Shenendehowa Code Sprint

May 2015

Rules and Scoring

1. Each team can have up to three members and can only use one computer.
2. Use the internet for programming language reference only; do not try to search for solutions to the problems.
3. There are seven problems. Each problem comes in two versions: easy and hard. The hard version has larger and more complex test data. For each version of a problem, if your submission gets every test case correct for that version, your team gets one point. With **7 problems** and two versions of each, the max score is **14 points**.
4. Getting a test case correct means your program outputted the correct answer within the time and memory constraints. The default constraints are 1 second per test case and 512MB memory usage (generally, memory will not be a problem).
5. In the event of a tie, we look at time penalties. They are calculated as follows: every team starts with a penalty of zero. After a team gets a correct submission, the team's time penalty is incremented by the number of minutes since the competition began. For example, a team that solves 1-easy at 10 minutes and 1-hard a minute afterwards has a time penalty of 21 and a score of 2.
6. You can re-submit solutions to a problem as often as you want, but **an incorrect submission will increase your time penalty by 10**.

Notes

- Don't re-submit code to a version of a problem if you've already gotten it correct.
- For the hard versions of many problems, the test data is quite large and your code may exceed the time limit. It might be better for you to solve the easier versions of other problems before coming up with a more efficient approach to the current problem.
- If you want to test the performance of your program on large test data, consider creating another program to generate test data for you.
- The range of `int` is roughly from -2×10^9 to 2×10^9; beware of overflow.
- The problems should be ordered by how difficult their "hard" versions are. But estimating difficulty precisely is hard. **The scoreboard is the best indicator of what problems are easiest**. The more teams have solved something, the easier it is.

Acknowledgments

Thanks to Team 20 for encouraging me to create this contest, and for helping lay out the logistics of the contest. I'm also very grateful to Shenendehowa for providing the venue for this competition, and to Alex Wei and Felix Sun for test-solving my problems and estimating their difficulty. Finally, thanks to HackerRank for providing such a convenient service for creating programming competitions!

-Philip Sun

Some Definitions

- Alphanumeric: consisting only of letters (upper or lower case) and numbers
- Lattice point: a point (x, y) such that x and y are both integers
- Vertex: a "corner" on a polygon
- Factor: x is a factor of y if $y \div x$ has no remainder and x, y are both positive integers
- Prime: a positive integer with exactly two factors
- Conjecture: a statement that hasn't been proven true

1 Patterns

Sammy sees a string written on a very large chalkboard. Sammy appends (adds to the end) to the string a copy of itself. He does this again, and again, and the board becomes filled with infinite copies of this original string. What is the mth character of this resulting string?

Input Format

Line 1: S, the original string on the chalkboard. S will be alphanumeric.
Line 2: m, where $m = 1$ corresponds to the first character of the original string

- Easy: S has between 1 and 100 characters; $1 \leq m \leq 1000$
- Hard: S has between 1 and 10^4 characters; $1 \leq m \leq 10^{15}$

Sample Input

TrXY120
22

Output Format

Line 1: The character asked for

Sample Output

T

Output Details

Writing a few copies of the string, we get: TrXY120TrXY120TrXY120TrXY120. The twenty-second character of this string is T.

2 Cow Fencing

There are n cows in a field, which can be imagined as points on a two-dimensional grid. They occupy distinct lattice points. Farmer John wants to construct a rectangular fence that encloses all the cows. A cow is considered enclosed if it is strictly inside the rectangle (being on the edge of the rectangle doesn't count). In addition, the sides of the fence must be parallel to the x and y axes, and the corners of the fence must be lattice points. What is the minimum area that such a fence must enclose?

Input Format

Line 1: n, a positive integer, indicating the number of cows.
Lines 2 to $n+1$: line $i+1$ gives the x and y coordinates (in that order) of cow i. The coordinates are separated by a space.

- Easy: $n \leq 20$ and the absolute value of each x and y coordinate will be below 100.
- Hard: $n \leq 10^4$ and the absolute value of each x and y coordinate will be below 10^8.

