

Nome do professor: Carlos Veríssimo

Nome do Aluno: Carlos Eduardo Chiquesi De Almeida

Nome da disciplina: PROGRAMAÇÃO ORIENTADA A OBJETOS

**Diagrama de Caso de Uso**:

Atores:

* Usuário do Campeonato: Este ator representa qualquer pessoa que interaja com o sistema, como os organizadores do campeonato.

Casos de Uso (Ações do Sistema):

* Gerenciar Times: Este caso de uso incluiria as ações de adicionar times ao campeonato, contratar jogadores para times e listar jogadores de um time.
* Gerenciar Partidas: Este caso de uso incluiria a ação de agendar partidas no campeonato.
* Gerar Tabela de Classificação: Este caso de uso representaria a ação de gerar a tabela de classificação com base nos resultados das partidas.

**Diagrama de Classes**:

+----------------------------+ |

CampeonatoPaulista |

+-----------------------------+ |

* nome: String |
* ano: int |
* times: List<Time> |
* partidas: List<Partida> |
* tabelaClassificacao: |

List<TabelaClassificacao> |

+----------------------------------+

+----------------------------------+

Partida |

+----------------------------------+

* data: Date |
* resultado: String |

+----------------------------------+ +----------------------------------+

Time |

+----------------------------------+

* nome: String |
* cidade: String |
* jogadores: List<Jogador> |

+----------------------------------+

+----------------------------------+

TabelaClassificacao |

+----------------------------------+

* pontos: int |
* partidasJogadas: int |
* vitorias: int |
* empates: int |
* derrotas: int |
* golsMarcados: int |
* golsSofridos: int |

+----------------------------------+

+----------------------------------+

Jogador |

+----------------------------------+

* nome: String |
* numero: int |
* posicao: String |
* dataNascimento: Date |

+----------------------------------+

**Encapsulamento:**

1. Classe Time:
   1. encapsulamento é implementado por meio do uso de campos privados (private) para nome, cidade e jogadores. Os métodos contratarJogador e listarJogadores fornecem uma interface controlada para modificar e acessar a lista de jogadores, garantindo que ela seja manipulada apenas de maneira apropriada.
2. Classe TabelaClassificacao:
   1. encapsulamento é aplicado usando campos privados para todas as propriedades, como time, pontos, partidasJogadas, etc. Eles não podem ser acessados diretamente de fora da classe.
3. Classe Partida:

Os campos data, timeCasa, timeVisitante e resultado são privados, e o resultado da partida só pode ser definido por meio do método definirResultado.

1. Classe Jogador:

Os campos nome, numero, posicao e dataNascimento são privados, protegendo os detalhes do jogador de acesso direto.

1. Classe CampeonatoPaulista:

Os campos nome, ano, times, partidas e tabelaClassificacao são encapsulados, permitindo que o objeto da classe controle o acesso e a modificação desses dados.

**Baixo Acoplamento:**

1. Classe Time:

A classe Time não possui conhecimento direto das outras classes, como CampeonatoPaulista ou Partida, o que indica um baixo acoplamento.

2. Classe CampeonatoPaulista:

A classe CampeonatoPaulista mantém uma lista de times e partidas, mas ela não conhece detalhes internos sobre as classes Time ou Partida. Isso reduz o acoplamento entre as classes.

3. Classe Partida:

A classe Partida conhece apenas os objetos de Time relacionados a ela (como timeCasa e timeVisitante) e não está ciente de detalhes sobre o CampeonatoPaulista.

4. Classe TabelaClassificacao:

A classe TabelaClassificacao tem uma dependência em relação à classe Time, mas isso é esperado, pois precisa das informações do time para calcular a pontuação.