CTU Open 2023

Presentation of solutions

October 21, 2023

Natatorium

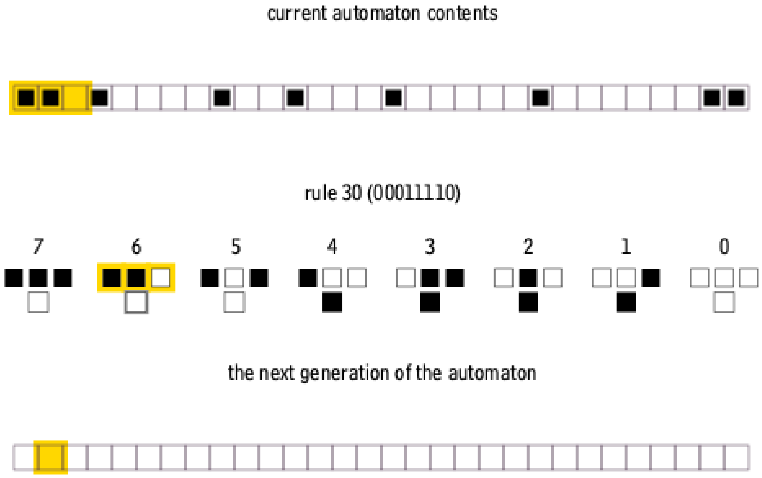
Natatorium

- Find the two primes P_i that divide C
- If C is a product of two primes P and Q, then P and Q are the only primes that divide C

Wall

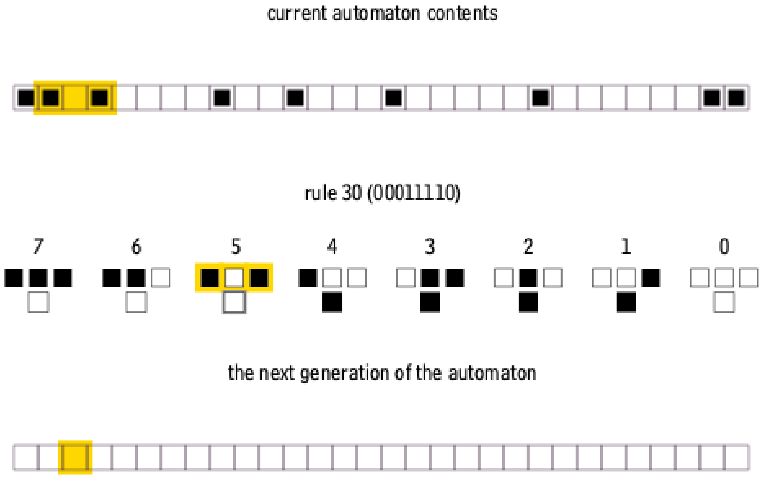
Wall

- **Task:** Simulate run of an elementary celular automata.



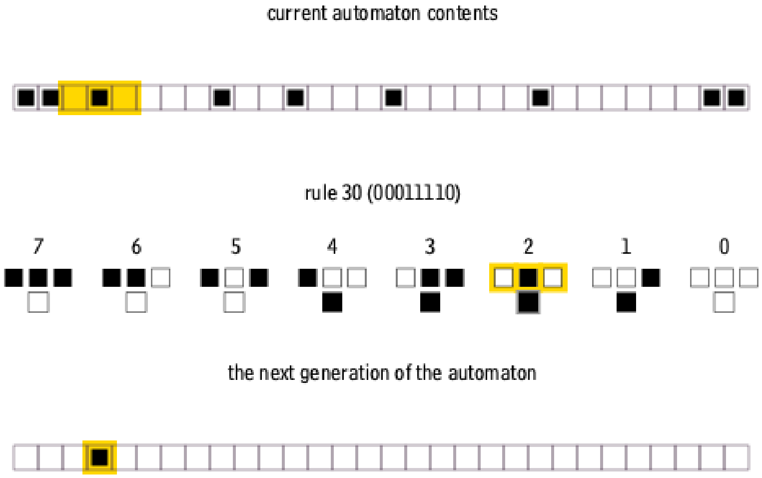
Wall

- **Task:** Simulate run of an elementary celular automata.



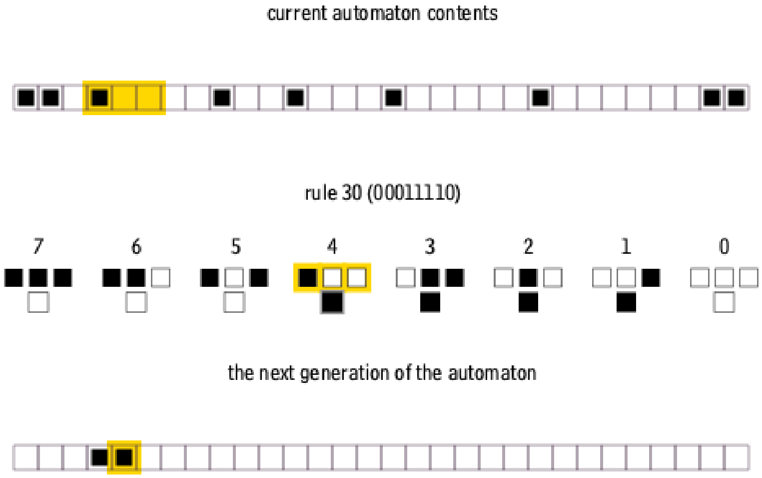
Wall

- **Task:** Simulate run of an elementary celular automata.



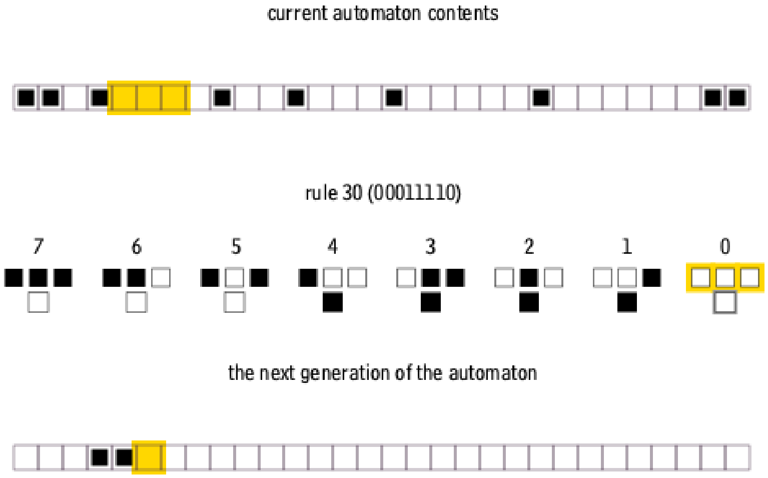
Wall

- **Task:** Simulate run of an elementary celular automata.



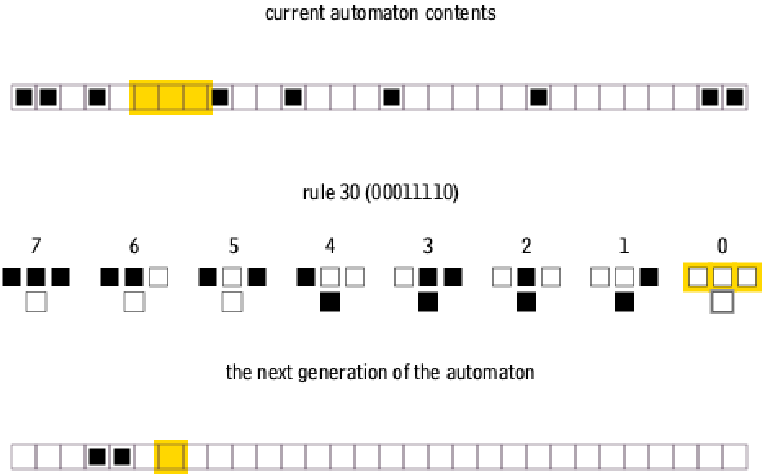
Wall

- **Task:** Simulate run of an elementary celular automata.



