Jury and Testers

Thanks to the jury:

- Michael Baer
- Alexander Dietsch
- Stefan Kraus
- Simon Rainer
- Philipp Reger
- Paul Wild

Thanks to our test readers:

- Gregor Behnke
- Christian Müller
- Alexander Raß
- Thorsten Wißmann

And many thanks to all the volunteers here and at all other contest sites!
L – Local Etiquette

Problem Author: Stefan Kraus
FAU Winter 2019 Solutions
Problem
You are taking part in a guessing game:

- There is a hidden integer that everybody has to guess.
- The guess with the highest difference loses.
- Guesses must be unique.
- You place your guess last.

Find a guess that ensures you will not lose.
**Problem**

You are taking part in a guessing game:
- There is a hidden integer that everybody has to guess.
- The guess with the highest difference loses.
- Guesses must be unique.
- You place your guess last.

Find a guess that ensures you will not lose.

**Solution**

- Your guess is safe $\Leftrightarrow$ there exist a lower and a higher guess.
- Find the lowest and highest guess and check if there is some guess between them that is still available.
- Time complexity: $O(n^2)$, but $O(n)$ is also possible.
```fortran
PROGRAM SOLUTION
INTEGER LO, HI, A(1000)
LOGICAL FOUND

C READ INPUT
READ(*,*) N, A(1: N)

C FIND MIN AND MAX
LO = 1000000000
HI = 0
DO 10 I = 1, N
   LO = MIN(LO, A(I))
   HI = MAX(HI, A(I))
10 CONTINUE

C LOOK FOR A NUMBER IN [LO, HI] THAT IS NOT IN THE LIST
DO 20 I = LO, HI
   FOUND = .FALSE.
   DO 30 J = 1, N
      IF (A(J) .EQ. I) FOUND = .TRUE.
30 CONTINUE
   IF (FOUND) GOTO 20
   WRITE(*,*) I
   GOTO 40
20 CONTINUE
   WRITE(*,*) 'impossible'
40 END
```

Problem Author: Stefan Kraus
FAU Winter 2019 Solutions
D – Drawing Numbers

Correct
Wrong-answer
Timelimit
Run-error
Compiler-error
No-output
Frozen

Problem Author: Philipp Reger
FAU Winter 2019 Solutions
Problem

How many numbers are needed to complete a $5 \times 5$ bingo card that is partially filled? Find and output such a set of numbers.
**Problem**

How many numbers are needed to complete a $5 \times 5$ bingo card that is partially filled? Find and output such a set of numbers.

**Solution**

- Try all 12 possible rows/columns/diagonals.
- Keep track of the best line.
A – A Race Against Time

Problem Author: Alexander Dietsch

FAU Winter 2019 Solutions
You are given a list of problems to fix, each with a base time $t_i$. The time needed to fix the problems increases over time. What is the minimal time needed to fix all problems?
You are given a list of problems to fix, each with a base time $t_i$. The time needed to fix the problems increases over time. What is the minimal time needed to fix all problems?

Suppose time $s$ has passed and we first solve $t_j$ and then $t_k$:

$$s + t_j + s \cdot \frac{t_j}{100} + 1 + t_k + (s + t_j + s \cdot \frac{t_j}{100} + 1) \cdot \frac{t_k}{100} + 1$$

$$= s + t_j + t_k + \frac{t_j \cdot t_k}{100} + \frac{s \cdot t_j}{100} + \frac{s \cdot t_k}{100} + \frac{s \cdot t_j \cdot t_k}{10000} + \frac{t_k}{100} + 2$$

The contribution of $t_j$ and $t_k$ to the final sum only differs by the term $\frac{t_k}{100}$.

$t_k$ has more influence on the result $\Rightarrow$ choose $t_k \leq t_j$.

Fix the problems by decreasing base time.
Problem

Spell out a word using the symbols of chemical elements:

fauerlangen $\rightarrow$ F Au Er La N Ge N
Problem
Spell out a word using the symbols of chemical elements:
\[
\text{fauerlangen} \rightarrow F \ Au \ Er \ La \ N \ Ge \ N
\]

Solution
- For each element symbol that is a prefix of the word, remove it from the beginning and recursively look for a solution.
- When you reach the end of the word, construct the solution as you return through the call stack.
- If the end is never reached, output impossible.
- To avoid running into the time limit, never process the same position twice (memoization/dynamic programming).
- Time complexity: $O(n)$. 

