

LISTA OBRIGATÓRIA PARCIAL DE PROJETO E ANÁLISE DE ALGORITMO

SEMANA 5

Para os problemas a seguir faça:

- a) Estruture a solução em pseudo-linguagem;
 - b) Calcule a complexidade do algoritmo;
 - c) Implemente o algoritmo e teste-o.
1. Dados dois conjuntos A e B , representados como vetores de tamanhos m e n , respectivamente, $m \leq n$, determine o conjunto interseção dos conjuntos dados em tempo $O((n+m) \log m)$.
 2. Suponha que sua entrada seja um vetor de n strings. Adapte o *HeapSort* para ordenar este vetor de acordo com o tamanho das strings. Caso duas strings tenham o mesmo tamanho, o vetor de saída deve preservar a ordem de ocorrência das strings no vetor original, ou seja se s precede t no vetor original e $\text{tamanho}(s) = \text{tamanho}(t)$, então s também deve preceder t no vetor de saída.
 3. Considere um vetor I de intervalos inteiros (x,y) , $x < y$.
 - a) Elabore um algoritmo $O(n \log n)$ para gerar todos os números inteiros dos intervalos, em que números iguais aparecem consecutivos, o n é a soma dos tamanhos dos intervalos.
Ex: entrada: $[(1,3), (7,9), (1,8)]$ saída: $[1,1,2,2,3,3,4,5,6,7,7,8,8,9]$
 - b) Dado um número z , e um vetor obtido em (a) localize se z ocorre no vetor. Em caso afirmativo, qual a posição inicial e final em que ele ocorre. Seu algoritmo deve rodar em $O(\log n)$. Este problema é similar ao problema 1520 do URI.
 4. Adapte o algoritmo de Ordenação por Contagem para dado um vetor A de n inteiros não negativos, encontrar o número de pares distintos (i, j) , tal que $j > i$ e $A[i] = A[j]$.
 5. Considere que você possua k sequências de inteiros ordenadas, cada uma delas armazenada em um vetor em memória primária. Usando a estratégia do *k-way-merge*, elabore um algoritmo para ordenar todos os elementos das k sequências. O vetor de saída também está em memória primária.