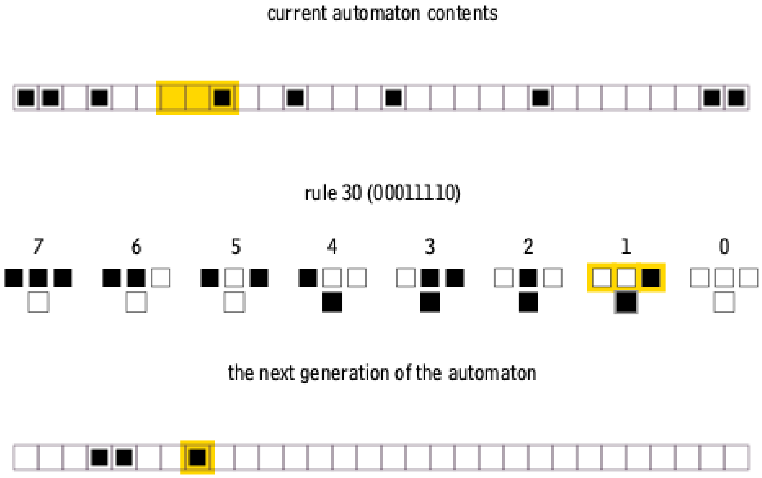
Wall

- **Task:** Simulate run of an elementary celular automata.



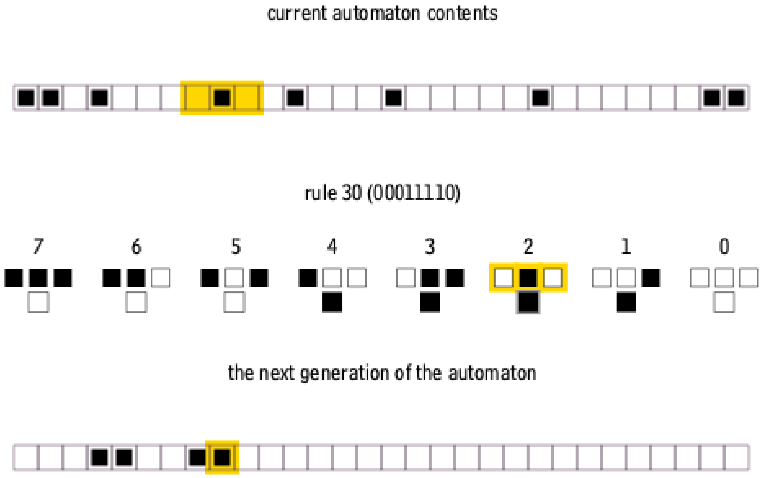
Wall

- **Task:** Simulate run of an elementary celular automata.



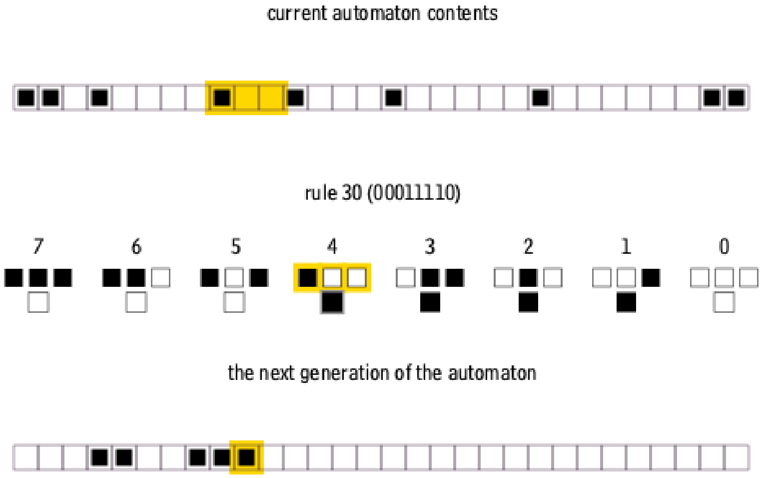
Wall

- **Task:** Simulate run of an elementary celular automata.



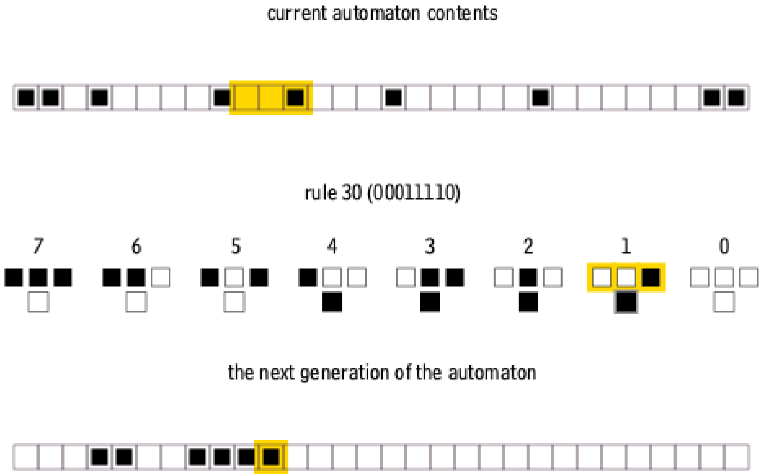
Wall

- **Task:** Simulate run of an elementary celular automata.



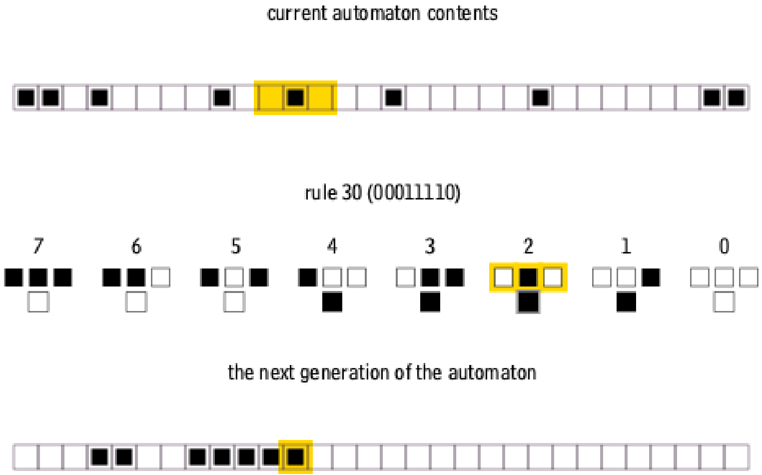
Wall

- **Task:** Simulate run of an elementary celular automata.



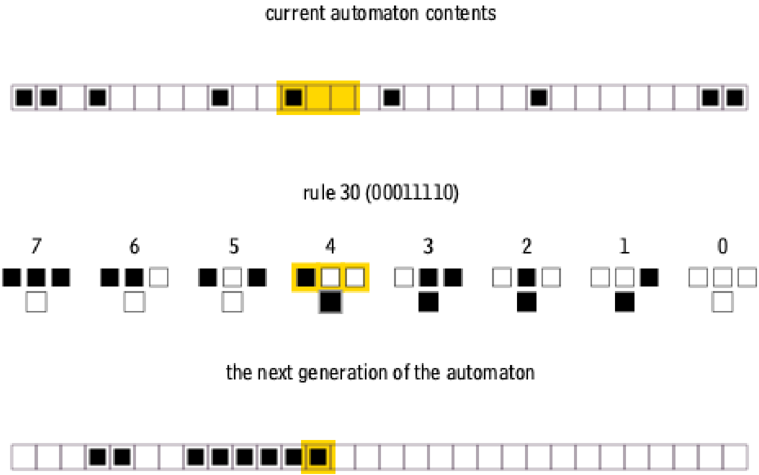
Wall

- **Task:** Simulate run of an elementary celular automata.



