

# Freshmen Programming Contests 2025

Solutions presentation

---

By the Freshmen Programming Contests 2025 jury for:

- AAPJE in Amsterdam
- FPC in Delft
- FYPC in Eindhoven
- GAPC in Groningen
- Contest in Mons

May 3, 2025



Please do not post the problems online

Other universities will have their contests in the coming weeks.

Please, do not post/discuss the problems online before

Saturday 17 May 2025 at 17:00

# A: Array Annihilation

Problem author: Leon van der Waal

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# A: Array Annihilation

Problem author: Leon van der Waal

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# A: Array Annihilation

Problem author: Leon van der Waal

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# A: Array Annihilation

Problem author: Leon van der Waal

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# A: Array Annihilation

Problem author: Leon van der Waal

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.



# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.

**Observation 1:** The bottle packaging can be in 2 possible orientations.

# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.

**Observation 1:** The bottle packaging can be in 2 possible orientations.

**Observation 2:** You can maximally have  $10^{18}$  bottles, hence we need 64-bit integers.

# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.

**Observation 1:** The bottle packaging can be in 2 possible orientations.

**Observation 2:** You can maximally have  $10^{18}$  bottles, hence we need 64-bit integers.

**Solution:** Compute  $w \cdot h - \max(\min(w, a) \cdot \min(h, b), \min(w, b) \cdot \min(h, a))$ .

# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.

**Observation 1:** The bottle packaging can be in 2 possible orientations.

**Observation 2:** You can maximally have  $10^{18}$  bottles, hence we need 64-bit integers.

**Solution:** Compute  $w \cdot h - \max(\min(w, a) \cdot \min(h, b), \min(w, b) \cdot \min(h, a))$ .

**Running time:**  $\mathcal{O}(1)$

# B: Bakfiets

Problem author: Jeroen Op de Beek

**Problem:** Minimize the area of one rectangle that cannot overlap with another.

**Observation 1:** The bottle packaging can be in 2 possible orientations.

**Observation 2:** You can maximally have  $10^{18}$  bottles, hence we need 64-bit integers.

**Solution:** Compute  $w \cdot h - \max(\min(w, a) \cdot \min(h, b), \min(w, b) \cdot \min(h, a))$ .

**Running time:**  $\mathcal{O}(1)$

Statistics: ... submissions, ... accepted, ... unknown

# C: Characterithmetic

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# C: Characterithmetic

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# C: Characterithmetic

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.



# C: Characterithmetic

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# C: Characterithmetic

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

**Observation 3:** Assuming we can split the big tree into  $n$  trees of size 1, the answer is only “impossible” when  $k > n$ .

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

**Observation 3:** Assuming we can split the big tree into  $n$  trees of size 1, the answer is only “impossible” when  $k > n$ .

**Solution:** First calculate the depth of each vertex in the tree using BFS/DFS, then remove vertices one-by-one from largest to smallest depth.

# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

**Observation 3:** Assuming we can split the big tree into  $n$  trees of size 1, the answer is only “impossible” when  $k > n$ .

**Solution:** First calculate the depth of each vertex in the tree using BFS/DFS, then remove vertices one-by-one from largest to smallest depth.

**Red herring:** The first sample cuts off larger AVL trees on purpose.



# D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

**Observation 3:** Assuming we can split the big tree into  $n$  trees of size 1, the answer is only “impossible” when  $k > n$ .

**Solution:** First calculate the depth of each vertex in the tree using BFS/DFS, then remove vertices one-by-one from largest to smallest depth.

**Red herring:** The first sample cuts off larger AVL trees on purpose.

**Running time:** Dominated by sorting by depth:  $\mathcal{O}(n \log n)$ .

## D: Delicious Trees

Problem author: Jeroen Op de Beek

**Problem:** Find any way to cut the AVL tree into some predetermined number of smaller AVL trees, or say this is impossible.

**Observation 1:** An AVL tree with only one vertex, is also an AVL tree.

**Observation 2:** Removing the deepest vertex from the tree can only decrease the depth of the largest of the two subtrees of any ancestor, so this will never introduce imbalanced vertices.

**Observation 3:** Assuming we can split the big tree into  $n$  trees of size 1, the answer is only “impossible” when  $k > n$ .

**Solution:** First calculate the depth of each vertex in the tree using BFS/DFS, then remove vertices one-by-one from largest to smallest depth.

**Red herring:** The first sample cuts off larger AVL trees on purpose.

**Running time:** Dominated by sorting by depth:  $\mathcal{O}(n \log n)$ .

Statistics: ... submissions, ... accepted, ... unknown

# E: Equation Extrapolation

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# E: Equation Extrapolation

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# E: Equation Extrapolation

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# E: Equation Extrapolation

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# E: Equation Extrapolation

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# F: Frog and Princess

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.



# F: Frog and Princess

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# F: Frog and Princess

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# F: Frog and Princess

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# F: Frog and Princess

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# G: Gambler's Dilemma

Problem author: Wietze Koops

**Problem:** Determine whether two playing cards have any of the four given properties.

# G: Gambler's Dilemma

Problem author: Wietze Koops

**Problem:** Determine whether two playing cards have any of the four given properties.

**Solution:** For each property, check whether the cards match it.

# G: Gambler's Dilemma

Problem author: Wietze Koops

**Problem:** Determine whether two playing cards have any of the four given properties.

**Solution:** For each property, check whether the cards match it.

**Pitfall:** Be careful of off-by-one errors when calculating the rank of a card.

# G: Gambler's Dilemma

Problem author: Wietze Koops

**Problem:** Determine whether two playing cards have any of the four given properties.

**Solution:** For each property, check whether the cards match it.

**Pitfall:** Be careful of off-by-one errors when calculating the rank of a card.

**Running time:**  $\mathcal{O}(1)$ .



# G: Gambler's Dilemma

Problem author: Wietze Koops

**Problem:** Determine whether two playing cards have any of the four given properties.

**Solution:** For each property, check whether the cards match it.

**Pitfall:** Be careful of off-by-one errors when calculating the rank of a card.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# H: Hopelessly Hungover

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# H: Hopelessly Hungover

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# H: Hopelessly Hungover

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# H: Hopelessly Hungover

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# H: Hopelessly Hungover

Problem author: Wietze Koops

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# I: Interesting Mountains

Problem author: Mihail Bankov

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# I: Interesting Mountains

Problem author: Mihail Bankov

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!



