Winter Contest 2024 Presentation of Solutions

The Winter Contest Jury

January 29, 2024

Winter Contest 2024 Jury

- **Philipp Fischbeck**
  Hasso-Plattner-Institute Potsdam
- **Rudolf Fleischer**
  Heinrich-Heine-University Düsseldorf, CPUlm
- **Brutenis Gliwa**
  University of Rostock
- **Niko Hastrich**
  Hasso-Plattner-Institute Potsdam
- **Florian Kothmeier**
  Friedrich-Alexander University
  Erlangen-Nürnberg
- **Felicia Lucke**
  Fribourg University CH, CPUlm
- **Jannik Olbrich**
  Ulm University, CPUlm
- **Erik Sünderhauf**
  Technical University of Munich
- **Christopher Weyand**
  Karlsruhe Institute of Technology, CPUlm
- **Paul Wild**
  Friedrich-Alexander University
  Erlangen-Nürnberg, CPUlm
- **Wendy Yi**
  Karlsruhe Institute of Technology
- **Michael Zündorf**
  Karlsruhe Institute of Technology, CPUlm

Winter Contest 2024 Test Solvers

- **Sebastian Angrick**
  Hasso-Plattner-Institute Potsdam
- **Michael Ruderer**
  Augsburg University, CPUlm
- **Jonas Schmidt**
  Hasso-Plattner-Institute Potsdam

Winter Contest 2024 Technical Team

- **Nathan Maier**
  CPUlm
- **Alexander Schmid**
  CPUlm
- **Pascal Weber**
  University of Vienna, CPUlm

Winter Contest 2022 Presentation of Solutions

January 29, 2022

Winter Contest 2022 Jury

- **Felicia Lucke**
  CPUlm
- **Nathan Maier**
  CPUlm
- **Jannik Olbrich**
  CPUlm
- **Gregor Schwarz**
  Technical University of Munich
- **Marcel Wienöbst**
  University of Lübeck
- **Paul Wild**
  Friedrich–Alexander University
  Erlangen–Nürnberg
- **Michael Zündorf**
  Karlsruhe Institute of Technology

Big thanks to our test solvers

- **Gregor Matl**

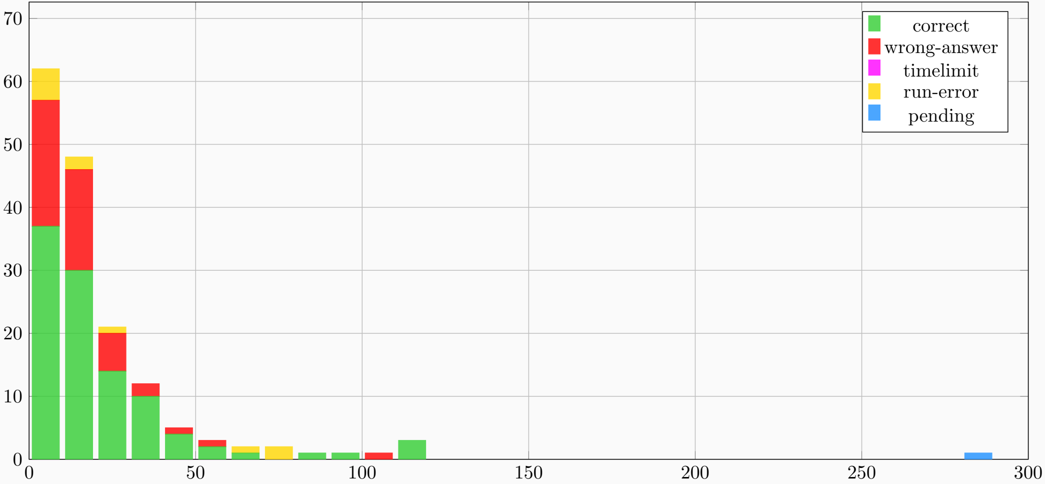
  Technical University of Munich

- **Michael Ruderer**

  CPUlm

A: Alphabetical Athletes

Problem Author: Felicia Lucke



A: Alphabetical Athletes

Problem Author: Felicia Lucke

Problem

Given a German word, check if its letters are lexicographically sorted (increasing or decreasing).

A: Alphabetical Athletes

Problem Author: Felicia Lucke

Problem

Given a German word, check if its letters are lexicographically sorted (increasing or decreasing).

Solution

- Sort the word and check if it is equal to the input or the reversed input.

A: Alphabetical Athletes

Problem Author: Felicia Lucke

Problem

Given a German word, check if its letters are lexicographically sorted (increasing or decreasing).