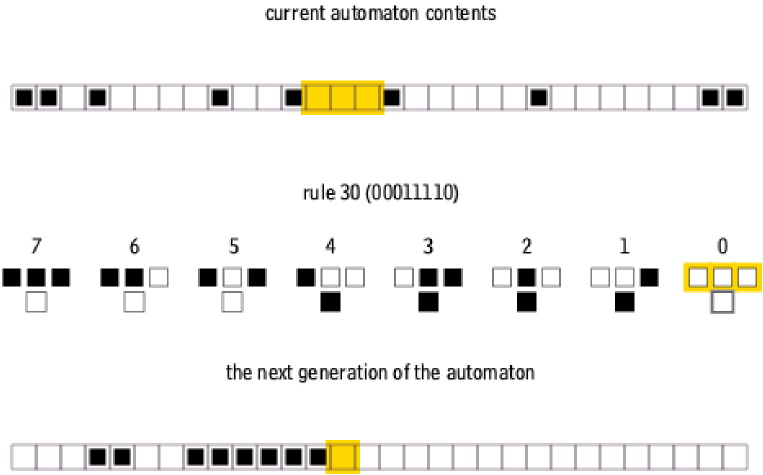
Wall

- **Task:** Simulate run of an elementary celular automata.



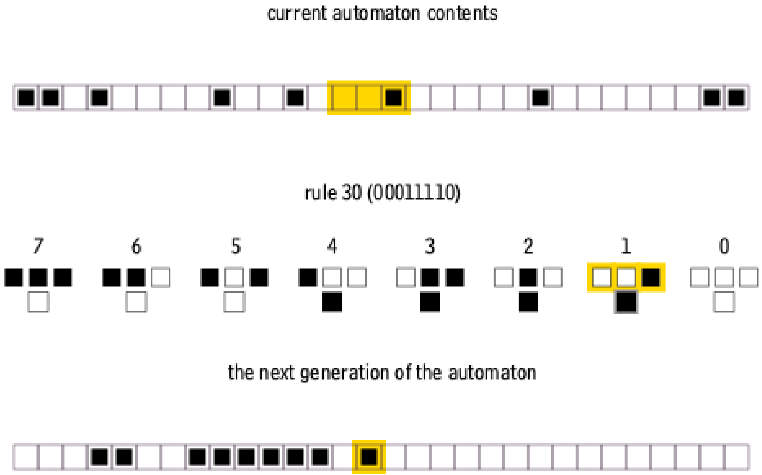
Wall

- **Task:** Simulate run of an elementary celular automata.



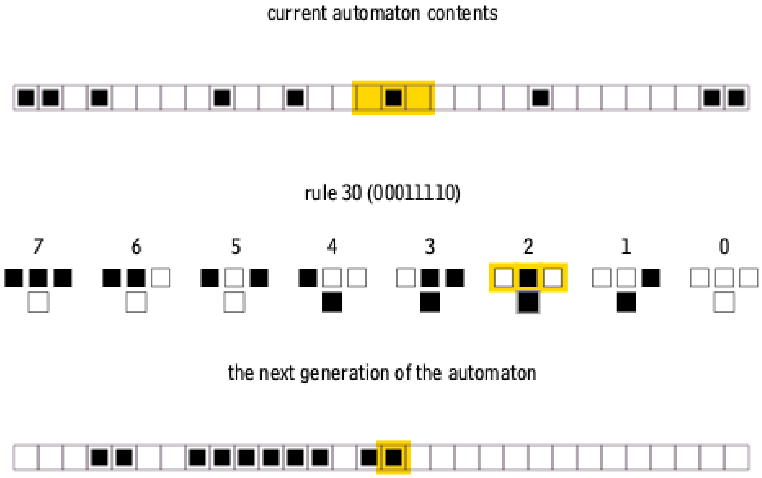
Wall

- **Task:** Simulate run of an elementary celular automata.



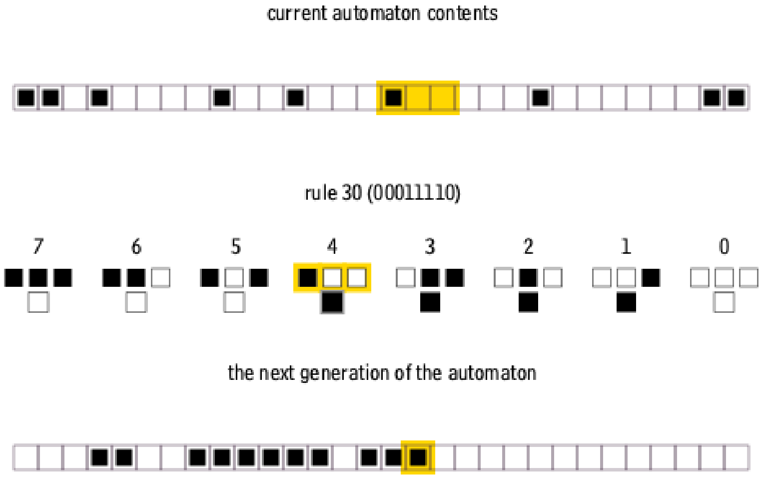
Wall

- **Task:** Simulate run of an elementary celular automata.



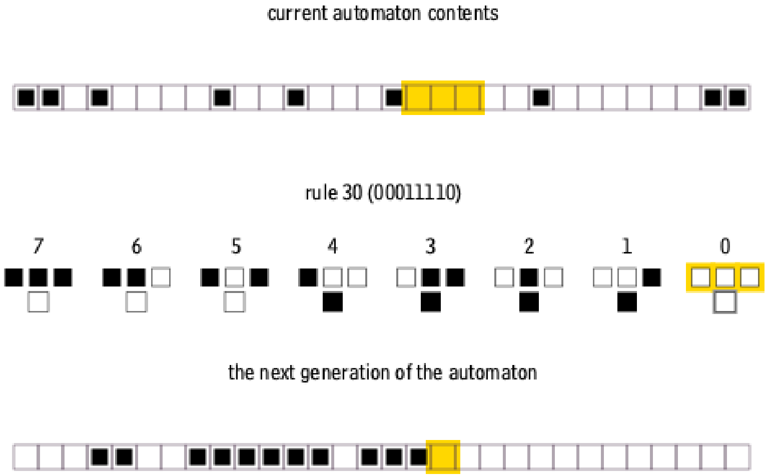
Wall

- **Task:** Simulate run of an elementary celular automata.



