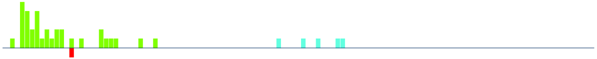
BAPC 2021 Preliminaries

Solutions presentation

October 9, 2021

D: Dickensian Dictionary

Problem Author: Mees de Vries

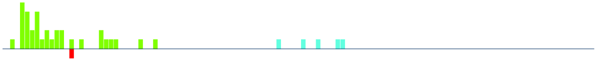


- **Problem:** Given a word, decide if it is *Dickensian*
  (i.e., typeable alternatingly with left and right hand)

Statistics: 39 submissions, 33 accepted, 5 unknown

D: Dickensian Dictionary

Problem Author: Mees de Vries

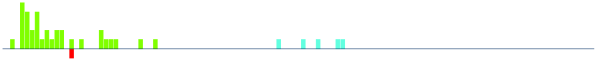


- **Problem:** Given a word, decide if it is *Dickensian*
  (i.e., typeable alternatingly with left and right hand)
- Solution: Check for every letter whether it is typeable with left or right

Statistics: 39 submissions, 33 accepted, 5 unknown

D: Dickensian Dictionary

Problem Author: Mees de Vries

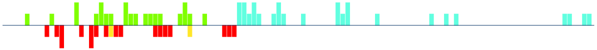


- **Problem:** Given a word, decide if it is *Dickensian*
  (i.e., typeable alternatingly with left and right hand)
- Solution: Check for every letter whether it is typeable with left or right
- Check if the resulting list is alternating
  - Note that you can start with either left or right

Statistics: 39 submissions, 33 accepted, 5 unknown

F: Fridge Distraction

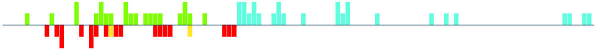
Problem Author: Robin Lee



- **Problem:** Keep Kevin busy by asking him to take items from his long fridge. Ask for as little items as possible.

Statistics: 68 submissions, 22 accepted, 26 unknown

F: Fridge Distraction



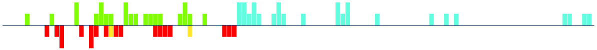
Problem Author: Robin Lee

- **Problem:** Keep Kevin busy by asking him to take items from his long fridge.
  Ask for as little items as possible.
- Solution: Keep asking items from the very back of the fridge
  - Maintaining a rotating list of items will do
  - For the last item, pick one from the middle, based on the number of seconds left

Statistics: 68 submissions, 22 accepted, 26 unknown

F: Fridge Distraction

Problem Author: Robin Lee

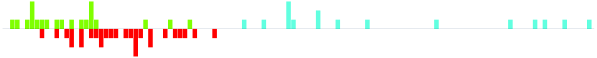


- **Problem:** Keep Kevin busy by asking him to take items from his long fridge.
  Ask for as little items as possible.
- Solution: Keep asking items from the very back of the fridge
  - Maintaining a rotating list of items will do
  - For the last item, pick one from the middle, based on the number of seconds left
- Optimization: You don't need to maintain a list if you do some math

Statistics: 68 submissions, 22 accepted, 26 unknown

B: Buffered Buffet

Problem Author: Boas Kluiving

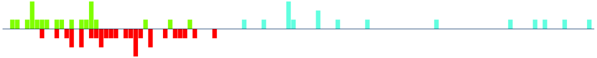


- **Problem:** What is the minimum circumference of the table such that everyone can comply with their social distancing requirement?

Statistics: 65 submissions, 21 accepted, 16 unknown

B: Buffered Buffet

Problem Author: Boas Kluiving

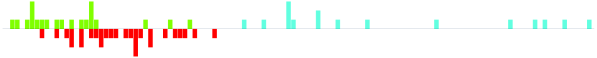


- **Problem:** What is the minimum circumference of the table such that everyone can comply with their social distancing requirement?
- Solution: Order guests by their required distance, then you can sum up their distances to get the table circumference.

