Pontifícia Universidade Católica do Rio Grande do Sul
Programação Orientada a Objetos
Prof. Marcelo H. Yamaguti
2024/1

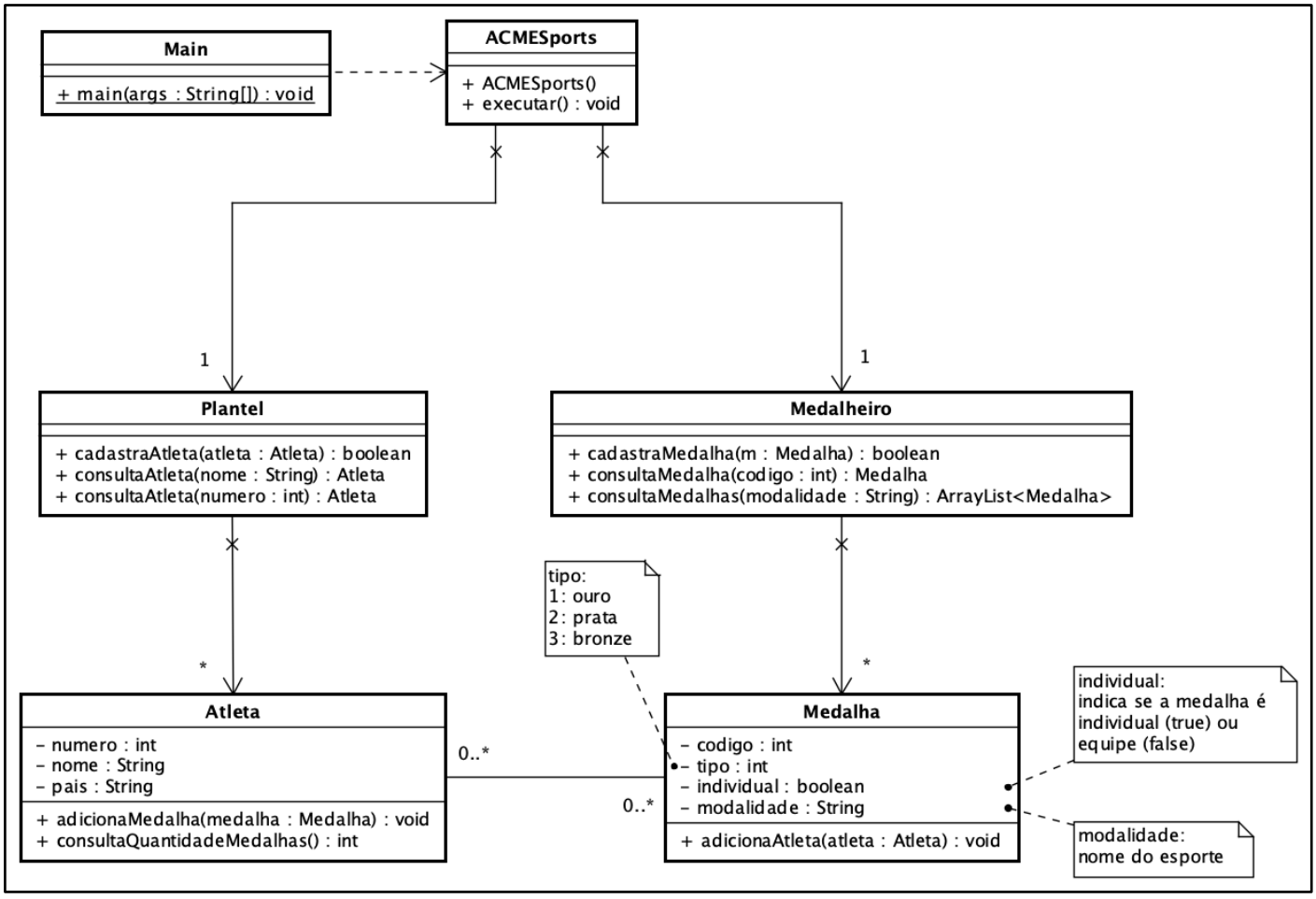
Exercício de Avaliação 1

1. **Enunciado geral:**

A ACMESports é uma empresa que desenvolve software para apoio a competições esportivas.

Você será responsável pelo desenvolvimento de um aplicativo que controlará os atletas e as medalhas de uma competição internacional.