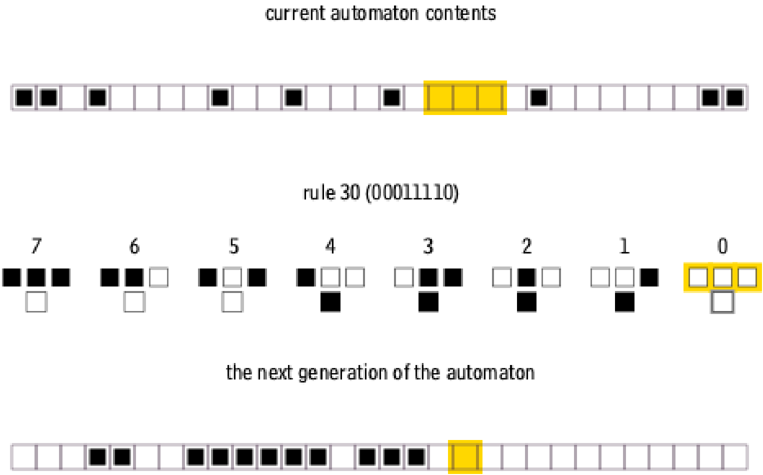
Wall

- **Task:** Simulate run of an elementary celular automata.



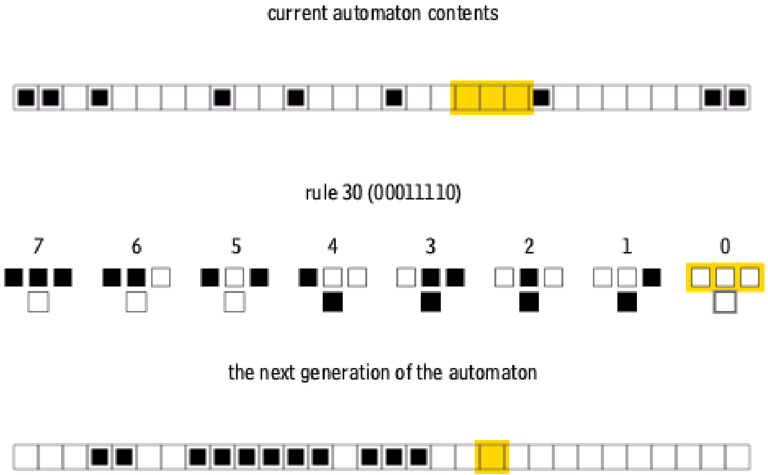
Wall

- **Task:** Simulate run of an elementary celular automata.



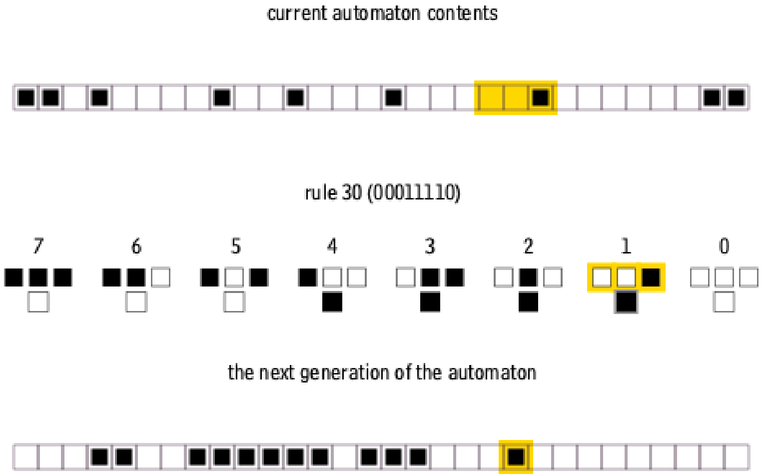
Wall

- **Task:** Simulate run of an elementary celular automata.



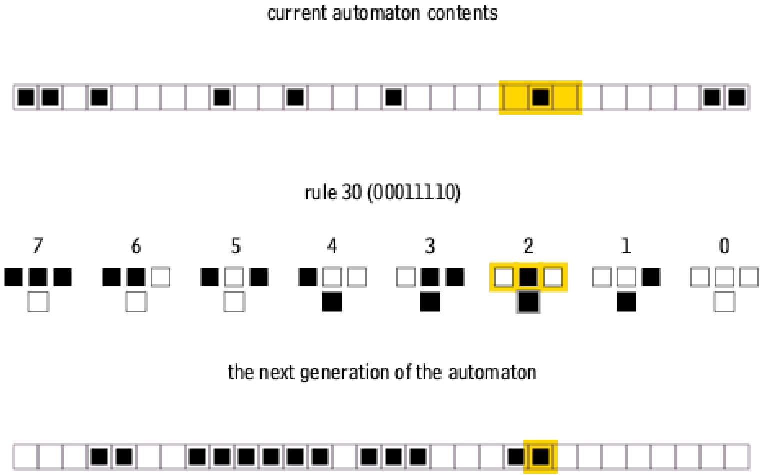
Wall

- **Task:** Simulate run of an elementary celular automata.



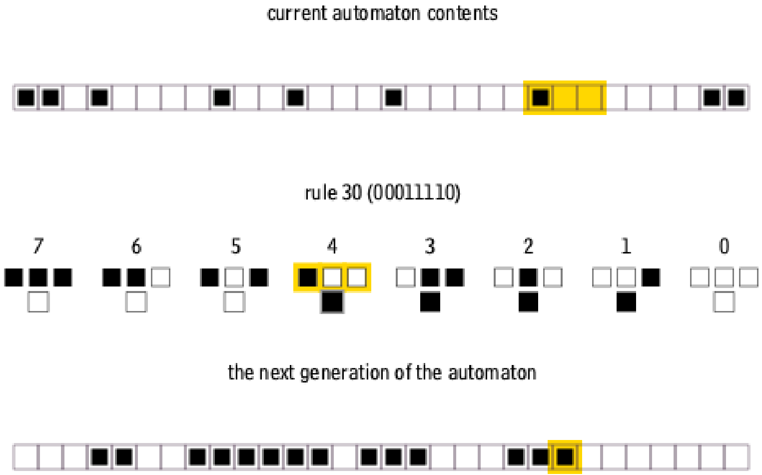
Wall

- **Task:** Simulate run of an elementary celular automata.



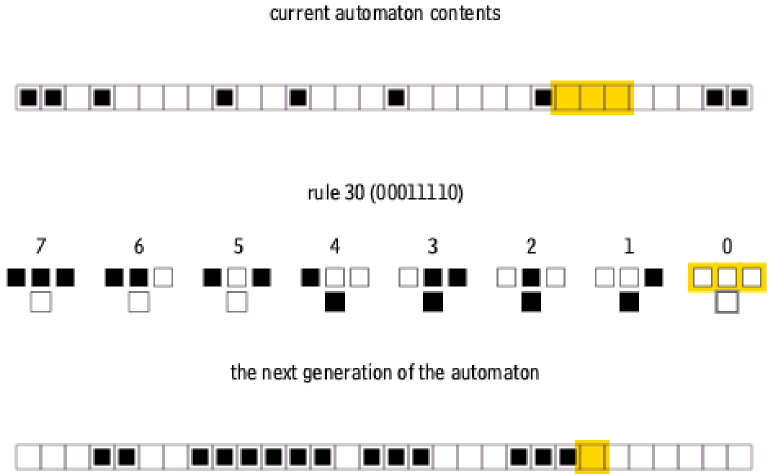
Wall

- **Task:** Simulate run of an elementary celular automata.



