- Problema A - Chopsticks

  Ordene o vetor da entrada em ordem crescente. Agora, é fácil perceber que, para minimizar a "badness" do conjunto de palitinhos, os dois menores devem ocupar posições vizinhas no vetor e o terceiro pode estar em qualquer posição à direita dos primeiros. Então, podemos resolver o problema com programação dinâmica, representando o estado com as variáveis p – indicando a posição atual no vetor – e r – indicando quantos conjuntos ainda faltam ser formados. Assumindo que o vetor comece a ser indexado por 0, sabemos que, se $\frac{N-p}{3} < r$, é impossível formar todos os r trios restantes. Caso contrário, devemos escolher a melhor opção entre duas: formar um par entre os palitinhos em p e em $p+1$ e pular para a posição $p+2$; ou ignorar o palitinho atual e pular para a posição $p+1$.

- Problema B - The Bus Driver Problem

  Esse problema é muito parecido com o "Load Balancing", que foi apresentado em aula. Basta combinar sempre a rota matutina mais curta com a rota noturna mais longa.

- Problema C - Caminhando pela Calçada

  Se passamos por k pontos com coordenadas inteiras, igualmente espaçados, depois que saímos do ponto (A, B) até chegar ao ponto (C, D), k deve dividir tanto $|A - C|$ quanto $|B - D|$. Como queremos maximizar k, ele deve ser exatamente o máximo divisor comum (MDC) entre $|A - C|$ e $|B - D|$. Os que usam o GCC podem economizar algumas poucas linhas de código na implementação, porque o compilador já fornece a função `__gcd()` para o cálculo do MDC.

- Problema D - Trapézio

  Escolha os trapezistas a partir daquele que tem a maior soma $P + F$ até aquele com a menor dessas somas. Se, seguindo essa ordem, algum deles tiver a força menor do que a soma do peso de todos os restantes, é impossível encontrar uma configuração válida.

  Para ajudar a ganhar intuição sobre esse problema, vamos provar que a solução localmente ótima é, de fato, escolher o trapezista com maior $P+F$. Para isso, vamos considerar dois trapezistas, i e j, tais que a soma do peso e força de i é maior que a de j, isto é,

  $$P_j + F_j < P_i + F_i. \tag{1}$$

  Agora, se o trapezista i não é capaz de sustentar os pesos de todos os outros trapezistas, temos:

  $$F_i < \sum_{k \neq i} P_k. \tag{2}$$

Somando, membro a membro, as desigualdes (1) e (2):

$$F_i + P_j + F_j < P_i + F_i + \sum_{k \neq i} P_k$$

$$\Rightarrow \qquad F_j < \sum_{k \neq j} P_k, \tag{3}$$

ou seja, o trapezista j, com soma de peso e força menor que a de i, também não é capaz de sustentar todos os outros pesos. Isso mostra que, se existe pelo menos um trapezista que pode ser posicionado no topo do trapézio, aquele com maior $P + F$ é um deles.

- Problema E - Assalto ao Banco Central

  Conte quantas chaves diferentes você consegue sequestrando cada subconjunto dos diretores. Dentre os subconjuntos que tornam possível a abertura do cofre, escolha aquele com menos elementos.

- Problema F - CD

  Basta usar qualquer estrutura de dados que permita buscas eficientes por um elemento para determinar quais os elementos comuns aos dois conjuntos.

- Problema G - Scarecrow

  Alguns times enviaram soluções corretas, mas muito mais complicadas do que o necessário. Basta ler os caracteres da entrada da esquerda para a direita e, quando encontrar um '.', posicionar um espantalho imediatamente à direita dessa célula.

- Problema H - Soya Milk

  O problema consiste apenas em calcular o volume de um prisma, o que é bem simples, mas tenha cuidado com o caso em que $\tan\theta > h/l$.

- Problema I - Claw Decomposition

  Basta checar se o grafo dado é bipartido. Faça um percurso no grafo e tente colori-lo com duas cores, sendo que dois vértices ligados por uma aresta não devem ter a mesma cor. Se isso for possível, o grafo é bipartido.

- Problema J - Coral Perfeito

  A nota final deve ser a média aritmética de todas as notas iniciais, então, se essa média não for inteira, a resposta é -1. Se houver solução, a resposta

deve ser o somatório das diferenças entre as notas maiores que a média e a própria média.