FPC 2018 problem presentation; spoiler alert!

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

FPC 2018

A

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

B - BINGO!

FPC 2018

Problem description

Given an $n \times n$ BINGO grid and m events, calculate when you can shout "BINGO!"

Observation

All events happen only once, so we incrementally count how many items are crossed in every row/column/diagonal

Edge Case

When the grid size is 1×1, you immediately win (due to the free square in the center, which is the only square)

Cryptography (1/3)

Problem description

- Given a number n $1 \leq n \leq 10^{10}$, decide whether it's a prime number or not.

Things to notice

- Since n can be 10 billion you have to use longs, not integers as they can only store up to 2.1 billion.
- The problem becomes a lot more easy if you know the modulo (%) operator.

FPC 2018

FPC 2018

Cryptography (2/3)

Naive approach

- if $n < 2$ output BROKEN
- else if $n == 2$ output SAFE
- else loop from $i = 2$ to n and check if a number $i \% n == 0$. If true output BROKEN else output SAFE.
- This takes approximately 10^{10} steps which would result in TIME LIMIT EXCEEDED.

First optimization

- Notice that after $n/2$ no divisor can be found anymore, so loop from 2 to $n/2$. This reduces the number of steps to approximately 5 billion, which is unfortunately still too much.

Cryptography (3/3)

A
B - BINGO!
Cryptography
D
E - Efficient Printing
F - Floor Price Calculator
G - Guessing Game
H - Hungry Wolves
I

FPC 2018

Correct approach

- The correct approach is to loop until the square root of n.
- You are looking for pairs of numbers a and b so that $a * b = n$ if n happens to be a composite number. You would only need the smallest of the two and this number must be smaller or equal to $\sqrt{n}$, if this would not be the case both a and b would be strictly greater than $\sqrt{n}$ contradicting the fact that $a * b = n$.
- Using this approach you end up with approximately $\sqrt{10^{10}} = 10^5 = 100.000$ which is perfectly fine.
- An optional optimization is to check if $n\%2 == 0$ and if not loop from i=3 to $\sqrt{n}$ where you skip all even numbers by incrementing i with 2 every time. This would leave you with approximately 50.000 steps.

D

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

FPC 2018

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

FPC 2018

E - Efficient Printing (1/3)

Problem description

Given an integer n, count the trailing zeroes of $n!$ ($n \leq 10^{18}$)

Brute Force

- Calculate $n!$ using e.g. Java `BigInteger` (very slow)

Primes

- The prime factorization of 10 is $2 \cdot 5$
- Count all prime factors 2 and 5 in the result of $n!$
- Only count the prime factor 5, as 2 occurs more often
- To do: 10^{18} times prime factorization (still slow)

A
B - BINGO!
Cryptography
D
E - Efficient Printing
F - Floor Price Calculator
G - Guessing Game
H - Hungry Wolves
I

FPC 2018

E - Efficient Printing (2/3)

Smart Counting

- For all i between 1 and n, count how many times you can divide i by 5
- Still have to do this 10^{18} times, so still too slow

Optimized Counting

- Using integer division: $n/5$ is the amount of numbers that are divisible by 5 in the range $[1, n]$
- Similarly for $n/25$, $n/125$, ...
- Sum of all divisions = number of 5s in prime factors of $n!$
- Complexity: $\mathcal{O}(\log_5 n)$, takes approx. 26 iterations for 10^{18}

E - Efficient Printing (3/3)

A
B - BINGO!
Cryptography
D
E - Efficient Printing
F - Floor Price Calculator
G - Guessing Game
H - Hungry Wolves
I

FPC 2018

Example

- $42! =$ 1405006117752879898543142606244511569936384000000000
- $42/5 = 8$
- $42/25 = 1$
- $42!$ has $8 + 1 = 9$ trailing zeroes

Example

- $256!$ has 63 trailing zeroes
- $256/5 = 51$
- $256/25 = 10$
- $256/125 = 2$

F - Floor Price Calculator

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

FPC 2018

Problem description

Determine the total number of squares in a n x n floor.

Solution:

- The total number of squares $f(n)$ in a n x n floor is given by the following summation:
- $f(n) = \sum_{i=1}^{n} i^2$
- Can be implemented by using one for-loop. Create a variable to keep track of the current answer.
- You should be using the *long* data type in Java in order to avoid overflow errors for a big value of n.

A
B - BINGO!
Cryptography
D
E - Efficient Printing
F - Floor Price Calculator
G - Guessing Game
H - Hungry Wolves
I

FPC 2018

G - Guessing Game

Problem

Guess the correct number.

- You can't guess every number, that would be too slow.
- Use binary search!

Solution

- Keep track of a lower l and upper u limit and repeat:
  - guess $x = (l + u)/2$
  - If x is too low, set $l = x + 1$
  - If x is too high, set $u = x - 1$

H - Hungry Wolves

A

B - BINGO!

Cryptography

D

E - Efficient Printing

F - Floor Price Calculator

G - Guessing Game

H - Hungry Wolves

I

Problem description

- Given the area A
- Compute the radius $r = \sqrt{\frac{A}{\pi}}$
- Compute the perimeter $P = 2\pi r$
- Round up the solution $\frac{\lceil P \cdot 10 \rceil}{10}$

I

A
B - BINGO!
Cryptography
D
E - Efficient Printing
F - Floor Price Calculator
G - Guessing Game
H - Hungry Wolves
I