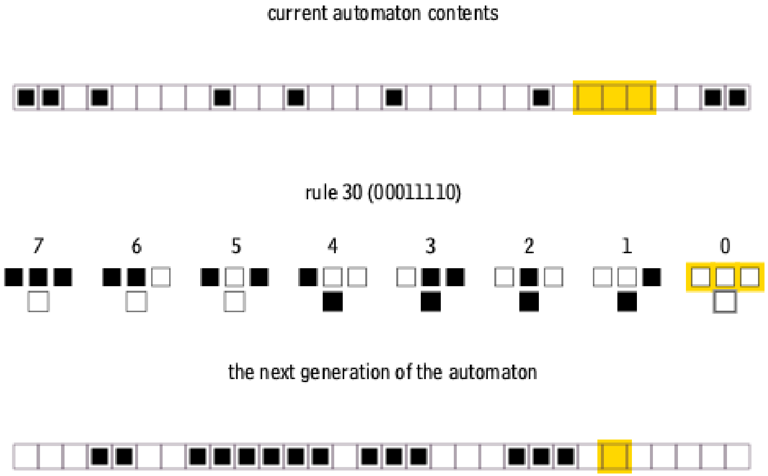
Wall

- **Task:** Simulate run of an elementary celular automata.



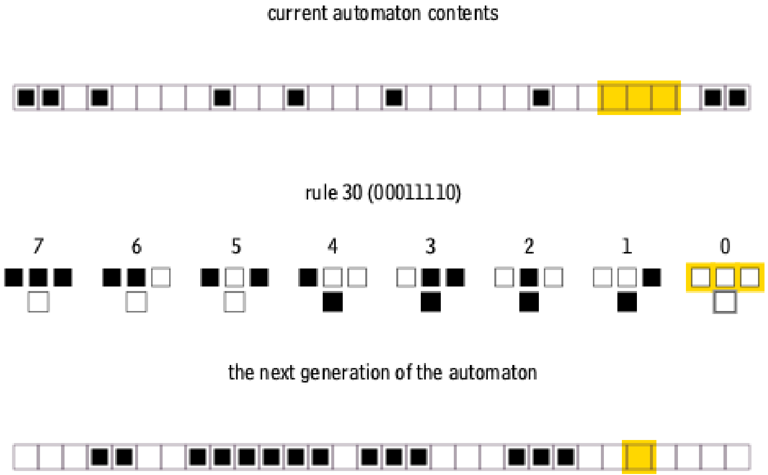
Wall

- **Task:** Simulate run of an elementary celular automata.



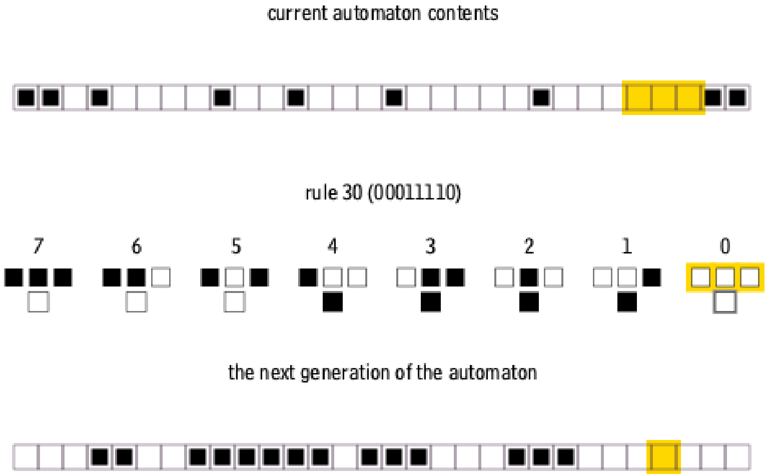
Wall

- **Task:** Simulate run of an elementary celular automata.



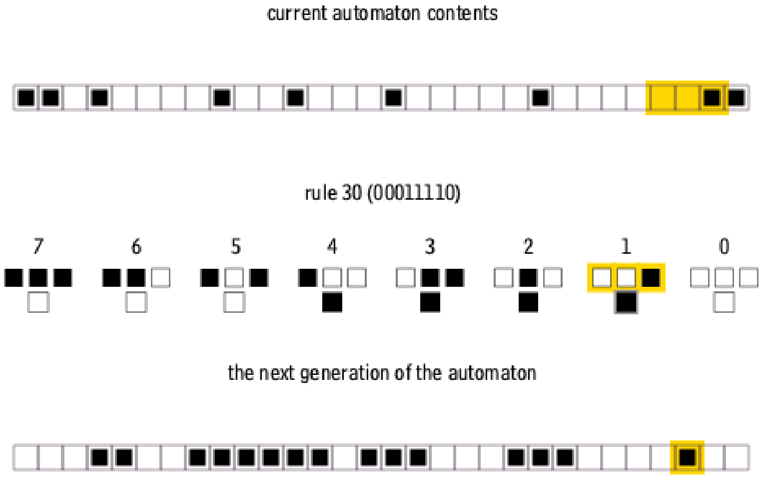
Wall

- **Task:** Simulate run of an elementary celular automata.



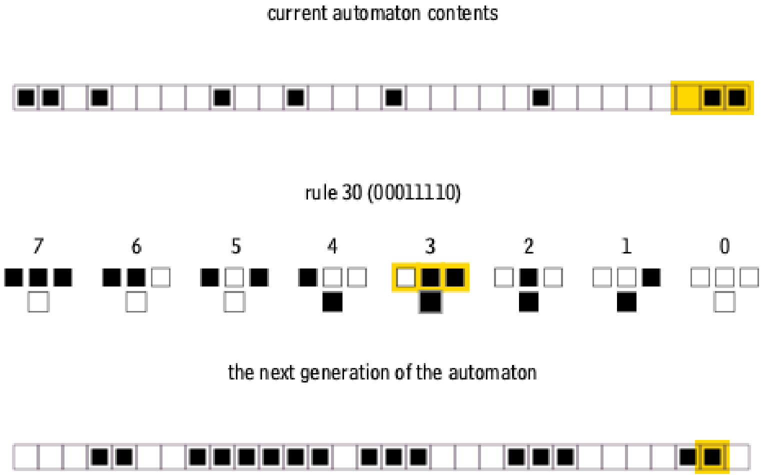
Wall

- **Task:** Simulate run of an elementary celular automata.



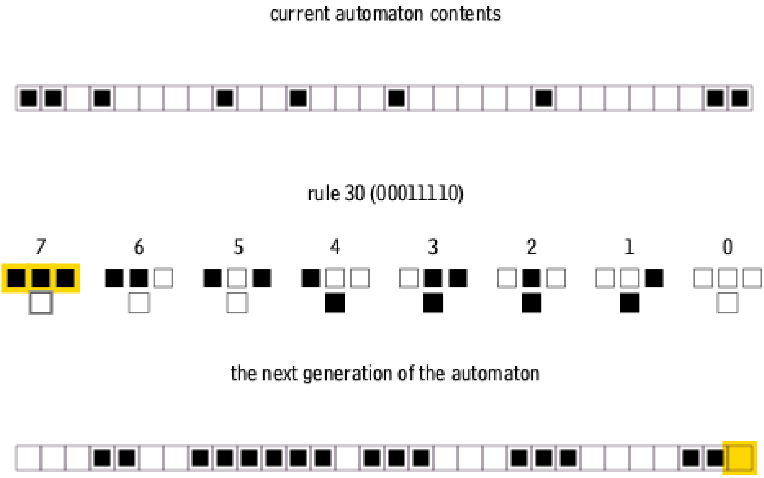
Wall

- **Task:** Simulate run of an elementary celular automata.



Wall

- **Task:** Simulate run of an elementary celular automata.



Beth

Beth's Cookies

- Valid bracket sequence on the input.
- Create an expression with the following rules and evaluate it.
  - `()` $\rightarrow$ `(1)`
  - `)(` $\rightarrow$ `)*(`
  - `))` $\rightarrow$ `)+1)`

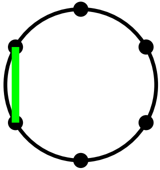
Proglute

Proglute

- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.