Solution

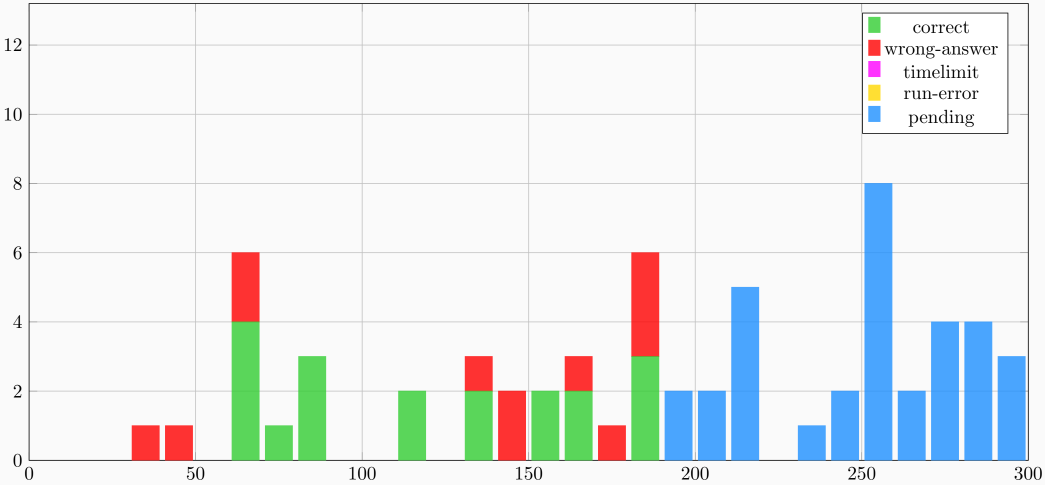
- Sort the word and check if it is equal to the input or the reversed input.

Possible Pitfalls

- The first letter may be capitalized.
- Reversed alphabetical order is considered sorted.
- Did not test all samples.

B: Bright Beacons

Problem Author: Brutenis Gliwa



B: Bright Beacons

Problem Author: Brutenis Gliwa

Problem

Given a grid of mountain heights, what is the shortest path from the top-left to the bottom-right when adjacency is determined by line-of-sight between mountains?

B: Bright Beacons

Problem Author: Brutenis Gliwa

Problem

Given a grid of mountain heights, what is the shortest path from the top-left to the bottom-right when adjacency is determined by line-of-sight between mountains?

Solution

- Compute line of sight function $f(x) : ax + b$ for each pair of mountains along the same row or column ($f(x)$ crosses both peaks).
- There is no line of sight if any mountain in between is higher than $f(x)$ at that position.
- Create a graph: each mountain is a node, add edge between mountains if there is a line of sight.
- Traverse graph with breadth-first-search.

C: Cellar Chase

Problem Author: Felicia Lucke, Jannik Olbrich

Problem

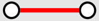
Given a two-terminal-series-parallel (TTSP) graph G, find the size of a maximum cut that separates the graph into exactly two components such that two specified vertices s and t are in different components of the graph.

C: Cellar Chase

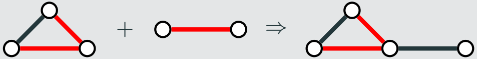
Problem Author: Felicia Lucke, Jannik Olbrich

Solution

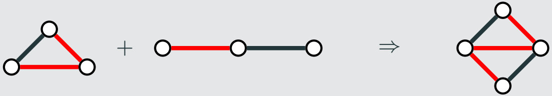
- For a graph G denote by $\mathrm{cut}(G)$ the maximum size of a cut as defined above.
- Use the recursive structure of the graph:
  - If G is “()”, $\mathrm{cut}(G) = 1$.

  - If G is $A + B$, where A and B are both TTSP, then $\mathrm{cut}(G) = \max(\mathrm{cut}(A), \mathrm{cut}(B))$.

  - If G is $A * B$, where A and B are both TTSP, then $\mathrm{cut}(G) = \mathrm{cut}(A) + \mathrm{cut}(B)$.

- Calculate the size of the cut recursively.

D: Document Dimensions

Problem Author: Michael Zündorf

Problem

Given a text with n words separated by spaces with total length W, replace some spaces with newlines such that the total height plus width of the text is minimized.

D: Document Dimensions

Problem Author: Michael Zündorf

Problem

Given a text with n words separated by spaces with total length W, replace some spaces with newlines such that the total height plus width of the text is minimized.

Solution

- For a given width w we can find the minimal height greedily by only adding newlines when needed.
- The next position where a newline is needed can be found in $\mathcal{O}(1)$ with a prefix sum over the lengths of the words.
- Therefore, the minimal height can be found in $\mathcal{O}(\frac{W}{w})$.