Statistics: 65 submissions, 21 accepted, 16 unknown

B: Buffered Buffet

Problem Author: Boas Kluiving

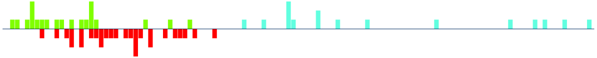


- **Problem:** What is the minimum circumference of the table such that everyone can comply with their social distancing requirement?
- Solution: Order guests by their required distance, then you can sum up their distances to get the table circumference.
  - The guest requiring the smallest distance is always satisfied on both sides, so this guest should not be counted.
  - The guest requiring the largest distance, requires this distance on both sides, so this guest should be counted twice.
  - All other guests are automatically satisfied on the side where somebody with lesser requirements is sitting, so they only need to be counted once.

Statistics: 65 submissions, 21 accepted, 16 unknown

B: Buffered Buffet

Problem Author: Boas Kluiving

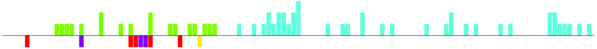


- **Problem:** What is the minimum circumference of the table such that everyone can comply with their social distancing requirement?
- Solution: Order guests by their required distance, then you can sum up their distances to get the table circumference.
  - The guest requiring the smallest distance is always satisfied on both sides, so this guest should not be counted.
  - The guest requiring the largest distance, requires this distance on both sides, so this guest should be counted twice.
  - All other guests are automatically satisfied on the side where somebody with lesser requirements is sitting, so they only need to be counted once.
- Final answer: $\sum_i d_i + \max_i(d_i) - \min_i(d_i)$

Statistics: 65 submissions, 21 accepted, 16 unknown

H: Histogram

Problem Author: Abe Wits

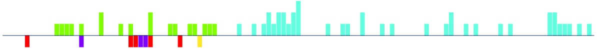


- **Problem:** Print a histogram with the given data.

Statistics: 67 submissions, 18 accepted, 40 unknown

H: Histogram

Problem Author: Abe Wits

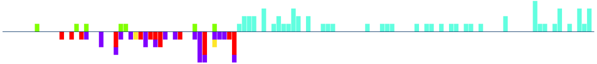


- **Problem:** Print a histogram with the given data.
- Solution: First count the size for each bin, then print the histogram.
  - Make sure to calculate the height of the histogram beforehand

Statistics: 67 submissions, 18 accepted, 40 unknown

I: Ice Growth

Problem Author: Jorke de Vlas

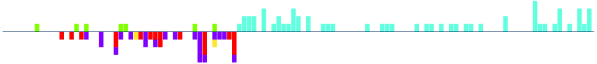


- Given a weather report for n days and k people that have a required minimal ice thickness, how many days can each person skate?

Statistics: 106 submissions, 7 accepted, 58 unknown

I: Ice Growth

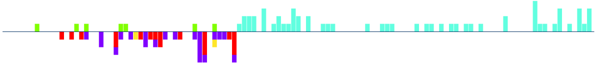
Problem Author: Jorke de Vlas



- Given a weather report for n days and k people that have a required minimal ice thickness, how many days can each person skate?
- Compute and store the ice thickness for each day.
  - Ice-thickness can't be negative.
  - Use integers to count 'degrees of frost'.

Statistics: 106 submissions, 7 accepted, 58 unknown

I: Ice Growth



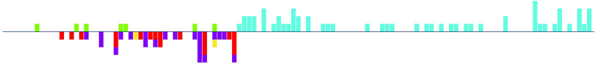
Problem Author: Jorke de Vlas

- Given a weather report for n days and k people that have a required minimal ice thickness, how many days can each person skate?
- Compute and store the ice thickness for each day.
  - Ice-thickness can't be negative.
  - Use integers to count 'degrees of frost'.
- Sort the days by ice thickness [$\mathcal{O}(n \log(n))$].

Statistics: 106 submissions, 7 accepted, 58 unknown

I: Ice Growth

Problem Author: Jorke de Vlas

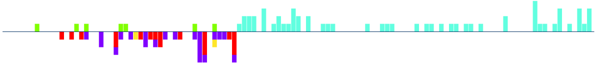


- Given a weather report for n days and k people that have a required minimal ice thickness, how many days can each person skate?
- Compute and store the ice thickness for each day.
  - Ice-thickness can't be negative.
  - Use integers to count 'degrees of frost'.
- Sort the days by ice thickness [$\mathcal{O}(n \log(n))$].
- For each person binary search how many days have the required thickness [$\mathcal{O}(k \log(n))$].