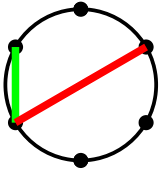
Proglute

- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.



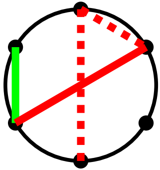
Proglute

- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.



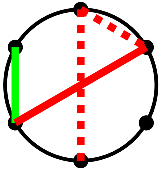
Proglute

- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.



Proglute

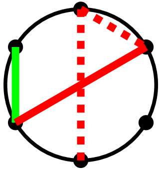
- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.



- Therefore, for any such fixed subpath containing at most $N-2$ points we have two possibilities how to extend the subpath.

Proglute

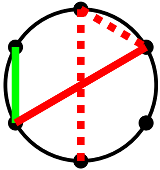
- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.



- Therefore, for any such fixed subpath containing at most $N-2$ points we have two possibilities how to extend the subpath.
- The path can start in any point but it is not oriented.

Proglute

- **Task:** There are N points on a circle, connect them all with a path that does not cross itself.
- Observation: Any subpath containing an end of the path contains only consecutive points.

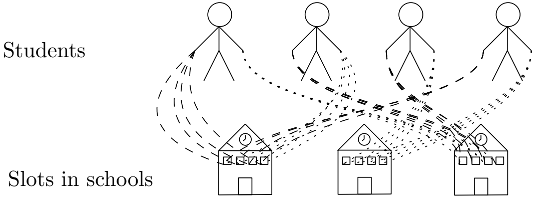


- Therefore, for any such fixed subpath containing at most $N-2$ points we have two possibilities how to extend the subpath.
- The path can start in any point but it is not oriented.
- In total there are $2^{N-2}\frac{N}{2} = 2^{N-3}N$ such paths.

Digitalisation

Digitalisation

- Task: Match students with $M \cdot C$ slots in schools based on preferences on both sides
- **Stable marriage problem**
  - *Stable* = no local improvement possible
  - *Local improvement* = a slot and a student connect, possibly breaking their current ties, to improve the situation for both
  - Known: There exists a matching that is stable



Digitalisation – Does it always finish?

- Write a string, one symbol per student, from best to worst:
  - f for first choice, s for second choice, x for nothing
- Each local improvement makes the string lexicographically smaller
  - A student gets to a better state, a worse student may get worse

Digitalisation – Can it be done faster and simpler?

- Find the lexicograhically smallest string right away
  - Go from the best student to the worst and assign their most preferred choice that's not full yet
- Time $O(N + M)$
  - Faster than the general solution for Stable marriage problem, thanks to all schools having the same preferences

Expressions

Expressions

- First observation: We care about modulo 2 for each element.

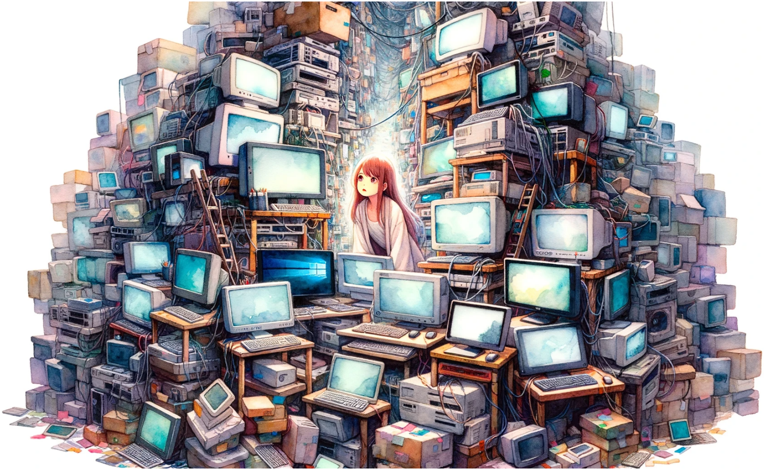
Expressions

- First observation: We care about modulo 2 for each element.
- Second observation: We care for "blocks of products".

Expressions

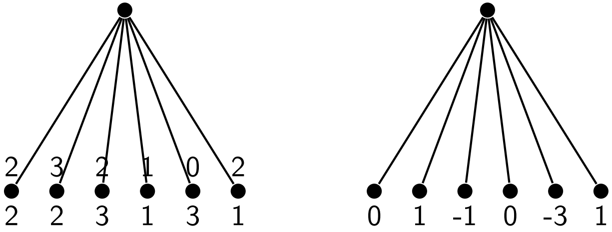
- First observation: We care about modulo 2 for each element.
- Second observation: We care for "blocks of products".
- Preprocess - $\mathcal{O}(n)$, Query - $\mathcal{O}(1)$.

Movers



Movers

- Quickly decide which of two commodities are more prevalent in neighborhood of a vertex.
  $\rightarrow$ *We can keep only info on their difference.*

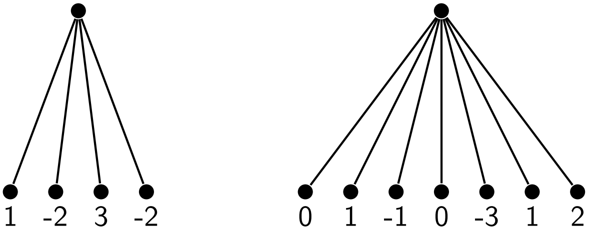


- Quickly update number of commodities in a vertex.

Movers

Let N be the input size:

- Elementary approach – always iterate neighbors $\mathcal{O}(N^2)$
- Faster approach –
  partition vertices by their degrees $\leq \sqrt{N}$ and $> \sqrt{N}$

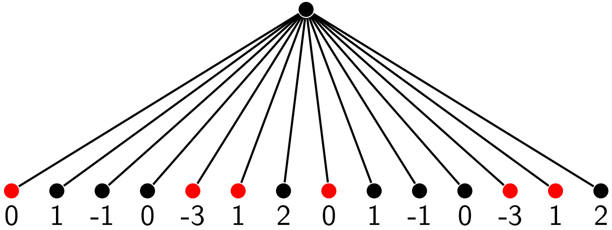


- for small degrees – iterate all neighbors $\mathcal{O}(N\sqrt{N})$
- for big degrees – update its final sum $\mathcal{O}(N\sqrt{N})$

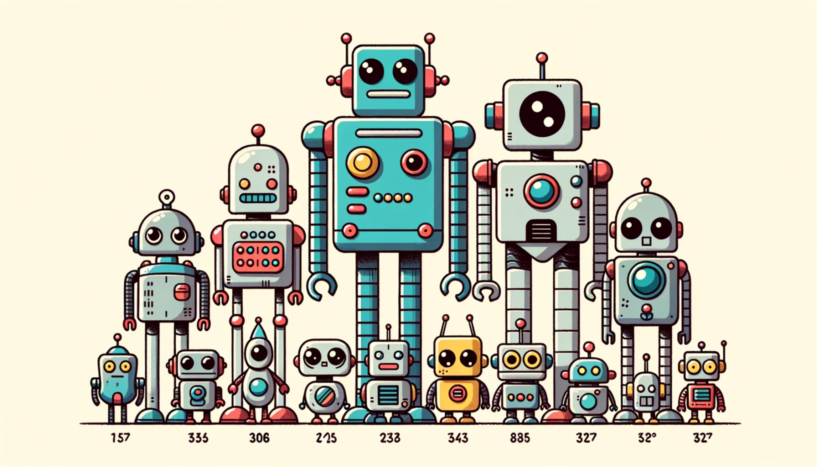
Movers

For big degree vertices:

- keep the final sum in the vertex
- there are at most $\sqrt{N}$ high degree vertices
- update the final sum whenever a neighbor vertex is updated
- $\rightarrow \mathcal{O}(N\sqrt{N})$



Gcd



GCD

- **Task:** Find order of an array A such that $S = \sum_{i=1}^{N-1} \gcd(A_i, A_{i+1})$ is maximized.