D: Document Dimensions

Problem Author: Michael Zündorf

Problem

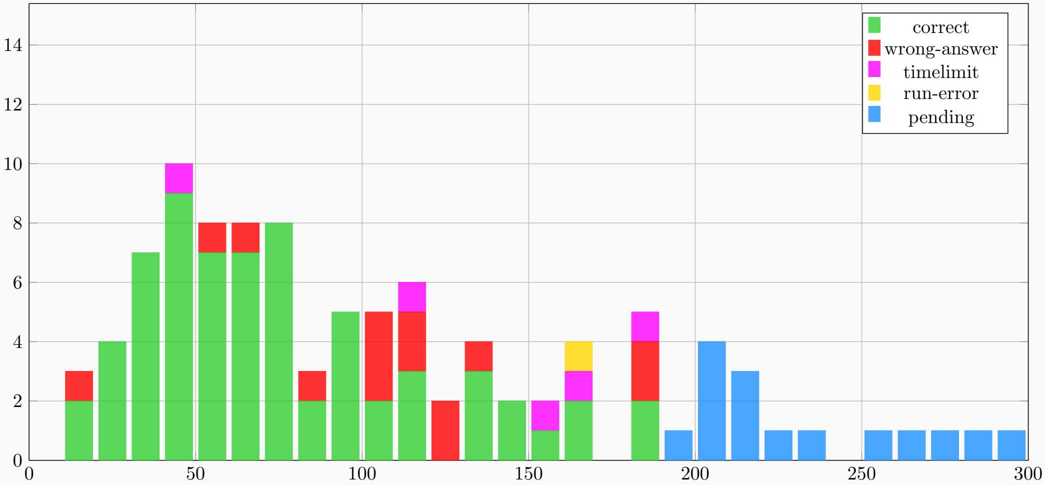
Given a text with n words separated by spaces with total length W, replace some spaces with newlines such that the total height plus width of the text is minimized.

Solution

- For a given width w we can find the minimal height greedily by only adding newlines when needed.
- The next position where a newline is needed can be found in $\mathcal{O}(1)$ with a prefix sum over the lengths of the words.
- Therefore, the minimal height can be found in $\mathcal{O}(\frac{W}{w})$.
- Calculating this for every width is in $\mathcal{O}(W \log(W))$.

E: Euroexpress

Problem Author: Michael Zündorf

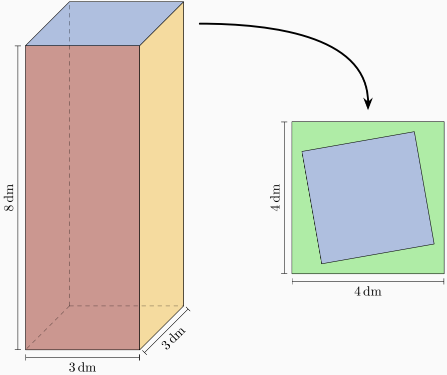


E: Euroexpress

Problem Author: Michael Zündorf

Problem

Given n rectangles (w_i, h_i), find the largest box where each side can be covered by one of the rectangles.



E: Euroexpress

Problem Author: Michael Zündorf

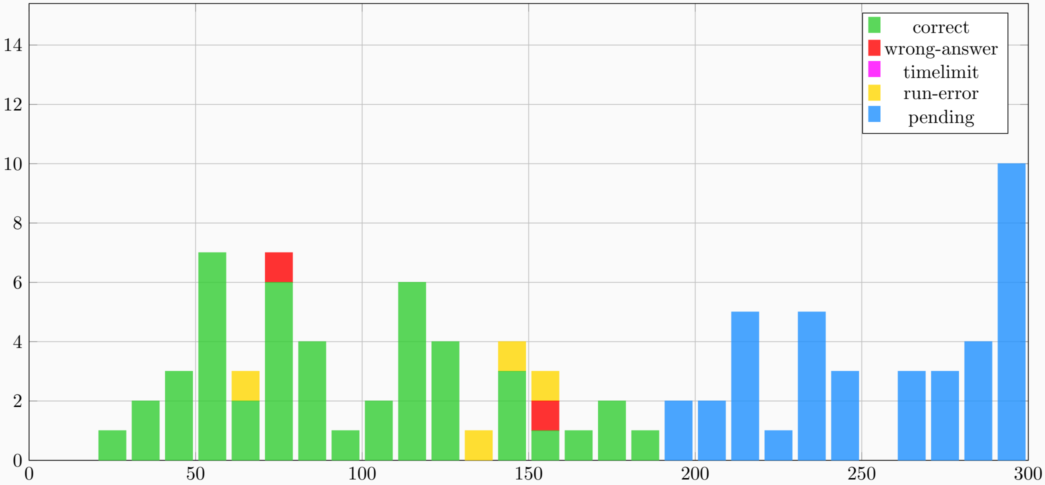
Solution

- All sides of the largest box can always be covered with the same rectangle.
- For a given rectangle, the largest box has size $w \times h \times \min(w, h)$.
- Try all rectangles and take the maximum over all.

