Instituto Infnet
Disciplina: Algoritmos Avançados
Aluno(a): Nizshime Samara Cotta Mathildes Scorzello

QUESTÃO 4. # Defina problemas P, NP e NP-completos.

- **P (Problemas Polinomiais):**
  Problemas em P são aqueles que podem ser resolvidos em tempo polinomial, o que significa que o tempo de execução do algoritmo para resolver o problema é limitado por uma função polinomial do tamanho da entrada.

Exemplo: Multiplicação de dois números inteiros.
Um algoritmo de multiplicação de números inteiros tem uma complexidade de tempo polinomial, pois o tempo de execução cresce em função do número de dígitos nos números inteiros.

- **Problemas NP (Não-Determinísticos Polinomiais):**
  Problemas em NP são aqueles para os quais, dado um candidato a solução, você pode verificar a validade da solução em tempo polinomial.
  No entanto, encontrar a solução em si pode não ser tão eficiente.
  Em programação, problemas em NP geralmente envolvem encontrar soluções que podem ser verificadas rapidamente.

Exemplo: O problema de encontrar um conjunto de números que some a um valor específico (Subset Sum).
Dado um conjunto de números e um valor, verificar se existe um subconjunto cuja soma seja igual ao valor é rápido, mas encontrar esse subconjunto é mais desafiador.

- **Problemas NP-Completos (Não-Determinísticos Polinomiais Completos):**
  Problemas NP-completos são uma classe especial de problemas em NP que são considerados entre os mais difíceis. Não se sabe se é possível encontrar soluções eficientes
  (tempo polinomial) para eles.
  Em programação, esses problemas são notoriamente difíceis de resolver e muitas vezes requerem abordagens heurísticas.

Exemplo: O Problema do Caixeiro Viajante, onde você deve encontrar a rota mais curta que visite um conjunto de cidades exatamente uma vez e retorne ao ponto de partida.
Não existe um algoritmo conhecido que resolva todos os casos em tempo polinomial.

Classificação:
Na programação, a classificação de problemas em P, NP ou NP-completos é usada para avaliar a dificuldade e a viabilidade de resolução eficiente.
Problemas em P são aqueles que podemos resolver eficazmente, ***problemas NP*** são desafiadores, mas podemos verificar soluções rapidamente, e ***problemas NP-completos*** são os mais difíceis e desafiadores de todos.

A classificação é feita com base em evidências teóricas, e a questão de se **P** é igual a **NP** (ou seja, se todos os problemas em **NP** podem ser resolvidos em tempo polinomial)
é uma das questões mais importantes e não resolvidas na ciência da computação.

QUESTÃO 5. # Explique o funcionamento dos algoritmos abaixo.

- **Bubblesort:**
  O Bubblesort é um algoritmo de ordenação.
  Ele é um algorítimo de ordenação, serve para colocar os itens de uma lista em ordem.
  No funcionamento deste algorítimo, você começa no início da lista e compara o primeiro número com o próximo.
  Se o primeiro for maior que o próximo, você os troca de lugar.
  Você faz isso para todos os pares de números na lista até que nenhuma troca seja necessária durante uma passagem completa.
  Em seguida, você repete esse processo até que a lista esteja completamente ordenada.
  O Bubblesort não é a maneira mais rápida de fazer isso, mas é fácil de entender.

- **Insertion Sort:**
  O Insertion Sort é um algoritmo de ordenação, ele trabalha inserindo cada elemento no local apropriado em relação aos elementos já ordenados.
  É um algoritmo de ordenação simples e eficaz para listas pequenas ou parcialmente ordenadas. Ele é indicado para listas pequenas ou parcialmente ordenadas.

- **Mergesort:**
  O Merge Sort é um algoritmo de ordenação eficiente e geralmente indicado para listas grandes onde a estabilidade é necessária e o uso adicional de memória não é uma preocupação.
  Ele trabalha dividindo a lista em sub-listas menores (utiliza a abordagem "dividir para conquistar"), ordenando-as recursivamente e em seguida, combinando-as para produzir a lista ordenada final.
  Este algorítimo é mais eficiente do que o Bubblesort e o Insertion Sort para listas grandes.

Em resumo, a respeito das três técnicas de Algorítimos:
Esses algoritmos ajudam a organizar informações em ordem, mas cada um tem suas próprias características e eficiências. A melhor escolha do algoritmo vai depender do tamanho dos dados e da eficiência desejada para o problema.