O analista de sistemas identificou as seguintes classes iniciais, com alguns atributos e relacionamentos apresentados a seguir.



O analista identificou operações básicas iniciais das classes:

- **Atleta**: classe que representa um atleta da competição:
  - **adicionaMedalha**(Medalha): adiciona uma nova medalha ao atleta.
  - **consultaQuantidadeMedalhas**(): retorna a quantidade de medalhas do atleta.
- **Medalha**: classe que representa uma medalha da competição:
  - **adicionaAtleta**(Atleta): adiciona um atleta à medalha.
- **Plantel**: classe catálogo que gerencia o cadastro de atletas:

    - **cadastraAtleta(Atleta)**: recebe como parâmetro um novo Atleta e o cadastra no sistema. Não pode haver atletas com o mesmo número. Retorna *true* se o cadastro teve sucesso; ou false em caso contrário.
    - **consultaAtleta(String)**: retorna o atleta com o nome indicado. Se não houver nenhum atleta com este nome retorna *null*.
    - **consultaAtleta(int)**: retorna o atleta com o número indicado. Se não houver nenhum atleta com este número retorna *null*.
- **Medalheiro**: classe catálogo que gerencia o cadastro de medalhas:
    - **cadastraMedalha(Medalha)**: recebe como parâmetro uma nova medalha e a cadastra no sistema. Não pode haver medalhas com o mesmo código. Retorna *true* se o cadastro teve sucesso; ou *false* em caso contrário.
    - **consultaMedalha(int)**: retorna a medalha com o código indicado. Se não houver medalha com este código retorna *null*.
    - **consultaMedalhas(String)**: retorna uma coleção de medalhas com a modalidade indicada. Se não houver nenhuma medalha com esta modalidade retorna *null*.
- **ACMESports**: classe do aplicativo:
    - **ACMESports()**: construtor do aplicativo.
    - **executa()**: executa o funcionamento do aplicativo.
- **Main**: classe principal (inicial) do sistema:
    - **main(String[])**: cria um objeto ACMESports e depois chama o método executa().

O método executa() da classe ACMESports deve realizar a sequência de passos:

1. **Cadastrar atletas**: lê todos os dados de cada atleta e, se o número não for repetido, cadastra-o no sistema. Para cada atleta cadastrado com sucesso no sistema, mostra os dados do atleta no formato: **`1:número,nome,país`**
2. **Cadastrar medalhas**: lê todos os dados de cada medalha e, se o código não for repetido, cadastra-a no sistema. Para cada medalha cadastrada com sucesso no sistema, mostra os dados da medalha no formato: **`2:codigo,tipo,é individual?,modalidade`**
3. **Cadastrar medalhas e atletas correspondentes**: adiciona uma medalha para cada atleta e vice-versa. Para cada cadastramento com sucesso mostra os dados no formato: **`3:código,número`**
4. **Mostrar os dados de um determinado atleta por número**: lê o número de um determinado atleta. Se não existir um atleta com o número indicado, mostra a mensagem de erro: "**`4:Nenhum atleta encontrado.`**". Se existir, mostra os dados do atleta no formato: **`4:número,nome,país`**
5. **Mostrar os dados de um determinado atleta por nome**: lê o nome de um determinado atleta. Se não existir um atleta com o nome indicado, mostra a mensagem de erro: "**`5:Nenhum atleta encontrado.`**". Se existir, mostra os dados do atleta no formato: **`5:número,nome,país`**
6. **Mostrar os dados de uma determinada medalha**: lê um código de medalha. Se não existir uma medalha com o código indicado, mostra a mensagem de erro: "**`6:Nenhuma medalha encontrada.`**". Se existir, mostra os dados da medalha no formato: **`6:codigo,tipo,é individual?,modalidade`**
7. **Mostrar os dados dos atletas de um determinado país**: lê o nome de um país. Se não existir nenhum país com o nome indicado, mostra a mensagem de erro: "**`7:Pais nao encontrado.`**". Se existir, mostra os dados de cada atleta no formato: **`7:número,nome,país`**
8. **Mostrar os dados atletas de um determinado tipo de medalha**: lê o tipo de uma medalha. Se não houver nenhum atleta com o tipo de medalha indicado, mostra a

mensagem de erro: "**`8:Nenhum atleta encontrado.`**", caso contrário, mostra os dados de cada atleta no formato: **`8:número,nome,país`**