Statistics: 106 submissions, 7 accepted, 58 unknown

I: Ice Growth

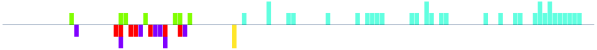
Problem Author: Jorke de Vlas



- Given a weather report for n days and k people that have a required minimal ice thickness, how many days can each person skate?
- Compute and store the ice thickness for each day.
  - Ice-thickness can't be negative.
  - Use integers to count 'degrees of frost'.
- Sort the days by ice thickness $[\mathcal{O}(n \log(n))]$.
- For each person binary search how many days have the required thickness $[\mathcal{O}(k \log(n))]$.
- Alternative: store the number of days for each ice-thickness $\leq 10^6$, and accumulate once $[\mathcal{O}(k + n)]$.

Statistics: 106 submissions, 7 accepted, 58 unknown

C: Candy Contribution



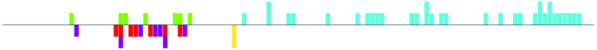
Problem Author: Ruben Brokkelkamp

- **Problem:** Given a graph, nodes s and t, a number of candies c and for each edge e an integer p_e denoting what percentage of the candies you are carrying you have to pay to use the edge (rounded up).

Statistics: 57 submissions, 7 accepted, 34 unknown

C: Candy Contribution

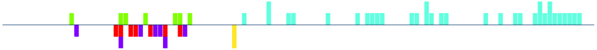
Problem Author: Ruben Brokkelkamp



- **Problem:** Given a graph, nodes s and t, a number of candies c and for each edge e an integer p_e denoting what percentage of the candies you are carrying you have to pay to use the edge (rounded up).
  - What is the maximum number of candies you can bring from s to t?

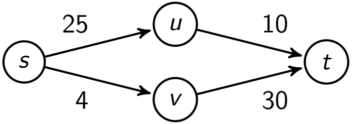
Statistics: 57 submissions, 7 accepted, 34 unknown

C: Candy Contribution



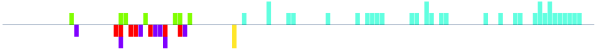
Problem Author: Ruben Brokkelkamp

- **Problem:** Given a graph, nodes s and t, a number of candies c and for each edge e an integer p_e denoting what percentage of the candies you are carrying you have to pay to use the edge (rounded up).
  - What is the maximum number of candies you can bring from s to t?
- Sample showed that computing path with lowest summed taxed percentage is not always best: $(1-0.25)(1-0.1) = 0.675 > 0.672 = (1-0.04)(1-0.3)$.



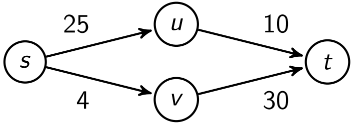
Statistics: 57 submissions, 7 accepted, 34 unknown

C: Candy Contribution



Problem Author: Ruben Brokkelkamp

- **Problem:** Given a graph, nodes s and t, a number of candies c and for each edge e an integer p_e denoting what percentage of the candies you are carrying you have to pay to use the edge (rounded up).
  - What is the maximum number of candies you can bring from s to t?
- Sample showed that computing path with lowest summed taxed percentage is not always best: $(1-0.25)(1-0.1) = 0.675 > 0.672 = (1-0.04)(1-0.3)$.



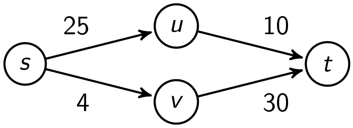
- So, cannot do a 'normal' additive dijkstra with tax percentages to find best path.

