- Problema A - Aggressive Cows

  Dada uma distância d, é fácil checar em $\mathcal{O}(N)$ se é possível acomodar todas as vacas a uma distância de pelo menos d uma da outra. Então, faça uma busca binária para encontrar o máximo valor possível para d.

- Problema B - Suba os Ultras

  Soluções com complexidade $\mathcal{O}(N \log N)$ são suficientes para esse problema, mas vamos descrever uma solução em tempo linear. Primeiro, vamos definir como *picos* os máximos locais no perfil de altitudes e como *vales* os mínimos locais. Agora, vamos nos restringir a identificar os picos que devem passar por um vale baixo o suficiente para ser considerado ultra no caminho para um pico mais alto à sua direita. Depois, basta repetir o mesmo procedimento da direita para a esquerda e os picos identificados nas duas vezes são os ultras.

  O algoritmo é o seguinte: mantenha uma pilha para guardar as alturas dos vales e uma para guardar as alturas dos picos. Leia os pontos da esquerda para a direita e, se o ponto atual for um vale, adicione-o à pilha dos vales. Se o ponto for um pico menor que o pico do topo da pilha, insira-o à pilha dos picos; caso contrário, vá retirando os picos e vales das pilhas até encontrar um pico que é maior do que o atual, e vá calculando a proeminência de todos os picos que aparecem durante o procedimento.

- Problema C - Banda

  Teste todas as $\mathcal{O}(N^3)$ formações possíveis e escolha a que tem maior nível de entrosamento.

- Problema D - Paths in a Complete Binary Tree

  O comprimento do caminho da raiz da árvore até um nó qualquer é, no máximo, igual a $\log n$. Então, é sempre possível construir o percurso da raiz da árvore até o nó da consulta atual para respondê-la.

- Problema E - Guarda Costeira

  Queremos saber se o fugitivo percorre 12 milhas náuticas mais rapidamente do que a guarda percorre $\sqrt{12^2 + D^2}$ milhas náuticas. Para isso, precisamos apenas checar se $V_f/12 > V_g/\sqrt{12^2 + D^2}$.

- Problema F - Conta de Eletricidade

  Faça uma busca binária para encontrar o seu consumo.

- Problema G - Final do ICPC

  Existem outras soluções menos eficientes e mais intuitivas para o problema, mas vamos descrever uma com complexidade esperada linear no número de pontos da entrada. Vamos definir uma função que tem como argumentos dois conjuntos de pontos, A e B, e retorna o menor círculo que engloba todos os pontos de $A \cup B$, desde que todos os pontos de B estejam na borda desse círculo. Assim, o problema pode ser resolvido calculando essa função quando A é o conjunto dos pontos da entrada e B é o conjunto vazio.

  O algoritmo processa os pontos de A em ordem aleatória. A cada passo, ele checa se o próximo ponto p a ser processado pertence ao círculo que seria formado sem ele. Se ele já é englobado pelo círculo, ótimo, não precisamos nos preocupar com esse ponto; caso contrário, chamamos o algoritmo recursivamente com os conjuntos $A \setminus \{p\}$ e $B \cup \{p\}$. O caso base da recursão acontece quando temos três pontos na borda, que já determinam unicamente um círculo, ou quando não há mais pontos a serem englobados.

  Observe o código abaixo, que ilustra a implementação da parte principal do algoritmo.

```
circulo menor_circulo(pt A[],int na,pt B[],int nb) {
    if (na==0 || nb==3) {
        return constroi_circulo(B, nb);
    }

    random_shuffle(A, A+na);
    circulo C = menor_circulo(A, na-1, B, nb);
    if (!ponto_pertence_ao_circulo(A[na-1], C)) {
        B[nb++] = A[na-1];
        C = menor_circulo(A, na-1, B, nb);
    }

    return C;
}
```

- Problema H - O Bolo de Apostas

  A maioria dos alunos resolveu o problema com um algoritmo linear usando programação dinâmica, mas também é possível resolvê-lo com um método de divisão e conquista. Para isso, divida o vetor de entrada em duas metades e retorne o máximo entre as seguintes opções: a soma máxima na metade esquerda do vetor; a soma máxima na metade direita do vetor; a soma máxima que usa o elemento mais à esquerda da metade esquerda e o elemento mais à direita da metade direita.

- Problema I - Solve It

  Todas as parcelas do lado esquerdo da equação são monotônicas não-crescentes no intervalo $[0, 1]$, então, podemos usar o método da bisseção nesse intervalo para encontrar a raiz.

- Problema J - Jogo de Varetas

  Conte o número de pares de lados iguais e divida esse número por dois.