# Solution outlines

## BAPC Preliminaries

October 2, 2010

# B - Have a Nice Day

- To check if digits appear equal number of times: loop over all digits and compare counts.
- To check if it can be split: hardcode all possibilities.
- This leads to an $O(1)$ solution.

# F - Stock Market

- Loop over all numbers and keep current sum
- If the previous sum was $< 0$, set sum $= 0$ and set start to current
- Check whether current sum is bigger than the previous best and update best values
- This leads to an $O(N)$ solution

# E - The Great Cleanup

- Build a trie with a flag for each filename whether it should be deleted or not.
- To get the answer for a node:
  - If the current node and all children should be deleted, return 1
  - Otherwise, return the sum of the answers for all children, and add 1 if the current node should be deleted
- This leads to an $O((N_1 + N_2) * L)$ solution, where $L$ is the maximum length of a filename.

# J - My Cousin Obama

- Recursively check the line of ancestry, starting with A0
- For a person $i$:
  - If this is B0, return 0
  - result = INF;
  - If father[i] != 0 then
    result = min(result, answer for father[i])
  - If mother[i] != 0 then
    result = min(result, answer for mother[i] + 1)
  - return result
- Use *memoization* to store the result of each person
- This leads to an $O(N)$ solution.

# C - Searial Numbers

- Dynamic Programming solution:
- Work modulo $M$, keep a table of length $M$ that indicates how many guitars so far could have had a sum of $i\%M$
- Start with table[0] $= 1$
- For each guitar $i$
  - nexttable $=$ table
  - for each $0 \leq j < M$:
    - If table[j] $> 0$ then
      nexttable[(j+S[i])%M] $=$ max(nexttable[(j+S[i])%M], nexttable[j] $+ 1$)
  - table $=$ nexttable
- The answer is $table[0] - 1$
- This leads to an $O(N * M)$ solution, with $O(M)$ space

# H - Farmer John

- For each pair of points, check if they can be connected in a straight line (i.e. do not overlap with any fence)
- Construct a graph from this, run a shortest path algorithm on this graph
- This leads to an $O(N^3)$ solution

# A - Evolution

- Store the set of *used* DNA-strings as a bitmask (at most $2^{15} = 32768$ values)
- Create a recursive function that takes the previous creature and the set of used creatures, and returns the probability that the rest of the creatures are ancestors of this one
- For each of the creatures, calculate this value and use the sums to normalize all values
- Use *memoization* to store the intermediate results
- This leads to an $O(N^2 2^N)$ solution

# G - Acrobat Reader

- Sort the two sets of points (on X and then on Y, as long as you do it the same for both)
- To check the scaling, find maximal and minimal x and y values, and use this to find the relative scaling
- Then check each pair of points using the first point as origin and use the scaling
- Rotate one of the sets and do the check again for the other 3 rotations
- This leads to an $O(N \log N)$ solution

# D - Equal Is Not Really Equal

- Construct a graph with a node for each character
- For every pair of characters, add an edge in this graph
- The question now is: are there at least 2 Euler Paths in the graph?
- Because there is always 1 Euler Path in the graph, we can check in $O(N)$ time for each node if there are 2 differrent outgoing paths
- This leads to an $O(N * A)$ solution, where $A = 26$ is the alphabeth size

# I - Imagine

- Use a 2-dimensional Binary Indexed Tree to store the sums
- This gives $O(\log^2 N)$ update time and $O(\log^2 N)$ read time