$\Rightarrow$ Runtime: $\mathcal{O}(n)$

F: Football Figurines

Problem Author: Rudolf Fleischer



F: Football Figurines

Problem Author: Rudolf Fleischer

Problem

- Given are n floors where stairs go either one or two levels up, and m queries that consist of two floors each.
- For each query, compute the total number of staircases used on all possible different routes between the two queried floors modulo $10^9 + 7$.

F: Football Figurines

Problem Author: Rudolf Fleischer

Problem

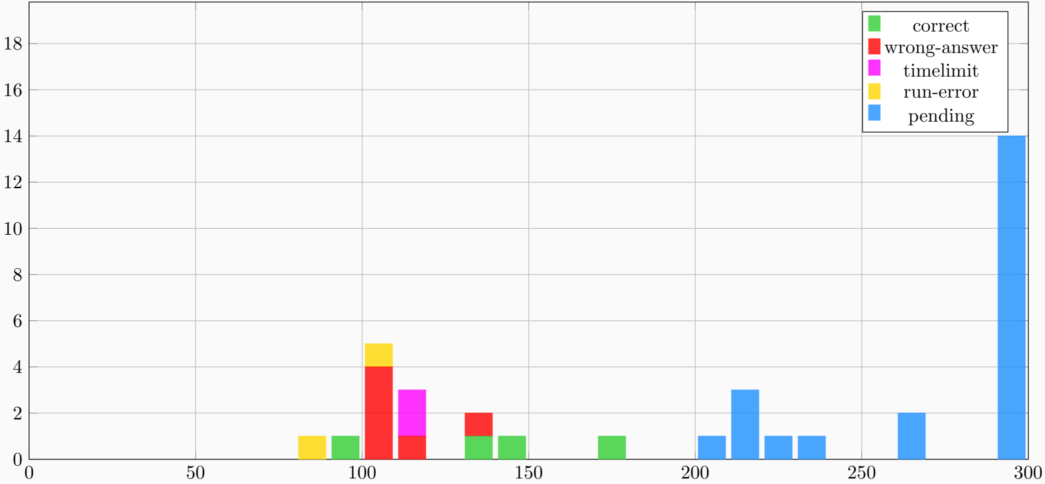
- Given are n floors where stairs go either one or two levels up, and m queries that consist of two floors each.
- For each query, compute the total number of staircases used on all possible different routes between the two queried floors modulo $10^9 + 7$.

Solution

- The number of routes to climb up k floors is the kth Fibonacci number F_k.
- The total number of staircases used is $L_k = L_{k-1} + L_{k-2} + F_k$, where $L_0 = 0$ and $L_1 = 1$.

G: Genius Gamer

Problem Author: Niko Hastrich



G: Genius Gamer

Problem Author: Niko Hastrich

Problem

Given tiles with a color and a numerical value (without duplicates), decide wether they can be partitioned into sets of size at least three that either

- share the same numerical value (group), or
- share the same colour and have consecutive numerical values (run).

G: Genius Gamer

Problem Author: Niko Hastrich

Problem

Given tiles with a color and a numerical value (without duplicates), decide wether they can be partitioned into sets of size at least three that either

- share the same numerical value (group), or
- share the same colour and have consecutive numerical values (run).

