour from hours 2 to 6 (4 hours total), and 10 potions per hour from hours 6 to 7 (1 hour total). In total, we have $20 \cdot 2 + 30 \cdot 4 + 10 \cdot 1 = 170$.

/

Optimal Riders

I own a carnival ride that takes 3 people at a time. Each time the ride runs I incur a certain amount of operating costs. Help me reduce my costs, by maximizing the number of people who ride at once.

Due to COVID, groups will request to come and I will schedule them to different sessions. Since I am doing the scheduling I can combine multiple groups into a single session so that I can optimize my operating costs. A session is considered optimal if the number of people in the session is divisible by 3. I would still like to be COVID conscious, so I would like to maximize the number of optimal sessions. If there are extra groups, I can tell them we are booked full and to come another day How many optimal sessions can I create given a list of groups?

For example, for a day if I had the groups [4, 1, 3, 5] I can combine groups of size 1 and 5 to get the following groups, [4, 6, 3], where only two of these groups are optimal.

Input Format

The first line for each day contains an integer n, the number of groups for the day.

The second line for each day contains n integers $g_1, g_2, \ldots, g_n$, the size of each group.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq g_i \leq 10^9$

Output Format

An integer s for each day in the input

Sample Input 0

```
4
4 1 3 5
```

Sample Output 0

```
2
```

/

Monster Numbers

You are organizing a puzzle tournament for your young brother and his friends. You went to buy some digit cards at a local stationery store and saw the price of each type of digit card is not equal. Initially, you were a bit annoyed by that! But eventually, you think of making a new puzzle for the tournament out of this scenario.

You have a certain amount of money in your pocket and the prices of each type of digit card. The task of the puzzle is to buy digit cards using the limited amount of available balance and construct the biggest possible number using those cards.

For example, you have 49 cents and the price of 0 is 10, 1 is 12, and 2 is 14. The largest possible number you can construct would be 2200 by spending the amount of money you have in your pocket. Note that, you do not need to spend all of your balance.

Write a program to validate the answer of the puzzle for the tournament!

Input Format

Each test case starts with a line containing two integers d and n specifying the types of digits available and the balance respectively. The following line has d integers. The i^{th} element (zero-indexed) of d integers represents the price of digit i (i.e: 0-9).

Constraints

$0 < d \leq 10$

$0 < n \leq 50$

$0 < d_i \leq 50$

Output Format

Output the largest number you can construct without having any leading zeros. It is guaranteed to be able to construct a single integer based on the given input.