Sample Input

```
3
-1 -1
2 2
3 1
```

Output Format

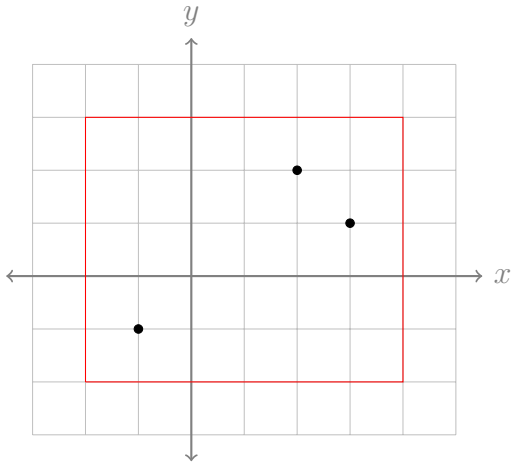
Line 1: a, the area of the smallest possible fence that satisfies Farmer John's conditions.

Sample Output

```
30
```

Output Details

The optimal fence design is shown in red to the right. The fence's corners are at $(-2,-2)$, $(4,-2)$, $(4,3)$, and $(-2,3)$. The fence is therefore 6 units wide and 5 units tall, giving an area of 30.



3 Making Burritos

You are a worker at Sammy's Burrito Shop. It takes you m seconds to make a burrito—this is constant and does not vary between burrito orders. Given the times that each of n orders comes in, calculate the earliest possible time for you to finish making all the burritos. You are allowed to serve the orders in any arrangement you choose, and as soon as you finish making one burrito, you can instantly transition to making another. But, you can only work on one burrito at a time, and cannot start making a burrito that no one has ordered yet.

In this problem, time is simply described as an integer t, the number of seconds since the start of your work day. The ith order comes in at time t_i, where t_i is a positive integer.

Time Limit: 2 seconds

Input Format

Line 1: m and n, separated by a space
Lines 2 to $n+1$: line $i+1$ describes the time in which the ith burrito order comes in

- Easy: $1 \leq n \leq 100$, $t_i \leq 1000$, and $1 \leq m \leq 1000$
- Hard: $100 < n \leq 10^5$, $t_i \leq 10^9$, and $1 \leq m \leq 10^4$

Sample Input

```
5 3
3
0
7
```

Output Format

Line 1: the minimum time t in which you can finish processing all burrito orders. The answer is guaranteed to be below two billion.

Sample Output

```
15
```

Output Details

From $t = 0$ to $t = 5$, we work on the second order. We then start the first order and finish at $t = 10$. We then make the last burrito, completing it at $t = 15$.

4 Numbering the FIRST Game Manual

The game manual for the 2016 FIRST season has been written. Now, its pages must be numbered. The first page is numbered 1, the second 2, and so on; the last page of an n-page manual is numbered n. Obviously, the numbers are written in base 10. Help the people of FIRST by determining how many digits must be written to number the entire manual.

Input Format

Line 1: a positive integer n, the number of pages in the manual

- Easy: $1 \leq n \leq 10^5$
- Hard: $1 \leq n \leq 10^{15}$

Sample Input

11

Output Format

Line 1: the number of digits required to number the manual

Sample Output

13

Output Details

Writing 1, 2, 3, 4, 5, 6, 7, 8, 9 requires nine digits. The final two pages, 10 and 11, require two digits each. We do $9 + 2 \times 2$ to get our answer of 13.

5 Goldbach's Conjecture

A famous, unsolved, but simple conjecture is that every even number greater than 2 can be expressed as the sum of two primes. This is known as Goldbach's Conjecture. Help mathematicians by testing this conjecture on some numbers.

Input Format

Line 1: x, an even integer greater than 2

- Easy: $x \leq 100$
- Hard: $x \leq 10^5$

Sample Input

10

Output Format

Line 1: two primes p_1 and p_2 such that $p_1 \leq p_2$ and $p_1 + p_2 = x$. Separate the primes with a space. If there are multiple p_1, p_2 that satisfy the criteria, choose p_1, p_2 such that $p_2 - p_1$ is minimized (the two primes that are as close together as possible).

Sample Output

5 5

Output Details

Five is a prime number, and $5 + 5 = 10$. Note that $p_1 = 3$, $p_2 = 7$ also add to 10, but $7 - 3 > 5 - 5$, so $p_1 = p_2 = 5$ is the only correct answer.

