NWERC 2010
Solutions to the problems

The Jury

Jacobs University Bremen

A - Fair Division

- Sort persons according to maximum contribution
- Tie-breaker: position in list
- `for (i=0 ... N-1)`
- `  contrib[i] = min(max[i] , price/(N-i))`
- `  price -= contrib[i]`
- Don't print a trailing space

A - Fair Division

- Sort persons according to maximum contribution
- Tie-breaker: position in list
- `for (i=0 ... N-1)`
- `  contrib[i] = min(max[i] , price/(N-i))`
- `  price -= contrib[i]`
- Don't print a trailing space

- Statistics: 119 submissions, 51 correct, first 27 minutes

H - Stock Prices

- While bid price larger than ask price, process deals
- Output prices or a dash if it doesn't exist

H - Stock Prices

- While bid price larger than ask price, process deals
- Output prices or a dash if it doesn't exist

- Statistics: 112 submissions, 48 correct, first 40 minutes

C - High Scores

- Loop over starting with *going left* or *right*
- Loop over where to *turn around*
- Count the number of moves until you are done

C - High Scores

- Loop over starting with *going left* or *right*
- Loop over where to *turn around*
- Count the number of moves until you are done

- Statistics: 192 submissions, 43 correct, first 34 minutes

E - Rankings

- Start with `newrank[i] = oldrank[i]`
- For a swap (i, j), increase/decrease `newrank[i|j]`
- Check consistency: if i and j swapped, `newranks` and `oldranks` must be in opposite order
- There are never question marks in the answer
- Topological sorting also works

E - Rankings

- Start with `newrank[i] = oldrank[i]`
- For a swap (i, j), increase/decrease `newrank[i|j]`
- Check consistency: if i and j swapped, `newranks` and `oldranks` must be in opposite order
- There are never question marks in the answer
- Topological sorting also works

- Statistics: 77 submissions, 37 correct, first 65 minutes

B - Free Goodies

- Sort the goodies to Petra's valuations
- $O(n^2)$ dynamic programming:
- `best[n goodies taken][Jan took k]`
- Also $O(n \log n)$ greedy solution possible!

B - Free Goodies

- Sort the goodies to Petra's valuations
- $O(n^2)$ dynamic programming:
- `best[n goodies taken][Jan took k]`
- Also $O(n \log n)$ greedy solution possible!

- Statistics: 25 submissions, 9 correct, first 135 minutes

F - Risk

- Binary search on the weakest link
- Use maximum flow algorithm to determine if answer is possible
- Graph vertices: source, sink, and 2 vertices for each land you control
- Graph edges:
  - source $\rightarrow$ 1st land (cap=num. armies)
  - 1st land $\rightarrow$ 2nd land (if connected)
  - 2nd land $\rightarrow$ sink (cap=needed armies)

F - Risk

- Binary search on the weakest link
- Use maximum flow algorithm to determine if answer is possible
- Graph vertices: source, sink, and 2 vertices for each land you control
- Graph edges:
  - source → 1st land (cap=num. armies)
  - 1st land → 2nd land (if connected)
  - 2nd land → sink (cap=needed armies)
- Statistics: 18 submissions, 8 correct, first 159 minutes

J - Wormly

- Note: legs $2 \ldots L - 1$ don't really matter
- Greedily move first leg, then last leg, then bubbles
- Repeat until finished
- Watch out for overflow

J - Wormly

- Note: legs $2 \dots L - 1$ don't really matter
- Greedily move first leg, then last leg, then bubbles
- Repeat until finished
- Watch out for overflow

- Statistics: 102 submissions, 9 correct, first 161 minutes

G - Selling Land

- Process rows one by one
- For each column c of row r, count the number of grass squares above (c, r)
- Process columns and keep a list of possible end squares
- This takes amortized time $O(1)$ per square

G - Selling Land

- Process rows one by one
- For each column c of row r, count the number of grass squares above (c, r)
- Process columns and keep a list of possible end squares
- This takes amortized time $O(1)$ per square

- Statistics: 21 submissions, ≥ 1 correct, first 254 minutes

D - Hill Driving

- Drive through the landscape with constant speed
- *(derive for two segments with equations, then use induction)*
- Binary search and check how much fuel is used
- Be careful:
  - Don't gain fuel when going downhill, but go faster
  - Don't exceed maximum speed
- Linear search possible too

D - Hill Driving

- Drive through the landscape with constant speed
- *(derive for two segments with equations, then use induction)*
- Binary search and check how much fuel is used
- Be careful:
  - Don't gain fuel when going downhill, but go faster
  - Don't exceed maximum speed
- Linear search possible too

- Statistics: 28 submissions, ?? correct, first ?? minutes

I - Telephone Network

- All sets of requests are possible, so we can add dummy requests to get bipartite graph with $deg(v) = 1$ for all v
- Reduce this graph modulo 2^{n-1} to get a bipartite graph with $deg(v) = 2$ for all v
- Split this graph in two graphs with all degrees 1 and you get two instances of the same problem with $n' = n/2$
- Solve recursively and construct solution

I - Telephone Network

- All sets of requests are possible, so we can add dummy requests to get bipartite graph with $deg(v) = 1$ for all v
- Reduce this graph modulo 2^{n-1} to get a bipartite graph with $deg(v) = 2$ for all v
- Split this graph in two graphs with all degrees 1 and you get two instances of the same problem with $n' = n/2$
- Solve recursively and construct solution

- Statistics: 6 submissions, ?? correct, first ?? minutes

The end