Sample Input 0

```
3 49
10 12 14
```

Sample Output 0

/

```
2200
```

Sample Input 1

```
3 4
1 2 3
```

Sample Output 1

```
100
```

Sample Input 2

```
2 4
2 6
```

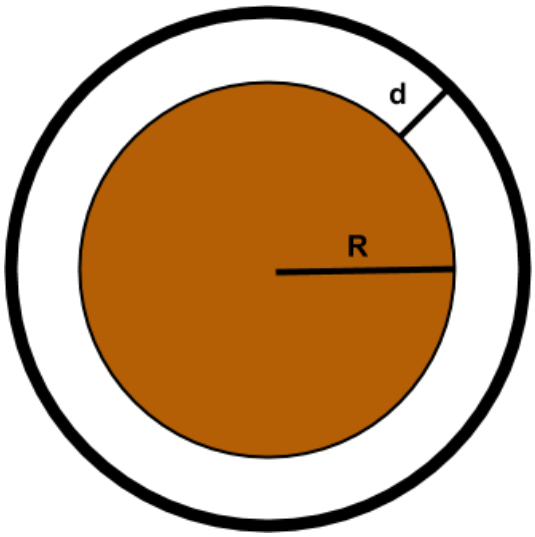
Sample Output 2

```
0
```

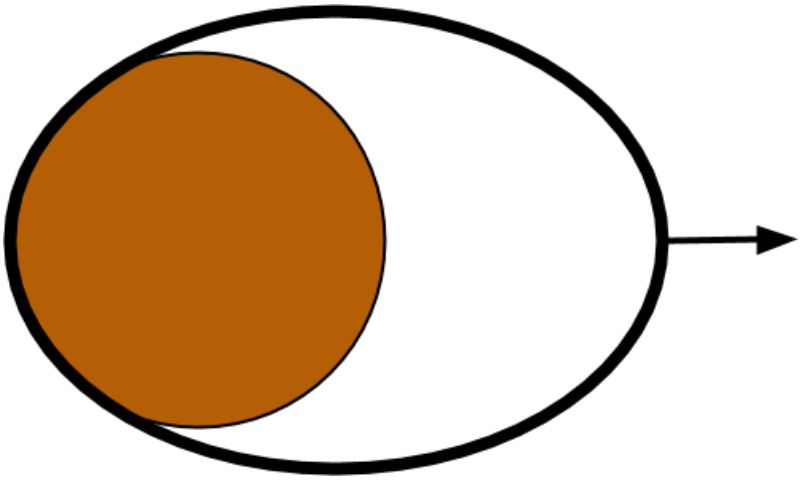
/

The New Sisyphean Task

In Greek mythology, king Sisyphus was doomed to push a boulder up a hill repeatedly for eternity as a punishment for his hubris. However, pushing was deemed to be too easy. The gods demanded that he should be pulling the boulder instead. Thus, for a boulder with radius **R**, they procured a rope encircling the boulder, with a uniform distance of **d** from its outer edge.



They began to design the hill in which he would need to pull, but they realized the situation was different from when the boulder was being pushed. When Sisyphus would initially start pulling on the rope, there would be some time and distance required to make the rope taut on the edge of the boulder. Depending on the length of the rope and size of the boulder, this distance needed to be computed to ensure the hill was at least this length.



Assume the boulder is modeled as a perfect sphere, Sisyphus modeled by a point, and the rope as an unstretchable string. What is the distance from Sisyphus to the closest surface of the boulder when the rope becomes fully taut when pulled directly away? Assume the rope is perfectly centered on the boulder.

Input Format

- Two floating point values **R** and **d** separated by a space

Constraints

- $0 < \mathbf{R} \leq 10^9$

/

- $0 \leq \mathbf{d} \leq 10^9$

Output Format

- A single float value **L** representing the distance from the furthest point when taut to the nearest surface of the boulder
- Precision check is of order 1e-5

Sample Input 0

```
1 0
```

Sample Output 0

```
0.0
```

Explanation 0

If **d** is 0, then there is no slack between the rope and the boulder. Thus, the length is 0.

Sample Input 1

```
23.5 3.14
```

Sample Output 1

```
16.06890035584027
```

Explanation 1

The input and output may be float values.

/

Crocodile Logistics

locked

Problem Submissions Discussions

[Photo Credit: Wikimedia Commons]

You live in the Australian outback. Your job involves fairly typical Australian activities like proactively ensuring that your neighbours and colleagues aren't eaten by saltwater crocodiles.

You are responsible for the logistics of relocating large, wild, and hungry crocodiles away from a human habitation in the town of (**A**)ntimony. Your goal is to get all of the crocodiles secured at your (**C**)orporate HQ, for later reintroduction into more remote areas. Between A and C, there is a (**B**)arge harbour. A and B are separated by a river, and B and C are separated by a road. You have at your disposal one boat on the river and one truck on the road. Your company has some number of workers, initially distributed across these sites.

Each vehicle can carry at most two entities (e.g. one person and one crocodile), and given that the crocodiles are unable or unwilling to operate motor vehicles, each mode of transport requires a human operator when in use. The time taken to complete each leg of the trip is approximately equal, so you've decided that it would be best to schedule all trips relative to a periodic time interval. (You

may assume that in a given cycle, all departures are simultaneous and all arrivals are simultaneous, with the two being separated by a short coordinated loading/unloading period.)

At site C, crocodiles are secured in holding cells, so no danger can come to humans there (even if they are sorely outnumbered). However, at any given point in time, at each of sites A and B, it is unsafe for crocodiles to strictly outnumber humans (provided that there are people present; leaving unattended crocodiles is permitted if there aren't also potential snacks around).

Given the initial locations of crocodiles, workers, and vehicles your task is to determine the minimum number of trip-cycles necessary to get ***all of the crocodiles and all of the workers*** to site C.