# I: Interesting Mountains

Problem author: Mihail Bankov

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# I: Interesting Mountains

Problem author: Mihail Bankov

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# I: Interesting Mountains

Problem author: Mihail Bankov

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

# J: Jumbled Keys

Problem author: Arnoud van der Leer

**Problem:** Decipher a message, using a series of mapped words.

# J: Jumbled Keys

Problem author: Arnoud van der Leer

**Problem:** Decipher a message, using a series of mapped words.

**Solution:** Use a map! Process every pair of words, and map every letter in the first word to the letter it corresponds to in the second word.

# J: Jumbled Keys

Problem author: Arnoud van der Leer

**Problem:** Decipher a message, using a series of mapped words.

**Solution:** Use a map! Process every pair of words, and map every letter in the first word to the letter it corresponds to in the second word.

**Edge case:** If 25 letters are mapped, the 26th letter maps to the only letter that has no other letter mapped to it.

# J: Jumbled Keys

Problem author: Arnoud van der Leer

**Problem:** Decipher a message, using a series of mapped words.

**Solution:** Use a map! Process every pair of words, and map every letter in the first word to the letter it corresponds to in the second word.

**Edge case:** If 25 letters are mapped, the 26th letter maps to the only letter that has no other letter mapped to it.

**Running time:**  $\mathcal{O}(n \cdot \ell)$ , where  $\ell$  is the average length of the words.

# J: Jumbled Keys

Problem author: Arnoud van der Leer

**Problem:** Decipher a message, using a series of mapped words.

**Solution:** Use a map! Process every pair of words, and map every letter in the first word to the letter it corresponds to in the second word.

**Edge case:** If 25 letters are mapped, the 26th letter maps to the only letter that has no other letter mapped to it.

**Running time:**  $\mathcal{O}(n \cdot \ell)$ , where  $\ell$  is the average length of the words.

Statistics: ... submissions, ... accepted, ... unknown



# K: Kite Construction

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

# K: Kite Construction

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

# K: Kite Construction

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

# K: Kite Construction

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

# K: Kite Construction

Problem author: Jeroen Op de Beek

**Problem:** Print  $4\sqrt{n}$  with sufficiently many digits.

**Naive solution:** Do something stupid.  $\mathcal{O}(2^n)$  is too slow!

**Smart solution:** Just compute the answer.

**Running time:**  $\mathcal{O}(1)$ .

Statistics: ... submissions, ... accepted, ... unknown

### Jury work

- 265 commits (last year: 447)

---

<sup>1</sup>After codegolfing

### Jury work

- 265 commits (last year: 447)
- 533 secret test cases (last year: 357)

---

<sup>1</sup>After codegolfing

### Jury work

- 265 commits (last year: 447)
- 533 secret test cases (last year: 357)
- 121 accepted jury/proofreader solutions (last year: 120)

---

<sup>1</sup>After codegolfing



### Jury work

- 265 commits (last year: 447)
- 533 secret test cases (last year: 357)
- 121 accepted jury/proofreader solutions (last year: 120)
- The minimum<sup>1</sup> number of lines the jury needed to solve all problems is

$$2 + 1 + 6 + 5 + 1 + 5 + 2 + 2 + 6 + 3 + 6 = 39$$

On average 3.5 lines per problem, down from 6.0 last year

---

<sup>1</sup>After codegolfing

## Thanks to the proofreaders:

- Arnoud van der Leer (TU Delft)
- Dany Sluijk (TU Delft)
- Davina van Meer (Delft)
- Mattia Marziali (RU Groningen)
- Michael Zündorf   
(KIT Karlsruhe / NWERC jury)
- Pavel Kunyavskiy (JetBrains Amsterdam)
- Pierre Vandenhove (UMons)
- Thomas Verwoerd  
(TU Delft,  Kotlin Hero )
- Thore Husfeldt (ITU Copenhagen / BAPC Jury)
- Wendy Yi (KIT Karlsruhe / NWERC jury)

## Thanks to the Jury for the Freshmen Programming Contests:

- Alice Sayutina (VU Amsterdam)
- Angel Karchev (TU Delft)
- Bálint Kollmann (TU Delft)
- Jeroen Op de Beek (TU Delft)
- Leon van der Waal (TU Delft)
- Liudas Staniulis (VU Amsterdam)
- Maarten Sijm (TU Delft)
- Mihail Bankov (TU Delft)
- Moham Balfakeih (TU Delft)
- Wietze Koops (Radboud Nijmegen / RU Groningen)



Want to solve the problems you could not finish?  
Or have friends that like to solve algorithmic problems?

<https://fpcs2025.bapc.eu/>

Saturday 17 May 2025 13:00–17:00

Please, do not post/discuss the problems online before this time!

Excited to participate in the next contest?

Register for the BAPC Preliminaries in September!

Want to organize these contests?

Join the organizing committee!

Want to create programming problems for FPCs next year?

Either join the committee, or contact Maarten Sijm