Faculdade de Tecnologia de Franca "Dr. Thomaz Novelino"
Curso Tecnológico Superior em Análise e Desenvolvimento de Sistemas

ESTRUTURAS DE DADOS – 2020/2

Prof. Me. Fausto Gonçalves Cintra – professor@faustocintra.com.br

LEIA COM ATENÇÃO *TODAS* AS INSTRUÇÕES *ANTES* DE COMEÇAR A FAZER O TRABALHO

TRABALHO 1 (T1)

1 INSTRUÇÕES GERAIS

1. O trabalho é ***estritamente individual***.
2. A trabalhos idênticas, ou com alto grau de semelhança, será atribuída a nota ZERO.
3. O valor do trabalho é 10,0 (dez), conforme explicado no documento *[IED001-00] Apresentação*.

2 INSTRUÇÕES ESPECÍFICAS

1. Baixe os arquivos `covid-19.js` e `Ficha comparativa de algoritmos de ordenação.docx` que estão anexados junto a estas instruções.
2. Na sua máquina, instale o Node.js versão LTS a partir do *site* oficial. No momento em que essas instruções estão sendo redigidas, a versão LTS do Node.js é a 12.19.0.
   a) Não é recomendável executar no GitPod os testes dos algoritmos de ordenação, especificados nas instruções que vêm a seguir. A execução de algoritmos mais lentos, como o *bubble sort*, pode levar muito tempo, fazendo com que o GitPod interrompa o *workspace* por inatividade, inviabilizando a conclusão do teste.
   b) No GitPod, você pode clicar com o botão direito sobre o nome do arquivo e selecionar "Download" para baixar os arquivos dos algoritmos para o seu computador. Uma sugestão de editor local para editar os arquivos e efetuar os testes é o Visual Studio Code (que pode ser baixado aqui).
3. A massa de dados para teste se encontra no arquivo `covid-19.js`. Este arquivo contém 251.064 registros. Note que, no arquivo `Ficha comparativa de algoritmos de ordenação.docx`, deve ser preenchida uma tabela com o gasto de tempo e memória para 1.000, 25.000, 100.000 e para todos os registros. **Você deverá usar a função slice() para gerar um subvetor da amostra nos três primeiros casos, e submeter o subvetor gerado aos algoritmos de ordenação**.
4. Submeta cada amostra ao cada um dos algoritmos de ordenação, **passando a eles uma função de comparação que organize os registros primeiramente pelo atributo `date`, depois pelo atributo `state` e, finalmente, pelo atributo `city`**. Note que se trata de uma **única função** de comparação trabalhando a ordenação por três atributos diferentes.
5. É importante executar todos os testes sempre na mesma máquina (computador), para que os resultados possam ser comparados entre si. À medida que os testes vão sendo executados,

preencha a Ficha Comparativa com os resultados obtidos. **Ao anotar o tempo, efetue a conversão dos milissegundos para horas, minutos, segundos e milissegundos**.

   a) **Ao preencher a linha MELHOR RESULTADO, anote, à frente dos campos "Tempo" e "Memória", o nome do algoritmo que obteve o melhor desempenho nos quesitos medidos**.

6. Deverão ser entregues os seguintes itens, reunidos em um arquivo ZIP:
   a) a Ficha Comparativa de Algoritmos de Ordenação, devidamente preenchida; e
   b) os arquivos de código-fonte JavaScript utilizados para executar os testes.
7. Após criar o arquivo ZIP, faça *upload* dele no Microsoft Teams, na tarefa "**TRABALHO 1 (T1)**", até a data e hora de vencimento. ***NÃO SE ESQUEÇA DE CLICAR SOBRE O BOTÃO DE ENVIO AO FINAL***!
   a) Trabalhos enviados com até 24h de atraso terão 25% de desconto na nota; até 48h horas de atraso, 50% de desconto; até 72h, 75% de desconto. Não mais será possível enviar o trabalho após 72h do vencimento.