Solution

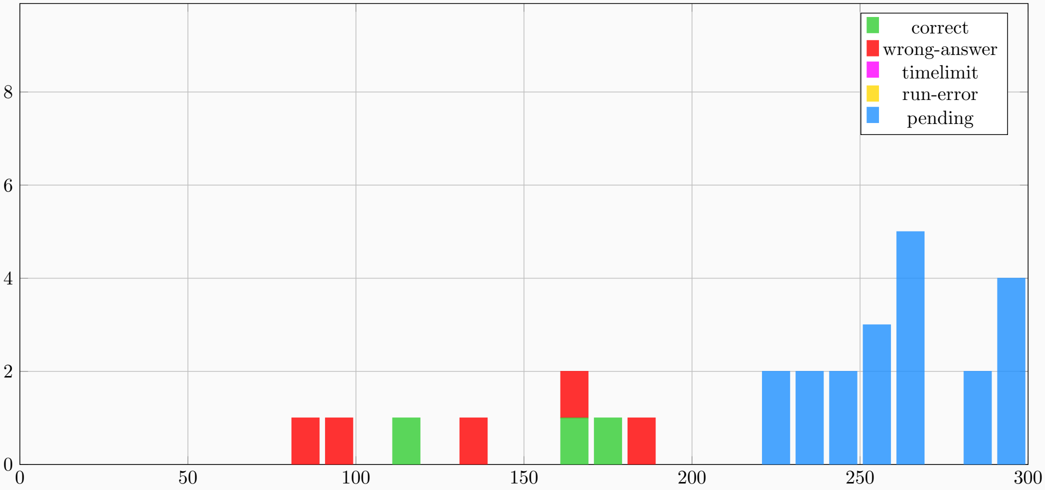
- Solvable via dynamic programming.

$$DP[i][a][b][c][d] = \begin{array}{l}\text{Is it possible to partition the pieces with value at most } i\text{, such that in the} \\ \text{first colour there ends a run of size } a\text{, in the second of size } b\text{, in the third} \\ \text{of size } c\text{, and in the last of size } d \text{ with the tile of value } i\text{.}\end{array}$$

- For a, b, c and d only states $\{0, 1, 2, \text{``} \geq 3\text{''}\}$ are interesting.
- Needs $\mathcal{O}(4^4 \max(\text{numerical value}))$ states, with amortized constant time transition.
- Due to small constraints alternative solutions possible (e.g. back-tracking, meet-in-the-middle).

H: Haggling over Hours

Problem Author: Felicia Lucke



H: Haggling over Hours

Problem Author: Felicia Lucke

Problem

Given a set of intervals, what is the smallest number of intervals to delete if you want to reduce the size of the maximum independent set (MIS) by at least 1.

H: Haggling over Hours

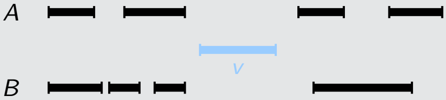
Problem Author: Felicia Lucke

Problem

Given a set of intervals, what is the smallest number of intervals to delete if you want to reduce the size of the maximum independent set (MIS) by at least 1.

Observation

- An inteval v in some MIS has the same number of intervals to the left of it in every MIS.



H: Haggling over Hours

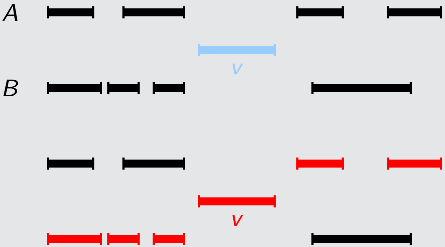
Problem Author: Felicia Lucke

Problem

Given a set of intervals, what is the smallest number of intervals to delete if you want to reduce the size of the maximum independent set (MIS) by at least 1.

Observation

- An inteval v in some MIS has the same number of intervals to the left of it in every MIS.



H: Haggling over Hours

Problem Author: Felicia Lucke

Step 1: Find all intervals contained in some MIS

- For interval v, let left(v) be the size of the MIS to the left of v, similar for right(v).
- Calculate left(v) and right(v) for all intervals using dynamic programming.
- All intervals where left(v)+1+right(v) is maximum are contained in a maximum independent set.

H: Haggling over Hours

Problem Author: Felicia Lucke

Step 1: Find all intervals contained in some MIS

- For interval v, let left(v) be the size of the MIS to the left of v, similar for right(v).
- Calculate left(v) and right(v) for all intervals using dynamic programming.
- All intervals where left(v)$+1+$right(v) is maximum are contained in a maximum independent set.

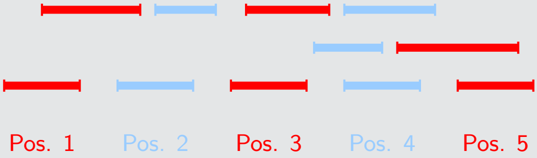
Observation

- For an interval v in an MIS, we say that pos(v) $=$ left(v) $+1$.
- Two intervals at the same position are always intersecting.

Step 2: construct Digraph

- One vertex per interval contained in some maximum independent set
- Add an arc (u, v) for vertices u and v if their corresponding intervals are at consecutive positions and the intervals do not intersect.
- Add a source s and sink vertex t.