Statistics: 57 submissions, 7 accepted, 34 unknown

C: Candy Contribution

Problem Author: Ruben Brokkelkamp

- **Problem:** Given a graph, nodes s and t, a number of candies c and for each edge e an integer p_e denoting what percentage of the candies you are carrying you have to pay to use the edge (rounded up).
  - What is the maximum number of candies you can bring from s to t?
- Sample showed that computing path with lowest summed taxed percentage is not always best: $(1 - 0.25)(1 - 0.1) = 0.675 > 0.672 = (1 - 0.04)(1 - 0.3)$.

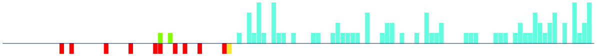


- So, cannot do a 'normal' additive dijkstra with tax percentages to find best path.
- Solution: Tweak dijkstra a bit. Instead of initializing every node to ∞ and lowering it everytime you find a shorter path. Initialize everything to 0 and raise it when you find a path where you hold on to more candies.

Statistics: 57 submissions, 7 accepted, 34 unknown

G: Git mv

Problem Author: Ragnar Groot Koerkamp

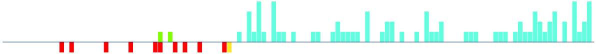


- **Problem:** Given a file movement $s_1/s_2/\ldots/s_n \rightarrow t_1/t_2/\ldots/t_m$ find the shortest move description, assuming that the s_i are distinct and the t_j are distinct.

Statistics: 91 submissions, 2 accepted, 78 unknown

G: Git mv

Problem Author: Ragnar Groot Koerkamp



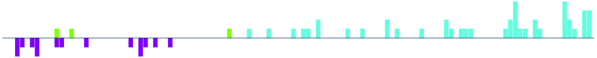
- **Problem:** Given a file movement $s_1/s_2/\ldots/s_n \to t_1/t_2/\ldots/t_m$ find the shortest move description, assuming that the s_i are distinct and the t_j are distinct.
- **Solution:** Greedy, i.e. find smallest i such that $s_i \neq t_i$ and smallest j s.t. $s_{n-j} \neq t_{m-j}$. Output:

$$s_1/s_2/\ldots/s_{i-1}/\left\{s_i/\ldots/s_{n-j} \Longrightarrow t_i/\ldots/t_{m-j}\right\}/s_{n-j+1}/\ldots/s_n.$$

Statistics: 91 submissions, 2 accepted, 78 unknown

A: Almost Always

Problem Author: Ragnar Groot Koerkamp

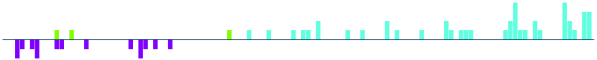


- **Problem:** Given $n = 5 \cdot 10^5$ integers between 1 and $a = 2 \cdot 10^9$, find two such that one divides the other.

Statistics: 62 submissions, 3 accepted, 44 unknown

A: Almost Always

Problem Author: Ragnar Groot Koerkamp

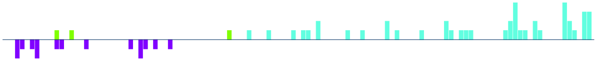


- **Problem:** Given $n = 5 \cdot 10^5$ integers between 1 and $a = 2 \cdot 10^9$, find two such that one divides the other.
- Naive solution: For each pair try whether x_i divides x_j. $\mathcal{O}(n^2)$ is too slow.

Statistics: 62 submissions, 3 accepted, 44 unknown

A: Almost Always

Problem Author: Ragnar Groot Koerkamp

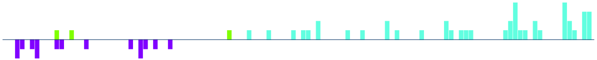


- **Problem:** Given $n = 5 \cdot 10^5$ integers between 1 and $a = 2 \cdot 10^9$, find two such that one divides the other.
- Naive solution: For each pair try whether x_i divides x_j. $\mathcal{O}(n^2)$ is too slow.
- Early break: stop as soon as you find a good pair. $\mathcal{O}(a/\ln(a)) \approx \mathcal{O}(10^8)$ expected steps is likely still too slow on the worst of the 100 test cases.

