Exercício - Revisão

1) Quando não devemos usar recursividade?

2) Quando um algoritmo é recursivo em cauda?

3) Quais são as três etapas utilizadas por algoritmos de divisão e conquista? Por que utilizar esse paradigma de projeto de algoritmos?

4) Quais as desvantagens do uso de técnicas de tentativa e erro? Quando o uso dessas técnicas é interessante?

5) Como poderíamos resolver o fibonacci usando programação dinâmica? Ele apresenta sobreposição de subproblemas?

6) Suponha que você tenha vários arquivos no seu computador, e deseja inserir o maior número possível de arquivos no seu pendrive. O pendrive, no entanto, tem um limite de capacidade. Forneça um algoritmo guloso para maximizar o número de arquivos no pendrive. Esse algoritmo encontra sempre a solução ótima?

7) Quais propriedades são necessárias para que um problema seja NP-Completo?

8) Problemas NP-Difícil são mais fáceis que problemas NP-Completo?

9) Como caracterizar um problema como sendo da classe NP?