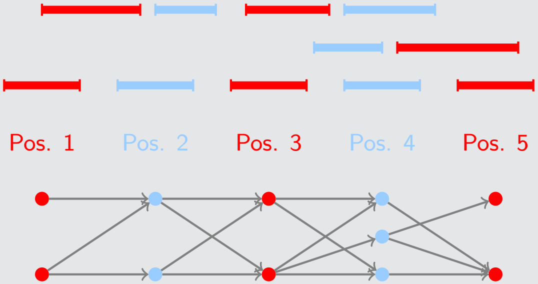
Every maximum independent set corresponds to an (s, t)-path in the graph. The size of a minimum vertex cut is the solution.



Step 2: construct Digraph

- One vertex per interval contained in some maximum independent set
- Add an arc (u, v) for vertices u and v if their corresponding intervals are at consecutive positions and the intervals do not intersect.
- Add a source s and sink vertex t.

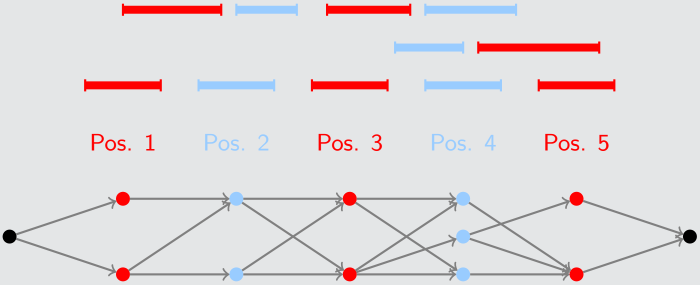
Every maximum independent set corresponds to an (s, t)-path in the graph. The size of a minimum vertex cut is the solution.



Step 2: construct Digraph

- One vertex per interval contained in some maximum independent set
- Add an arc (u, v) for vertices u and v if their corresponding intervals are at consecutive positions and the intervals do not intersect.
- Add a source s and sink vertex t.

Every maximum independent set corresponds to an (s, t)-path in the graph. The size of a minimum vertex cut is the solution.



I: Inconspicuous Identity

Problem Author: Gregor Schwarz

Problem

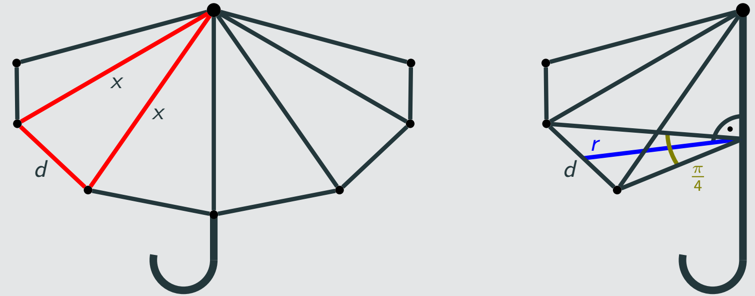
Given a square meters of fabric, compute the maximum area that can be kept dry by an umbrella which has 8 metal sticks of length x meters attached to its top.

I: Inconspicuous Identity

Problem Author: Gregor Schwarz

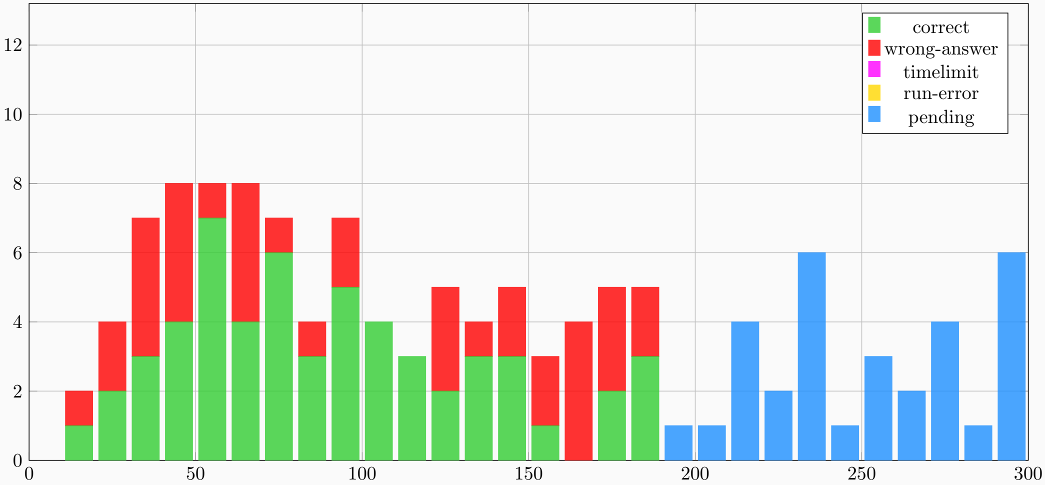
Solution

- Check whether the amount of fabric suffices to open the umbrella all the way (i.e. metals sticks are perpendicular to the handle).
- If not, use binary search or trigonometry to compute the maximum value for d so that the fabric suffices for the umbrella.
- Given d, compute the maximum area using trigonometry.



