

# Trabalho 2

## Sistema de Ranking – Cyber Arena Champions

Prof. Adriana Scherer

Disciplina: 4645G-04 - Algoritmos e Estruturas de Dados I

Escola Politécnica — PUCRS

### Situação Problema

Você e sua equipe (3 alunos) foram contratados pela Epic Games Studios para desenvolver o sistema de ranking do jogo Cyber Arena Champions. O sistema atual, baseado em listas lineares, está lento e não fornece análises estratégicas adequadas para os torneios competitivos.

### O Desafio

Implementar um sistema eficiente usando `BinarySearchTreeOfInteger` (e suas dependências) – implementadas na disciplina – que processe rankings de jogadores e forneça insights valiosos sobre a comunidade de jogadores.

### Funcionalidades que devem ser implementadas em sua aplicação

#### 1) Gerenciamento básico de pontuações:

- a) Inserção de pontuações: considere as seguintes pontuações iniciais, onde cada pontuação é de um jogador específico:

```
Integer [] pontuacoesIniciais = {1500, 1200, 1800, 900, 2100, 750, 1300, 1700, 950, 2000};
```

- b) Atualização de pontuação de jogadores:
  - a. derrota (remove uma pontuação maior e insere uma menor);
  - b. vitória (remove uma pontuação menor e insere uma maior).
- c) Contagem total de jogadores no ranking

#### 2) Visualizações do ranking:

- a) Ranking em ordem crescente (classificação oficial)
- b) Visualização em pré-ordem, pós-ordem e por níveis

#### 3) Consultas rápidas:

- a) Buscar uma pontuação específica
- b) Buscar a pontuação mais baixa

#### 4) Análises Estratégicas:

Para atender a estas funcionalidades, utilize como estrutura auxiliar LinkedListOfInteger – não utilize vetor.

a) Mostrar análise da comunidade: quantidade de jogadores por faixas de skill, conforme tabela abaixo:

Faixa de Skill	Pontos
Inicantes	< 1000
Intermediários	1000 a 1499
Avançados	1500 a 1999
Experts	+2000

b) Mostrar pontuações que estejam próximas de uma pontuação referência.

- Caso de teste: Entrada: Pontuação referência 1500  
Limite 200  
Saída: 1300 (diferença: 200)  
1400 (diferença: 100)  
1500 (diferença: 0)  
1600 (diferença: 100)  
1700 (diferença: 200)

c) Mostrar a média das pontuações.

### O que não pode ser feito

Utilizar outras estruturas de dados que não sejam as implementadas em aula e permitidas para este trabalho, a saber: BinarySearchTreeOfInteger, LinkedListOfInteger, Queue, EmptyQueueTreeOfInteger, EmptyTreeException.

Alterar os métodos implementados em aula, das classes BinarySearchTreeOfInteger, LinkedListOfInteger, Queue, EmptyQueueTreeOfInteger, EmptyTreeException.

Qualquer violação destes dois espectos invalidará o trabalho.

### Entrega esperada

- a) Software funcional com todas as operações implementadas – um arquivo .zip apenas com os arquivos .java de todas as classes utilizadas para solucionar o problema (classes básicas (BinarySearchTreeOfInteger, LinkedListOfInteger, Queue, EmptyQueueTreeOfInteger, EmptyTreeException) + app).
- b) Relatório - um arquivo .pdf - contendo:
  - a. Identificação do Gx
  - b. Integrantes (nome completo e matrícula)
  - c. Rápida descrição textual de todas as classes utilizadas no projeto
  - d. Screenshots da execução de cada uma das funcionalidades
  - e. Screenshots da visualização DOT da árvore após a execução do item a (inserção de das pontuações iniciais e do item b (atualização de pontuação de jogadores)

- f. Análise de complexidade Big O das operações implementadas no APP com as justificativas
- g. Fontes de consulta/ajuda utilizadas para elaboração do trabalho, explicitando em que parte a ajuda foi utilizada e o tipo de ajuda recebida
- h. Tipo de IA usada e os links com a sequência de prompts e o resultado do código mostrando o caminho e os refinamentos realizados. Justificar o porquê do uso deste recurso e o que ele auxiliou na execução do trabalho.
- i. Lições aprendidas com este trabalho
- j. Dificuldades encontradas e como as superaram
- k. Declaração de autoria e integridade acadêmica (cada membro descreve sua contribuição)

Postar na sala de entrega um arquivo .zip contendo: todas as classes que solucionam o problema e o relatório, com o seguinte padrão **Gx\_aluno1\_aluno2\_aluno3.zip**

## Como será avaliado

<p>Funcionalidade e requisitos do software entregue</p> <p>Valor: 5.0</p>	<p>5.0 - atende totalmente</p> <p>4.0 - atende parcialmente e faltam detalhes nas funcionalidades solicitadas</p> <p>3.0 – atende parcialmente, mas a maioria das funcionalidades não funcionam</p> <p>2.0 - tentativa que necessita de revisão</p> <p>1.0 – apenas as classes básicas já implementadas em aula foram entregues</p> <p>0 - não atende ao solicitado</p>	
<p>Correta utilização das Estruturas de Dados especificadas e seus métodos</p> <p>Valor: 2.0</p>	<p>2.0 - atende totalmente</p> <p>0 - não atende ao solicitado</p>	
<p>Relatório</p> <p>Valor: 3.0</p>	<p>3- atende totalmente</p> <p>2.0- atende parcialmente faltam alguns detalhes solicitados</p> <p>1.0 – tentativa que necessita de revisão</p> <p>0- não atende ao solicitado</p>	