Statistics: 62 submissions, 3 accepted, 44 unknown

A: Almost Always

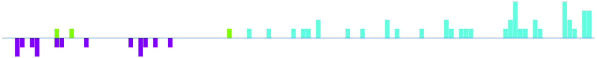
Problem Author: Ragnar Groot Koerkamp



- Observation: small numbers are more likely to divide another number.

A: Almost Always

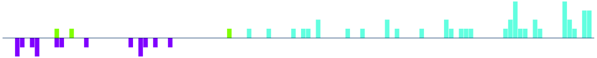
Problem Author: Ragnar Groot Koerkamp



- Observation: small numbers are more likely to divide another number.
- Greedy solution: Sort the input before doing the brute force with early break.

A: Almost Always

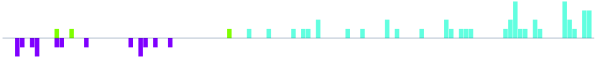
Problem Author: Ragnar Groot Koerkamp



- Observation: small numbers are more likely to divide another number.
- Greedy solution: Sort the input before doing the brute force with early break.
- Single pass solution: Keep the index of the smallest number seen so far, and check whether it divides the current number.

A: Almost Always

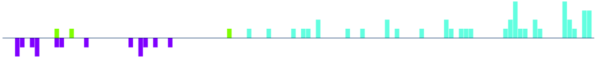
Problem Author: Ragnar Groot Koerkamp



- Observation: small numbers are more likely to divide another number.
- Greedy solution: Sort the input before doing the brute force with early break.
- Single pass solution: Keep the index of the smallest number seen so far, and check whether it divides the current number.
- Analysis:
  The expected value of the smallest integer is $s \approx a/n = 4000$, so likely below 8000.
  The probability that none of the $n = 5 \cdot 10^5$ integers is a multiple of $s \leq 8000$ is less than 10^{-27}.
  If s does not work, we just try the next smallest integer. (But the probability of needing this is 10^{-5}, so only trying the smallest one is sufficient.)

A: Almost Always

Problem Author: Ragnar Groot Koerkamp



- Bonus solution: Use the birthday paradox.
  The probability that all numbers in the list are distinct is only $7 \cdot 10^{-28}$, so we can just find and print the indices of two equal numbers.

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices

Statistics: 44 submissions, 1 accepted, 42 unknown

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices
- Challenge: the number of vertices is at most 2^{10^6}

Statistics: 44 submissions, 1 accepted, 42 unknown

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices
- Challenge: the number of vertices is at most 2^{10^6}
- Observation: the number of removed vertices is 2^{age}
  unless some subtree is already removed

Statistics: 44 submissions, 1 accepted, 42 unknown

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices
- Challenge: the number of vertices is at most 2^{10^6}
- Observation: the number of removed vertices is 2^{age}
  unless some subtree is already removed
- Therefore: remember for every vertex how many children are removed
  and propagate this value to ancestors

Statistics: 44 submissions, 1 accepted, 42 unknown

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices
- Challenge: the number of vertices is at most 2^{10^6}
- Observation: the number of removed vertices is 2^{age}
  unless some subtree is already removed
- Therefore: remember for every vertex how many children are removed
  and propagate this value to ancestors
- Solution: when removing vertex v with age i, return:

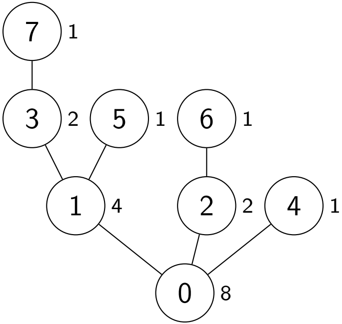
$$2^i - removed[v] \mod 10^9 + 7$$

Statistics: 44 submissions, 1 accepted, 42 unknown

K: Kudzu Kniving

Problem Author: Reinier Schmiermann

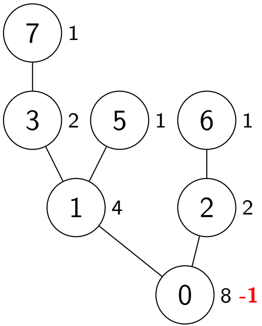
- **Problem:** count the number of removed vertices
- Solution: remember for every vertex how many children are removed



