Instituto Federal de Santa Catarina - Câmpus Garopaba
Curso Superior de Tecnologia em Sistemas para Internet
Unidade Curricular: Estrutura de Dados - EDA202203
Professor: Luiz Antonio Schalata Pacheco
Aluno:
Data: 25/05/2023

Avaliação 02

1. Explicação do funcionamento e implementação de um dos métodos de ordenação:
   (a) Shell Sort
   (b) Merge Sort
   (c) Quick Sort

   O Merge Sort e o Quick Sort usam recursividade, portanto, isso também precisa ser abordado na apresentação.

2. Realização de um comparativo do tempo de execução dos algoritmos de ordenação estudados (Bubble Sort, Selection Sort, Insertion Sort, Shell Sort, Merge Sort e Quick Sort):
   (a) Preencher um vetor ordenado com os elementos do vetor da questão anterior, medindo o tempo de execução;
   (b) Comparar o tempo obtido com o tempo de execução dos algoritmos de ordenação;
   (c) Fazer uma média de várias medidas.

3. **ATENÇÃO**:
   (a) Apresentação no 15/06/2023 com relatório e código entregues no SIGAA.
   (b) Nota com mesmo peso das avaliações escritas.
   (c) Os critérios incluem a codificação, o trabalho escrito (relatório com fundamentação teórica, etapas de desenvolvimento e conclusão) e a apresentação.
   (d) As arguições podem ser destinadas a um membro específico da equipe, que terá 2 ou 3 membros. Seis equipes devem ser compostas na turma.

   Exemplo:

```python
import numpy as np
import timeit
from VetorOrdenado import VetorOrdenado

def bubble_sort(vetor): # Recebe um vetor nao ordenado como parametro
  ...
  ...
  return vetor # Retorna um vetor ordenado

def selection_sort(vetor):
def insertion_sort(vetor):
def shell_sort(vetor):
def merge_sort(vetor):
```

```
14 def quick_sort(vetor, inicio, final):
15
16 def insere_ordenado(valores):
17   vetor = VetorOrdenado(len(valores)) # Classe implementada em aula
18   for i in range(len(valores)):
19     vetor.insere(valores[i])
20
21
22 if __name__ == "__main__":
23
24   # Gera vetor de 5000 posicoes com valores aleatorios entre 0 e 999
25   vetor = np.random.randint(1000, size=5000)
26   # print(vetor, len(vetor), type(vetor))
27
28   tempo_inicial = timeit.default_timer()
29   insere_ordenado(vetor.copy())
30   tempo_final = timeit.default_timer()
31   print(f'Vetor Ordenado: {tempo_final - tempo_inicial:.2f}')
32
33   tempo_inicial = timeit.default_timer()
34   bubble_sort(vetor.copy())
35   tempo_final = timeit.default_timer()
36   print(f'Bubble Sort: {tempo_final - tempo_inicial:.2f}')
37   ...
```