J: Jog in the Fog

Problem Author: Philipp Fischbeck



J: Jog in the Fog

Problem Author: Philipp Fischbeck

Problem

Given an initial position (x, y) and a looping route of n cells (x_i, y_i) on a 2D grid, find the expected time to reach someone running along the route if using the fastest strategy.

J: Jog in the Fog

Problem Author: Philipp Fischbeck

Problem

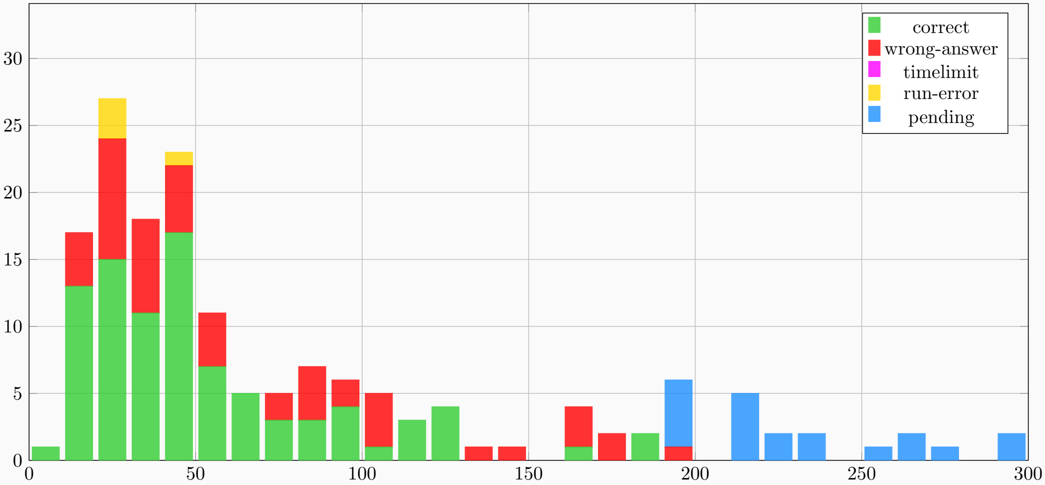
Given an initial position (x, y) and a looping route of n cells (x_i, y_i) on a 2D grid, find the expected time to reach someone running along the route if using the fastest strategy.

Solution

- Optimal strategy: reach the route as fast as possible, then run along the route in opposite direction.
- Reaching the route: $\min_{1 \le i \le n} |x - x_i| + |y - y_i|$
- Running along the route: $\frac{1}{n} \sum_{i=1}^{n} \frac{i-1}{2} = \frac{n-1}{4}$

K: Keeping Keys

Problem Author: Brutenis Gliwa



K: Keeping Keys

Problem Author: Brutenis Gliwa

Problem

Pressing a keyboard key costs 1 cent. What is the cost of printing a text consisting of *a-z*, *A-Z* or *spaces* when you are allowed to hold keys?

K: Keeping Keys

Problem Author: Brutenis Gliwa

Problem

Pressing a keyboard key costs 1 cent. What is the cost of printing a text consisting of *a-z*, *A-Z* or *spaces* when you are allowed to hold keys?

Solution

- Handle **SHIFT** and the letters **a-z␣** of the keyboard separately:

K: Keeping Keys

Problem Author: Brutenis Gliwa

Problem

Pressing a keyboard key costs 1 cent. What is the cost of printing a text consisting of *a-z*, *A-Z* or *spaces* when you are allowed to hold keys?

Solution

- Handle **SHIFT** and the letters **a-z␣** of the keyboard separately:
- **SHIFT**: remove spaces, replace a repeating capital letter with a single capital letter

K: Keeping Keys

Problem Author: Brutenis Gliwa

Problem

Pressing a keyboard key costs 1 cent. What is the cost of printing a text consisting of *a-z*, *A-Z* or *spaces* when you are allowed to hold keys?