K: Kudzu Kniving

Problem Author: Reinier Schmiermann

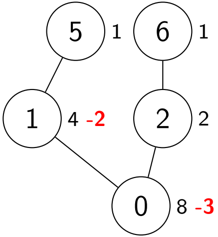
- **Problem:** count the number of removed vertices
- Solution: remember for every vertex how many children are removed



K: Kudzu Kniving

Problem Author: Reinier Schmiermann

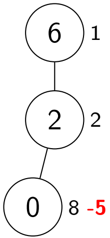
- **Problem:** count the number of removed vertices
- Solution: remember for every vertex how many children are removed



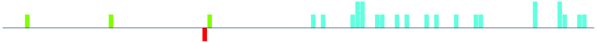
K: Kudzu Kniving

Problem Author: Reinier Schmiermann

- **Problem:** count the number of removed vertices
- Solution: remember for every vertex how many children are removed



J: Jack the Mole



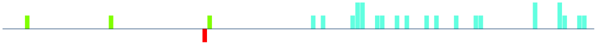
Problem Author: Pim Spelier

- **Problem:** Given n integers between 1 and 1000, find which of them can be removed such that the remainder can be partitioned into two sets of equal sum.

Statistics: 27 submissions, 3 accepted, 23 unknown

J: Jack the Mole

Problem Author: Pim Spelier

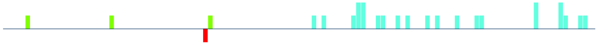


- **Problem:** Given n integers between 1 and 1000, find which of them can be removed such that the remainder can be partitioned into two sets of equal sum.
- Define $s := n \cdot w$ to be the sum of the integers.

Statistics: 27 submissions, 3 accepted, 23 unknown

J: Jack the Mole

Problem Author: Pim Spelier

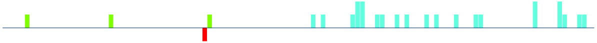


- **Problem:** Given n integers between 1 and 1000, find which of them can be removed such that the remainder can be partitioned into two sets of equal sum.
- Define $s := n \cdot w$ to be the sum of the integers.
- $\mathcal{O}(n^2 \cdot s) = \mathcal{O}(n^3 \cdot w)$ solution: For each mole run a $\mathcal{O}(n \cdot s)$ knapsack to check if a partitioning is possible.
  This is usually too slow, unless using bitsets in C++.

Statistics: 27 submissions, 3 accepted, 23 unknown

J: Jack the Mole

Problem Author: Pim Spelier



- **Problem:** Given n integers between 1 and 1000, find which of them can be removed such that the remainder can be partitioned into two sets of equal sum.
- Define $s := n \cdot w$ to be the sum of the integers.
- $\mathcal{O}(n^2 \cdot s) = \mathcal{O}(n^3 \cdot w)$ solution: For each mole run a $\mathcal{O}(n \cdot s)$ knapsack to check if a partitioning is possible.
  This is usually too slow, unless using bitsets in C++.
- $\mathcal{O}(n^2 \cdot w)$ solution:
  - For each prefix of moles, compute all possible weights of a subset in $\mathcal{O}(n \cdot s)$.
  - For each suffix of moles, compute all possible weights of a subset in $\mathcal{O}(n \cdot s)$.
  - Mole i can be left out if it is possible to make a subset of size l with the moles before i, and a subset of size $(s - w_i)/2 - l$ of the moles after i, for some l.

Statistics: 27 submissions, 3 accepted, 23 unknown

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?

Statistics: 20 submissions, 0 accepted, 20 unknown

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Observation: The exact scores of the players do not matter, only their difference does.

Statistics: 20 submissions, 0 accepted, 20 unknown

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Observation: The exact scores of the players do not matter, only their difference does.
- Idea: Use DP to find the score difference in remainder of the game, for every game state, assuming optimal play.

Statistics: 20 submissions, 0 accepted, 20 unknown

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Observation: The exact scores of the players do not matter, only their difference does.
- Idea: Use DP to find the score difference in remainder of the game, for every game state, assuming optimal play.
- Issue: $\mathcal{O}(3^n/\sqrt{n})$ game states, too many!