GCD

- **Task:** Find order of an array A such that $S = \sum_{i=1}^{N-1} \gcd(A_i, A_{i+1})$ is maximized.
- Observation: Put the same numbers together.
  Suppose the optimal order is $A = \ldots b\mathbf{a}c \ldots x\mathbf{a}y \ldots$ has sum S. We want to show that order $A' = \ldots bc \ldots x\mathbf{aa}y \ldots$ has sum $S' \geq S$.
  - $\gcd(b, a), \gcd(a, c) \leq \frac{a}{2}$:
    Then,
    $S' = S - \gcd(b, a) - \gcd(a, c) + a + \gcd(b, c) > S - \frac{2}{2}a + a \geq S$.
  - $\gcd(b, a) = a$:
    Then $\gcd(b, c) \geq \gcd(a, c)$. Therefore,
    $S' = S - \gcd(b, a) - \gcd(a, c) + a + \gcd(b, c) \geq$
    $S + a - \gcd(b, a) \geq S$.

GCD

- **Task:** Find order of an array A such that $S = \sum_{i=1}^{N-1} \text{gcd}(A_i, A_{i+1})$ is maximized.
- Observation: Put the same numbers together.
  Suppose the optimal order is $A = \ldots b\mathbf{a}c \ldots x\mathbf{a}y \ldots$ has sum S. We want to show that order $A' = \ldots bc \ldots x\mathbf{aa}y \ldots$ has sum $S' \geq S$.
  - $\text{gcd}(b, a), \text{gcd}(a, c) \leq \frac{a}{2}$:
    Then,
    $S' = S - \text{gcd}(b, a) - \text{gcd}(a, c) + a + \text{gcd}(b, c) > S - \frac{2}{2}a + a \geq S.$
  - $\text{gcd}(b, a) = a$:
    Then $\text{gcd}(b, c) \geq \text{gcd}(a, c)$. Therefore,
    $S' = S - \text{gcd}(b, a) - \text{gcd}(a, c) + a + \text{gcd}(b, c) \geq$
    $S + a - \text{gcd}(b, a) \geq S.$
- We can reduce the instance to only consider one of each numbers in A. There are at most 20 such numbers.

GCD

- Finding the solution is equal of solving a TSP on a complete graph $G = (V, E)$, where V are the unique values and an edge $\{u, v\} \in E$ has weight $\gcd(u, v)$.

GCD

- Finding the solution is equal of solving a TSP on a complete graph $G = (V, E)$, where V are the unique values and an edge $\{u, v\} \in E$ has weight $\text{gcd}(u, v)$.
- TSP can be solved with a DP in time $(2^{|V|} \cdot |V|^2)$.

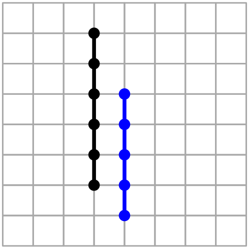
GCD

- Finding the solution is equal of solving a TSP on a complete graph $G = (V, E)$, where V are the unique values and an edge $\{u, v\} \in E$ has weight $\gcd(u, v)$.
- TSP can be solved with a DP in time $(2^{|V|} \cdot |V|^2)$.
- Can be further optimized by putting the number 1 and primes larger than $\frac{N}{2}$ in the front of the array as their GCD with any other number is 1.

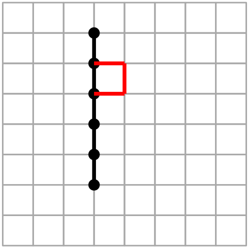
Hamster

Hamster

- **Task:** Given a set of unit length edges connecting some pairs of integer points, how many additional (unit length) edges do we need to create an enclosed region?
- Consider as a graph: vertices are integer points, edges are given on the input.
- First recognize connected components (DFS/BFS).

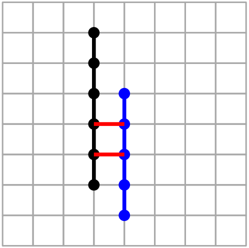


- **3 edges are always enough**: given one edge, use 3 more to create a unit square.

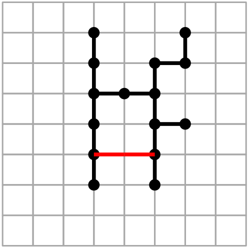


Are 2 edges enough?

- Case 1: We can add 2 edges between two components at different places.

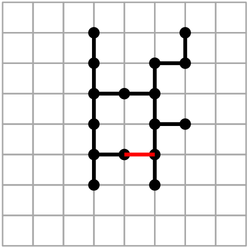


- Case 2: We can add 2 edges, connecting two different vertices of one connected component by a new path.



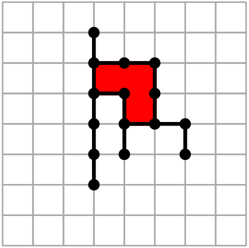
Is 1 edge enough?

- Only case: We can add 1 edge to place without an edge, connecting two different vertices of one connected component.

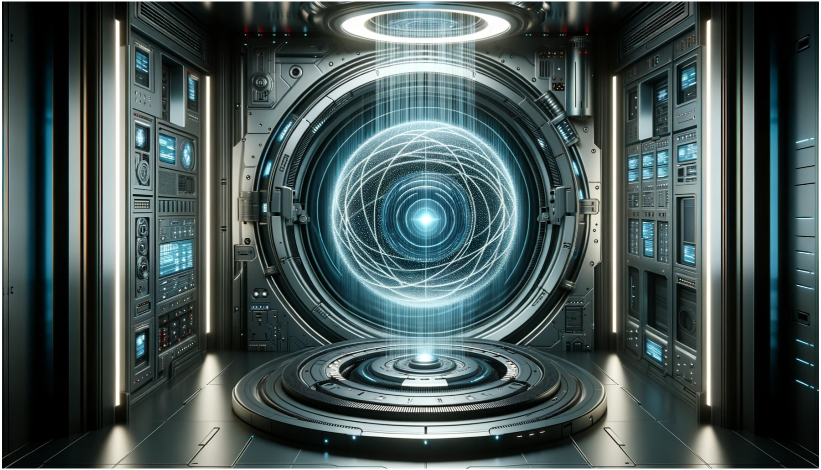


Is 0 edges enough?

- A connected component contains a cycle.



Screamers



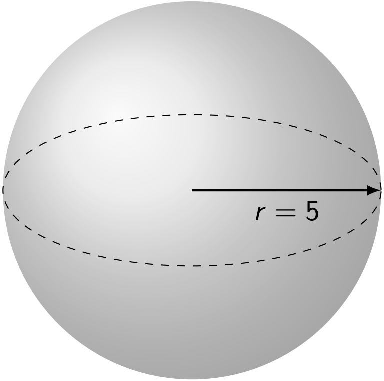
Screamers

- The *cost*(a) of an integer point $a = (a_1, a_2, \ldots, a_d)$ is $cost(a) = |a_1| + |a_2| + \cdots + |a_d|$.
- Given a d-dimensional ball with radius r, compute the sum of costs of all integer points inside it.