Input Format

The first line of the input consists of a single integer T, the number of test cases. The next T lines each consist of six integers (HA, HB, HC, CA, CB, CC) and two characters (Boat, Truck).

These represent the configuration of the problem at time zero.

HA, HB, and HC are the number of humans at A, B, and C, respectively. Similarly for CA, CB, CC and crocodiles.

Boat is either the character 'A' or 'B', and Truck is either the character 'B' or 'C'. These represent the starting position of the vehicles.

Constraints

$1 < T \leq 1000$

HA + HB + HC <= 20, CA + CB + CC <= 20. Inputs are always safe.

Output Format

The output consists of T lines.

For each case, if it is possible to get all of the people and crocodiles to site C, then output a line consisting of a single integer representing the earliest time at which this may be achieved.

Otherwise, output a line consisting of the string "IMPOSSIBLE" (without quotation marks).

Sample Input

```
1
1 0 1 0 2 0 A C
```

Sample Output

```
6
```

Submissions: 1
Max Score: 0

Rate This Challenge:

Download problem statement

Download all test cases

Suggest Edits

Collapse

Resilient Connections

[Photo Credit: a MAT 250 student]

CQ CQ This is Victor Alpha Three Mike India Delta transmitting on UHF. Are you doing your homework?

You recall that last summer, Long Island was struck by a storm that struck out power lines, and disconnecting a majority of the students taking the summer course which you were teaching and messing up your semester's schedule.

You would like to prevent this situation from arising in the future. Your general approach to problem solving incorporates a firm grasp of fundamentals with the use of timeless technologies. Navigating bureaucracy to fix Long Island's public infrastructure and ensuring that cell towers stay up is messy and complicated; if you try working with bloated conventional tools, you can't control their failure. Instead, you decide to set up your own communication network 'on the air' by setting up a HAM radio repeater network across the island, supported by backup generators. A repeater is a device that rebroadcasts any (appropriately prefixed) signal that it receives. (*Aside: Various controls on this system avoid feedback loops. You can assume that a given message will be transmitted from a given repeater at most once.*)

You are currently considering various potential layouts for your repeater network. It is your experience that sometimes atmospheric conditions or random radio finickiness impede transmission, so two given repeater stations might occasionally drop signals passed between them. You would like to study the resilience of your network up to single points of failure. That is, you're considering situations in which exactly one previously connected pair of repeaters fail to relay a given message.

In each configuration, you know which repeaters are within transmission distance of each other. Since the range on your student's battery-powered handheld radios is much shorter than the distance between any pair of repeaters, you can assume that all messages originate at a unique repeater. You would then like to determine the number of ordered

/

pairs or repeaters (i.e. pairs of entry and exit points) between which a signal may fail to be relayed (on the assumption that at most one previously connected pair of repeaters drops the message).

Input Format

The first line of the input consists of two integers **V** and **E**.

V is the number of repeaters in your configuration.

E is the number of pieces of repeater connection facts to be given.

The next **E** lines each consist of two integers **A** and **B** [1 ≤ **A** , **B** ≤ **V**], indicating that **A** and **B** are within transmission distance of each other. All of your repeaters are of equal power, so you may assume that if **A** is connected to **B**, then **B** is connected to **A** (and the input will not indicate this).

Constraints

0 ≤ **V** ≤ 20000

0 ≤ **E** ≤ 40000

Output Format

The output consists of a single line consisting of a single number **N**, which indicates the number of pairs (**C**,**D**) [0 ≤ **C**, **D** < **V** with **C** ≠ **D**] such that it is possible for a message to be relayed from **C** to **D** in the event of a single communication failure.

Sample Input 0

```
2 1
1 2
```

Sample Output 0

```
1
```

Sample Input 1

```
2 2
1 2
2 1
```

Sample Output 1

```
0
```

/

Solitaire Chess

locked

Problem Submissions Discussions

Cindy plays a Solitaire Chess on a n×n board. Solitaire Chess is a single player game that every move you make must result in a capture. Any piece can be used to capture the other piece. A player wins the game when there is only one piece remaining on the board.

Knight captures after moving in an "L" shape.