Problem Author: Paul Wild
FAU Winter 2019 Solutions
Play some games of tic-tac-toe against the computer. You may always make the first move, but you may never lose.
Problem

Play some games of tic-tac-toe against the computer. You may always make the first move, but you may never lose.

Solution

There are essentially two approaches:

- **Backtracking**: Explore the complete game tree and assign a score to each game position (minimax algorithm). Always move to the position with maximum score.

- **Case analysis**: Fix a first move and try to keep the number of cases to consider as low as possible.

Both approaches work fine for this problem, but the second is probably easier to get wrong.
Problem

Given a list of words, group together those that can be the result of encrypting the same message with different keys using the Vigenère cipher.

\[
\begin{align*}
\text{BLI ZZA RD} + 253 & \quad 253 \quad 25 \\
\text{= DQL BED TI} & \quad \text{BLI ZZA RD} + 977 \\
\text{= KSP IGH AK} & \end{align*}
\]
C – Cryptanalysis

Solution

- Idea: Find pairs of words where one can be transformed into the other using Vigenère.
- Processing all pairs is too slow, however.
- Instead, transform all words into some canonical form, e.g. apply a key that changes the first $k$ letters to A.
- This reduces the problem to finding duplicates, which can be done in many ways:
  - Sort all words ⇒ duplicates form blocks – $O(n \cdot |w| \cdot \log(n))$.
  - Maintain a (hash) set data structure with all (unique) words for an efficient existence check – $O(n \cdot |w| \cdot \log(n))$.
  - Add words to a trie data structure – $O(n \cdot |w|)$.
  - ...
B – Bongo Bongo Chess

Problem

Insert one field into a circular array of $n$ numbers to maximize the sum when adding every second element.
Problem

Insert one field into a circular array of $n$ numbers to maximize the sum when adding every second element.

Solution

- Cut open the current circle at an arbitrary position.
- For each position $p$, compute the sum $L[p]$ of every second number to the left. ($\leadsto$ prefix sum)
- Can be done in $O(1)$ for a single position $p$ by using $L[p - 2]$.
- Do the same for the right side $R[p]$.
- Now, when inserting the number at $p$, your team’s rating is $L[p - 2] + R[p + 1]$. Try all possibilities.
- Time complexity: $O(n)$.
- Alternative: Compute the final rating for one position. When going 2 steps to the right, only two animals change teams.
Problem

Given a connected undirected graph, find an order in which to delete its vertices such that the graph always remains connected.
I – Infiltration

Problem
Given a connected undirected graph, find an order in which to delete its vertices such that the graph always remains connected.

Solution
- If the graph is a tree, one possible solution is given by a post-order traversal:

```
1
  2
   3
    4 5
    6
```

\[\rightarrow \quad 4,5,2,6,3,1\]

- If the graph is not a tree, the problem is actually easier because there are more edges to keep the graph connected.
- Find a spanning tree and solve the problem for that tree. The order can also be found without constructing the tree.
Problem

Given a sequence $h_1, \ldots, h_n$ of height values, for each index $i$ find the nearest index $j$ such that $|h_i - h_j| \geq k$. 

Solution

By symmetry, it is enough to solve the case $j \leq i$, $h_j \leq h_i$.

Iterate over the sequence from left to right, maintaining a stack with increasing height values. At each index $i$:

1. Pop all values larger than $h_i$, then push $h_i$.

Because the stack has increasing values, the correct $j$ can be found with a binary search for $h_i - k$.

Combine the results from all four cases for the final answer.
Given a sequence $h_1, \ldots, h_n$ of height values, for each index $i$ find the nearest index $j$ such that $|h_i - h_j| \geq k$.

By symmetry, it is enough to solve the case $j \leq i$, $h_j \leq h_i$.

Iterate over the sequence from left to right, maintaining a stack with increasing height values. At each index $i$:

- Pop all values larger than $h_i$, then push $h_i$.
- Because the stack has increasing values, the correct $j$ can be found with a binary search for $h_i - k$.

Combine the results from all four cases for the final answer.
Problem

Given three center points in the plane, is it possible to draw a diamond shaped area around each of them such that the areas touch in a single meeting point?
Problem
Given three center points in the plane, is it possible to draw a diamond shaped area around each of them such that the areas touch in a single meeting point?