Solution

- Handle **SHIFT** and the letters **a-z␣** of the keyboard separately:
- **SHIFT**: remove spaces, replace a repeating capital letter with a single capital letter
- **a-z␣**: *.to_lower()* everything, replace repeating letters with a single letter

K: Keeping Keys

Problem Author: Brutenis Gliwa

Problem

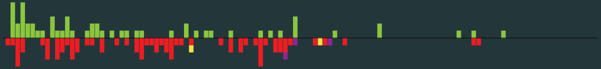
Pressing a keyboard key costs 1 cent. What is the cost of printing a text consisting of *a-z*, *A-Z* or *spaces* when you are allowed to hold keys?

Solution

- Handle **SHIFT** and the letters **a-z␣** of the keyboard separately:
- **SHIFT**: remove spaces, replace a repeating capital letter with a single capital letter
- **a-z␣**: *.to_lower()* everything, replace repeating letters with a single letter
- Print sum of resulting string lengths.

L: Longbottom Leap

Problem Author: Jannik Olbrich

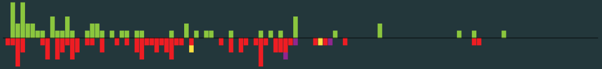


Problem

Given a binary string of length n, find the smallest integer $i \geq 1$ such that $32 \cdot 2^{i-1} \geq n$.

L: Longbottom Leap

Problem Author: Jannik Olbrich



Problem

Given a binary string of length n, find the smallest integer $i \geq 1$ such that $32 \cdot 2^{i-1} \geq n$.

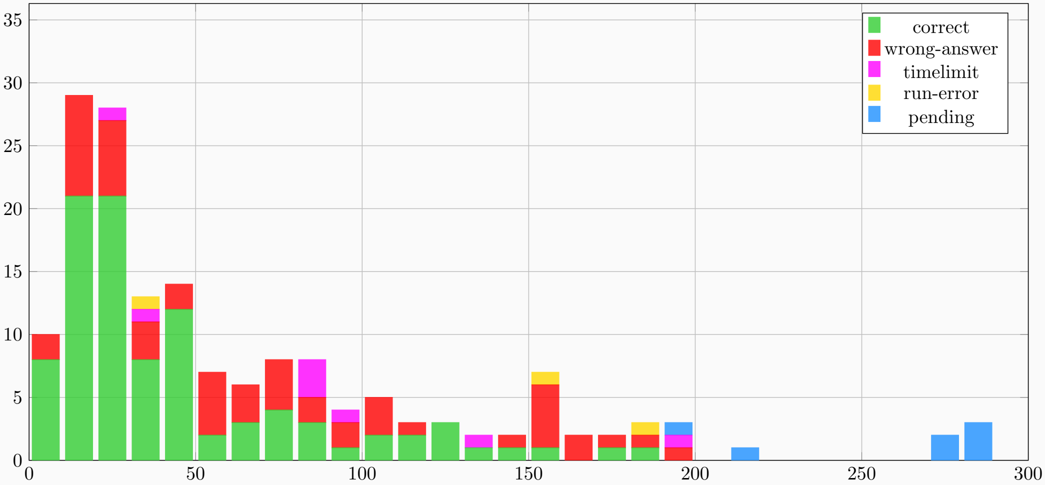
Solution

Start with $i = 1$ and increment i until $32 \cdot 2^{i-1} \geq n$.

Print i times “`long`”.

M: Montage Matrix

Problem Author: Florian Kothmeier



M: Montage Matrix

Problem Author: Florian Kothmeier

Problem

Arrange n people in w columns for a photo.

Constraint: Only people with lower height h_i may stand in front of others.

M: Montage Matrix

Problem Author: Florian Kothmeier

Problem

Arrange n people in w columns for a photo.

Constraint: Only people with lower height h_i may stand in front of others.

Solution 1 – Construct Arrangement

- Sort heights from tallest to smallest and rearrange into $w \times \frac{n}{w}$ grid
- For each entry, check that $h_{i,j} > h_{i,j+1}$
- Alternatively: Use only a single row, and replace items when processed

$\Rightarrow$ Runtime $O(n \cdot log(n))$.

M: Montage Matrix

Problem Author: Florian Kothmeier

Problem

Arrange n people in w columns for a photo.

Constraint: Only people with lower height h_i may stand in front of others.

Solution 1 – Construct Arrangement

- Sort heights from tallest to smallest and rearrange into $w \times \frac{n}{w}$ grid
- For each entry, check that $h_{i,j} > h_{i,j+1}$
- Alternatively: Use only a single row, and replace items when processed

⇒ Runtime $O(n \cdot log(n))$.

Solution 2 – Count Occurrences

- Constraint only fails if the person standing in front has the same height.
- This is only possible, when there are more than w people with the same height.

⇒ Can be computed in $O(n)$ by using HashMaps.

- Beware of off-by-one errors, e.g. exactly w people with the same height.