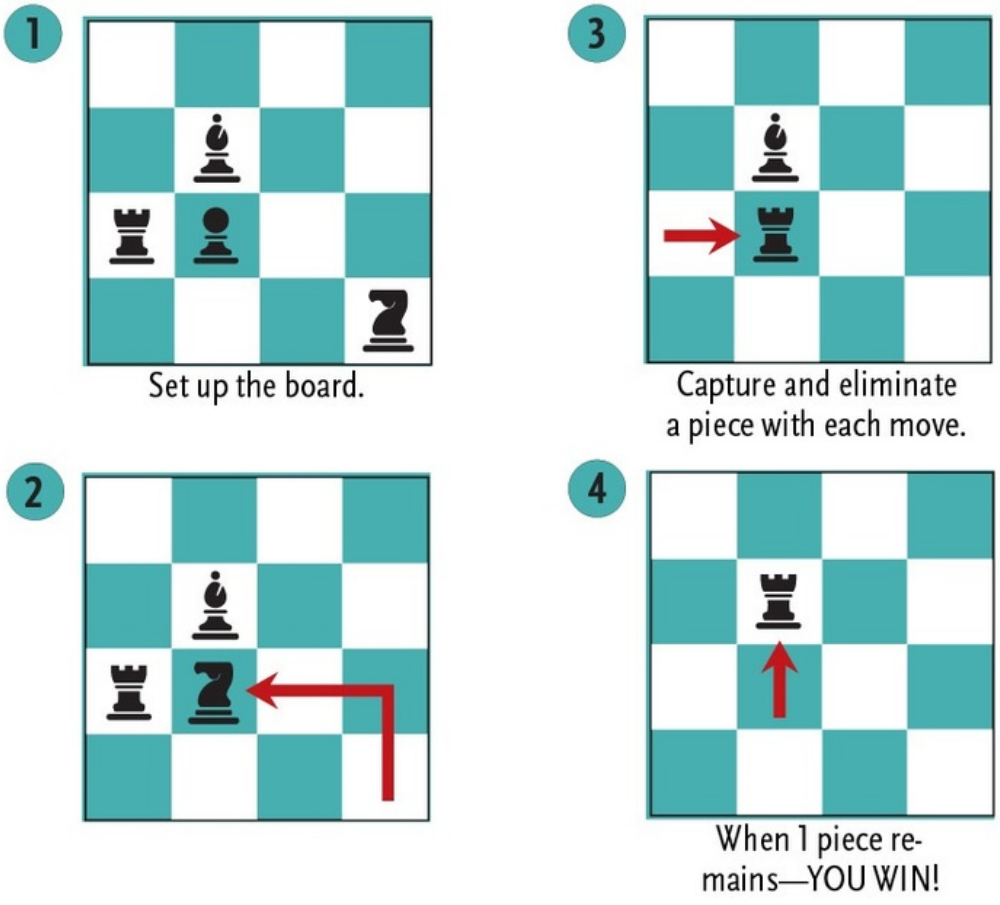
Bishop captures after moving across any number of cavant squares diagonally.

Rook captures after moving across any number if vacant squares vertically or horizontally.

Pawn capture after moving diagonally up one space. (upward only)

King captures after moving exactly one square horizontally, vertically, or diagonally.

Queen captures after moving across any number of vacant squares horizontally, vertically, or diagonally.



Input Format

It will contains n rows and for each row there are n integers.

For each integer in the matrix, 0 represents empty, 1 represents *Knight*, 2 represents *Bishop*, 3 represents *Rook*, 4 represents *Pawn*, 5 represents *King*, 6 represents *Queen*.

Constraints

n = 4

Output Format

True/False. (If Cindy can win, return True. Else, return False.)

Sample Input 0

```
1 0 0 0
0 2 0 0
4 0 0 0
0 0 0 0
```

Sample Output 0

```
True
```

Sample Input 1

```
0 0 0 0
0 0 0 0
0 0 4 0
0 0 1 0
```

Sample Output 1

```
False
```


C++

```
1 #include <cmath>
2 #include <cstdio>
3 #include <vector>
4 #include <iostream>
```

```
#include <algorithm>
using namespace std;


int main() {
    /* Enter your code here. Read input from STDIN. Print output to STDOUT */
    return 0;
}

```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code