Solution
- The meeting point $s$ must be the corner of at least two of the areas.
- It must be the left/right tip of one area and the top/bottom tip of another.
- The coordinates of $s$ must therefore be coordinates of the three center points.
- Try all combinations of coordinates and check if all areas are non-empty and intersect only in $s$. 
J – Journey Witnesses

Problem Author: Michael Baer  FAU Winter 2019 Solutions
Problem

We are given a directed unweighted graph and photos taken at some nodes at given times. For two adjacent photos, check if a path between the two nodes exists with exactly the length of the time difference $\Delta t$ between the photos.
Problem

We are given a directed unweighted graph and photos taken at some nodes at given times. For two adjacent photos, check if a path between the two nodes exists with exactly the length of the time difference $\Delta t$ between the photos.

Idea

- Regular path finding algorithms are too slow, because the $\Delta t$ can be up to $10^{12}$.
- Instead we need some precomputations.
Solution

- For each node determine all nodes that can be reached in exactly $2^k$ steps for all $0 \leq k \leq \lceil \log_2(10^{12}) \rceil = 40$.
- Each of these *layers* can be computed efficiently from the previous one.
- Using the binary representation of $\Delta t$, we know which layers we need (the order does not matter).
- $\leadsto$ Normal DFS/BFS with one step in each chosen layer.
- Using bit vectors to represent the possible steps can improve performance even more.
- Total time complexity: $O(n^2 \cdot (n + q) \cdot \log(\Delta t_{\text{max}}))$. 

Problem Author: Michael Baer

FAU Winter 2019 Solutions
F – Four-Letter Words

Problem Author: Philipp Reger and Paul Wild

FAU Winter 2019 Solutions
Find a secret word $x$ of length 4 by asking up to 100 questions of the form “is $x$ a substring of $w$?”, where $|w| \leq 10000$. 

Part 2 - Binary search

If a string contains $x$, it must occur in the left or right half. Check if the left half contains $x$. Either way we can eliminate half of the word. Make sure to handle occurrences near the center correctly.

Need $\log_2(10000) \approx 14$ queries.
Problem

Find a secret word $x$ of length 4 by asking up to 100 questions of the form “is $x$ a substring of $w$?”, where $|w| \leq 10000$.

Solution

There are two parts:

1. Find a string that contains the secret word $x$.
2. Use binary search to find out where $x$ occurs.
Problem
Find a secret word $x$ of length 4 by asking up to 100 questions of the form “is $x$ a substring of $w$?, where $|w| \leq 10000$.

Solution
There are two parts:
1. Find a string that contains the secret word $x$.
2. Use binary search to find out where $x$ occurs.

Part 2 - Binary search
- If a string contains $x$, it must occur in the left or right half.
- Check if the left half contains $x$. Either way we can eliminate half of the word.
- Make sure to handle occurrences near the center correctly.
- Need $\log_2(10000) \approx 14$ queries.
Part 1 - Finding a word

- Idea: join words together to form query strings.
- The total length of all words is $26^4 \cdot 4$, but we would need 183 queries to fit that.
- Shave off some queries by using overlaps, for instance:
  - Keep a list of unused words.
  - Form a long query string by joining unused words.
  - If the answer is no, remove all 4-letter substrings from the list.
- The optimal solution only needs 46 queries to cover all words, look up *de Bruijn sequences* if you are interested.
Problem

Construct a permutation of maximal order on $n$ elements. Part of the permutation is already fixed. The order of a permutation is the least number of repetitions needed to return to the original configuration.

<table>
<thead>
<tr>
<th>Insight</th>
</tr>
</thead>
<tbody>
<tr>
<td>The order of a permutation is the least common multiple of its cycle lengths. It suffices to consider partitions of $n$ into cycles. It is optimal to only use cycle lengths $p^k$ with $p$ prime. For instance, instead of a 12-cycle use a 3-cycle and a 4-cycle.</td>
</tr>
</tbody>
</table>
Problem

Construct a permutation of maximal order on \( n \) elements. Part of the permutation is already fixed. The order of a permutation is the least number of repetitions needed to return to the original configuration.

Insight

- The order of a permutation is the least common multiple of its cycle lengths.
- It suffices to consider partitions of \( n \) into cycles.
- It is optimal to only use cycle lengths \( p^k \) with \( p \) prime.
- For instance, instead of a 12-cycle use a 3-cycle and a 4-cycle.
Solution

- Use dynamic programming:

\[ f(n, k) = \text{the maximal LCM over partitions of } n \text{ into powers of the first } k \text{ primes} \]

- Many other DP approaches look promising, but will not work.
- The values of \( f \) can exceed \( 10^{100} \), so need to use big numbers.
- Time complexity: \( \mathcal{O}(n^2) \).