6 Mega-Tetris

Sammy is playing Mega-Tetris. This is like normal Tetris, but the pieces can be much more complex (in Tetris, each piece is composed of only four squares). The rules of Mega-Tetris are the same; there is a large grid, initially empty. Pieces appear from the top of the grid, and they keep "falling" down the grid until this is no longer possible. This occurs when the piece hits the bottom of the grid, or hits some piece below it. Sammy begins a new game and lets two pieces fall, without rotating or moving them. Output the resulting height of the blocks.

Height is defined as follows: number the bottommost row 1, the row above that 2, and so on. The height is the number of the highest row that is at least partially occupied.

Input Format

Line 1: n, the size of the pieces
Lines 2 to $n + 1$: these lines form an $n \times n$ grid of ones and zeros. This grid is a picture of what the **top** piece looks like. A 0 represents empty space, while a 1 represents a square occupied by the piece. Line 2 shows the topmost row of the piece, and each line after shows the row below.
Lines $n + 2$ to $2n + 1$: these lines describe the **piece on bottom**. Each line has n digits, and their meaning has been explained previously.

- Easy: $n \leq 10$
- Hard: $n \leq 500$

Sample Input

```
4
0100
0101
0111
1101
0000
0001
1101
1111
```

Output Format

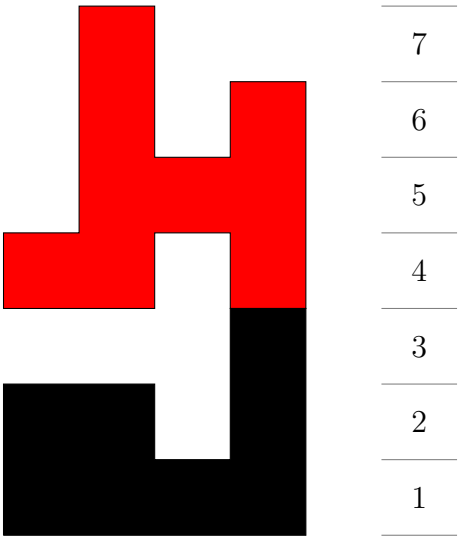
Line 1: a single integer h, the height of the grid after the two pieces have been dropped.

Sample Output

```
7
```

Output Details

The final arrangement of the two pieces is shown below, and the rows are numbered to the right:



What's a "Piece"?

Each piece will be non-empty; it will contain at least one 1. And all 1's within a piece will be connected to each other by shared edges. The following three examples are all **invalid** pieces for $n = 3$:

000
000
000

010
100
000

000
000
101

7 FRC Spending

A certain FRC team spends a lot of money through its p projects. Each project has a start time t_1, an end time t_2, and a "spending speed" s. All three will always be integers. Time in this problem simply means the number of milliseconds since the season began; it's a single positive number. A spending speed of s means that for every millisecond that passes in the interval $[t_1, t_2]$ the FRC team spends an additional s dollars on that project. The spending occurs between integer values of t. For example, $t_1 = 1, t_2 = 2, s = 2$ means that between $t = 1$ and $t = 2$, the robotics team spent \$2. At $t = 1$, the team had spent no money on the project. By $t = 2$ and for any $t > 2$, the team had spent \$2 total on that project.

Sammy has a list of all the FRC team's projects. He wants to find the smallest positive integer t such that by time t, the team had spent at least k dollars. Please help him.

Time Limit: 2 seconds

Input Format

Line 1: p and k, separated by a space. Both are positive integers.
Lines 2 to $p + 1$: line $i + 1$ describes project i. It gives the t_1, t_2, and s (in that order) of project i. These three integers are space-separated.

- Easy: $p \leq 50$; $k \leq 10^7$; $1 \leq t_1, t_2 \leq 10^4$; $1 \leq s \leq 100$
- Hard: $p \leq 10^4$; $k \leq 10^{18}$; $1 \leq t_1, t_2 \leq 10^{12}$; $-800 \leq s \leq 800$

Sample Input

2 100
1 6 15
3 9 10

Output Format

Line 1: the answer to Sammy's question. Output -1 if the FRC team never reaches at least k dollars of spending.

Sample Output

6

Output Details

Below is a table of how much the team spent total and on each project as time went on.

Time	1	2	3	4	5	6	7	8	9	10
Project 1	0	15	30	45	60	75	75	75	75	75
Project 2	0	0	0	10	20	30	40	50	60	60
Total	0	15	30	55	80	105	115	125	135	135