Statistics: 20 submissions, 0 accepted, 20 unknown

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:
  - you lose $\frac{1}{2}$ point for every unclaimed adjacent vertex

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:
  - you lose $\frac{1}{2}$ point for every unclaimed adjacent vertex
  - you get $\frac{1}{2}$ point for every claimed adjacent vertex

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:
  - you lose $\frac{1}{2}$ point for every unclaimed adjacent vertex
  - you get $\frac{1}{2}$ point for every claimed adjacent vertex
- Results in the same score difference as the original game.

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

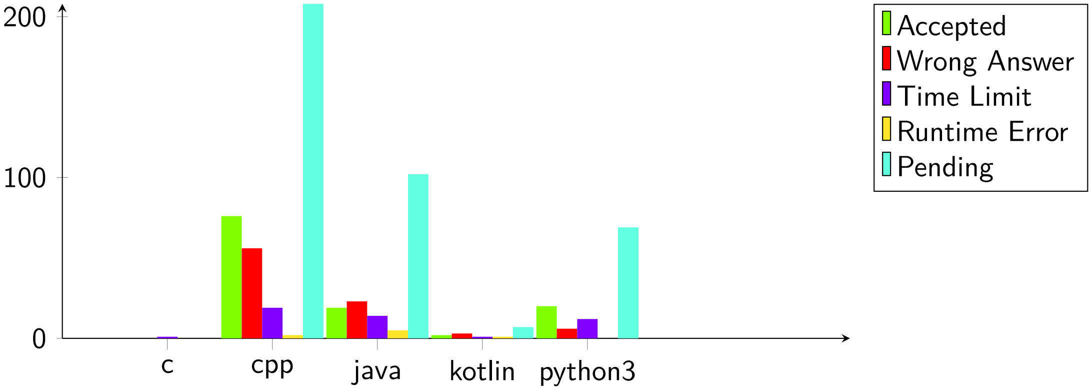
- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:
  - you lose $\frac{1}{2}$ point for every unclaimed adjacent vertex
  - you get $\frac{1}{2}$ point for every claimed adjacent vertex
- Results in the same score difference as the original game.
- Vertices are either "claimed" or "unclaimed", so only $\mathcal{O}(2^n)$ game states.

E: Entering Enemy Encampment

Problem Author: Reinier Schmiermann

- **Problem:** Two players take turns claiming vertices of a graph and get a point every time they claim a vertex adjacent to an enemy vertex. Who wins?
- Consider an alternative game, where after claiming a vertex:
  - you lose $\frac{1}{2}$ point for every unclaimed adjacent vertex
  - you get $\frac{1}{2}$ point for every claimed adjacent vertex
- Results in the same score difference as the original game.
- Vertices are either "claimed" or "unclaimed", so only $\mathcal{O}(2^n)$ game states.
- Using a subset DP: $\mathcal{O}(2^n \cdot n^2)$ time needed.

Language stats



Some stats

- 347 commits (last year: 527)
- 437 secret testcases (last year: 360)
- 175 jury solutions (last year: 221)
- The minimum number of lines the jury needed to solve all problems is

$$2 + 2 + 10 + 2 + 20 + 3 + 4 + 4 + 9 + 16 + 10 = 82$$

On average 7.5 lines per problem, down from 13.9 last year

Some tips

- Read the output specification carefully!
- Don't forget to remove debug prints!
- When integers get large, use 64-bit!
- Do not do string concatenation with "+" in a loop!
- Calling functions is more expensive than you might think!

Thanks to the Proofreaders!

Abe Wits
Nicky Gerritsen
Jaap Eldering
Mark van Helvoort
Kevin Verbeek

The Jury

Boas Kluiving
Erik Baalhuis
Freek Henstra
Harry Smit
Joey Haas
Jorke de Vlas
Ludo Pulles
Maarten Sijm
Mees de Vries
Ragnar Groot Koerkamp
Reinier Schmiermann
Robin Lee
Ruben Brokkelkamp
Timon Knigge
Wessel van Woerden