Screamers

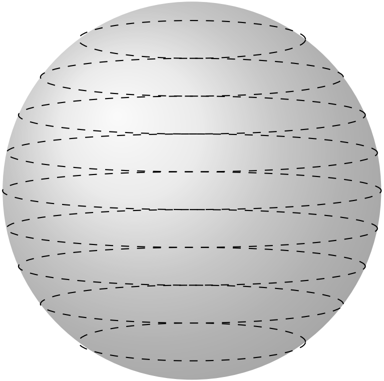
- The *cost*(a) of an integer point $a = (a_1, a_2, \ldots, a_d)$ is $cost(a) = |a_1| + |a_2| + \cdots + |a_d|$.
- Given a d-dimensional ball with radius r, compute the sum of costs of all integer points inside it.
- **First solve subtask**: Count the number of integer points in d-dimensional sphere of radius r.
- **Idea**: Decompose a d-dimensional sphere of radius r into $2r + 1$ $(d - 1)$-dimensional spheres.

Screamers



Screamers

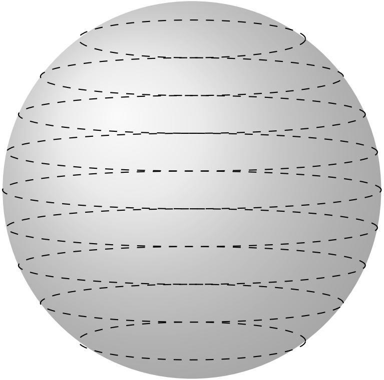
$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $0^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 0^2$

Screamers

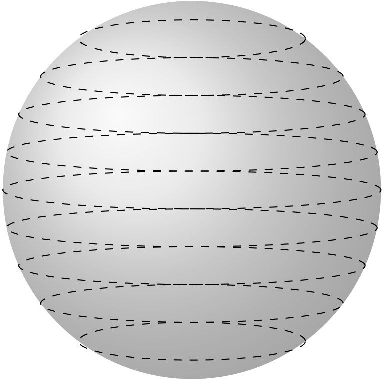
$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $1^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 1^2$

Screamers

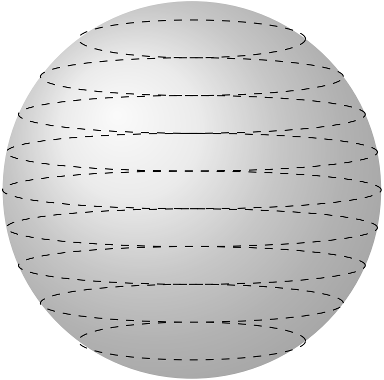
$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $2^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 2^2$

Screamers

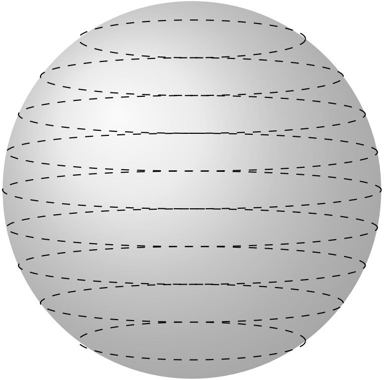
$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $3^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 3^2$

Screamers

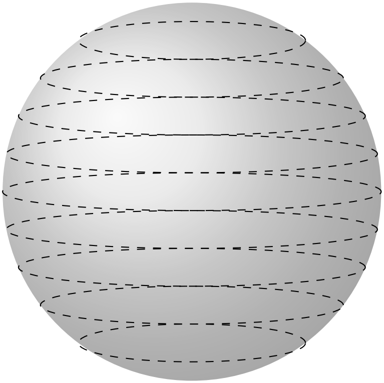
$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $4^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 4^2$

Screamers

$r^2 = 5^2 - 4^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 0^2$
$r^2 = 5^2 - 1^2$
$r^2 = 5^2 - 2^2$
$r^2 = 5^2 - 3^2$
$r^2 = 5^2 - 4^2$



- $x_1^2 + x_2^2 + x_3^2 \leq 5^2$
- $5^2 + x_2^2 + x_3^2 \leq 5^2$
- $x_2^2 + x_3^2 \leq 5^2 - 5^2$

Screamers

- Dynamic programming - parameters: dimension and radius squared.

$$f(d, rs) = \sum_{-\sqrt{rs} \le i \le \sqrt{rs}} f(d-1, rs-i^2)$$

$$f(1, rs) = 1 + 2\lfloor \sqrt{rs} \rfloor$$

- Extending to counting the costs is simple:

$$g(d, rs) = \sum_{-\sqrt{rs} \le i \le \sqrt{rs}} g(d-1, rs-i^2) + |i| \cdot f(d-1, rs-i^2)$$

$$g(1, rs) = 2\lfloor \sqrt{rs}(\sqrt{rs}+1)/2 \rfloor$$

- This DP is computed in $\mathcal{O}(dr^3)$.

Clubbing

Clubbing

- Lets firstly ensure that we can answer queries "does a set contain ANY club?"!

Clubbing

- Lets firstly ensure that we can answer queries "does a set contain ANY club?"!
- We can represent each club as bitmask. And for each mask we are able to precalculate sub-masks it contains in $\mathcal{O}(2^{|U|})$.

Clubbing

- Lets firstly ensure that we can answer queries "does a set contain ANY club?"!
- We can represent each club as bitmask. And for each mask we are able to precalculate sub-masks it contains in $\mathcal{O}(2^{|U|})$.
- Now we can iterate over all "minimal" substrings (with two pointers) and "keep" the set of clubs in it:$\mathcal{O}(L)$

Fragmentation

Fragmentation

- **Input**: array $a_1, a_2, \ldots, a_n$ of n numbers, $a_i \leq 10^6$, $n \leq 10^5$.
- **Task**: For each query s, t, k, find out if k divides the product $a_s \cdot a_{s+1} \cdot a_{s+2} \cdot \cdots \cdot a_{t-1} \cdot a_t$.
- First factorize all a_i. For instance using the Eratosthenes sieve, keeping track of the least prime divider.
- For each prime $p \leq 10^6$, keep sorted array of the indices where it appears.
- Answer each query in $\mathcal{O}(log(a_i) \log(n))$:
- Use binary search to count, how many times each prime appears in the interval.
- Check if each prime appears at least as many times in the product, as it appears in k.

Fragmentation

- **Input**: $2, 3, 6, 12, 4, 8, 16, 4$.
- Primes in input: $2, 3$. Indices where primes are found:
- $2: 0, 2, 3, 3, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 7$.
- $3: 1, 2, 3$.

Fragmentation

- **Input**: $2, \mathbf{3, 6, 12}, 4, 8, 16, 4$.
- Primes in input: $2, 3$. Indices where primes are found:
- $2 : 0, 2, 3, 3, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 7$.
- $3 : 1, 2, 3$.
- **Query**: $s = 1, t = 3, k = 72 = 2^3 \cdot 3^2$.

Fragmentation

- **Input**: $2, \mathbf{3, 6, 12}, 4, 8, 16, 4$.
- Primes in input: $2, 3$. Indices where primes are found:
- $2 : 0, \mathbf{2, 3, 3}, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 7$.
- $3 : \mathbf{1, 2, 3}$.
- **Query**: $s = 1, t = 3, k = 72 = 2^3 \cdot 3^2$.

Fragmentation

- **Input**: $2,$ **3, 6, 12**$, 4, 8, 16, 4$.
- Primes in input: $2, 3$. Indices where primes are found:
- $2: 0,$ **2, 3, 3**$, 4, 4, 5, 5, 5, 6, 6, 6, 6, 7, 7$.
- $3:$ **1, 2, 3**.
- **Our product is $2^3 \cdot 3^3$, thus it is divisible by** k.
- **Query**: $s = 1, t = 3, k = 72 = 2^3 \cdot 3^2$.

Thank you for your attention!