9. **Mostrar os dados atletas de uma determinada modalidade**: lê uma modalidade. Se não houver a modalidade no sistema, mostra a mensagem de erro: "**`9:Modalidade não encontrada.`**" Se uma medalha não tiver atleta, mostra uma mensagem no formato: **`9:modalidade,tipo,Sem atletas com medalha.`** Caso contrário, mostra os dados de cada atleta da medalha no formato: **`9:modalidade,tipo,número,nome,país`**
10. **Mostrar os dados do atleta com mais medalhas**: localiza o atleta com maior número de medalhas. Se não houver atletas com medalhas, mostra a mensagem de erro: "**`10:Nenhum atleta com medalha.`**". Caso contrário, mostra os dados do atleta e medalhas no formato: **`10:número,nome,país,Ouro:quantidade,Prata:quantidade,Bronze:quantidade`**

2. Definição do exercício:

O objetivo do exercício é implementar um sistema que capaz de atender as necessidades da empresa descrita no enunciado geral, e que atenda as restrições a seguir:

- A entrada de dados ocorrerá por leitura de arquivo de texto. Ajuste a classe ACMESports para ler e escrever em arquivos: veja na área Moodle da disciplina > Módulo: Materiais de apoio > CÓDIGOS AUXILIARES > Redirecionamento de entrada/saída de dados para arquivos.
- Os dados de entrada estarão no arquivo **'dadosin.txt'** e a saída será gravada no arquivo **'dadosout.txt'**
    - No passo **1. Cadastrar atletas**: cada linha corresponde ao número, nome e país de um atleta. Quando o número lido for -1, não há mais atletas a serem cadastrados.
    - No passo **2. Cadastrar medalhadas**: cada linha corresponde ao código, tipo, se é individual, e modalidade de uma medalha. Quando o código lido for -1, não há mais medalhas a serem cadastradas.
    - No passo **3. Cadastrar medalhas e atletas correspondentes**: cada linha corresponde ao código de uma medalha e ao número de um atleta correspondente. Quando o código lido for -1, não há mais dados a serem cadastrados.
    - As últimas linhas do arquivo de entrada correspondem a:
        - Número do atleta para o passo 4.
        - Nome do atleta para o passo 5.
        - Código de uma medalha para o passo 6.
        - Nome de um país para o passo 7.
        - Tipo de uma medalha para o passo 8.
        - Modalidade para o passo 9.
- Toda entrada e saída de dados com o usuário deve ocorrer apenas na classe ACMESports.
- É permitida a criação de novos métodos, atributos, classes e relacionamentos, mas as informações definidas no diagrama de classes original não podem ser alteradas.
- O diagrama de classes deve ser atualizado conforme as alterações realizadas e deve ser entregue em arquivo Astah ou PDF.

3. Critérios de avaliação

O exercício será avaliado conforme os seguintes critérios:

- Diagrama de classes atualizado e coerente com a solução: 1 ponto.
- Implementação correta conforme especificação e diagrama de classes: 4 pontos.

- Execução correta dos passos solicitados: 5 pontos.

- *Ponto extra (opcional) de 1 ponto (máximo de 10 pontos): após a execução dos passos indicados:*
    - *Mostrar o quadro geral de medalhas por país: nome de cada país, quantidade de medalhas de ouro, quantidade de medalhas de prata, quantidade de medalha de bronze.*
    - *Mostrar o quadro geral de medalhas completo: para cada país o seu nome, quantidade total de medalhas, e dados de cada atleta (nome do atleta e a quantidade de cada tipo de medalha).*

4. **Entrega:**
    - A entrega do exercício envolverá:
        - arquivos dos códigos-fonte do sistema (e demais arquivos necessários para a compilação do sistema).
        - arquivo com o diagrama de classes atualizado.
    - Deverá ser gerado um arquivo compactado (.zip ou .rar), com os itens acima, e entregue na tarefa da área Moodle da disciplina.
    - **Data de entrega**: 1º / 4 / 2024.

5. **Considerações finais:**
    - O exercício deve ser desenvolvido individualmente.
    - A implementação deve seguir o Java Code Conventions para nomes de identificadores e estruturas das classes.
    - Não será aceito exercício com erros de compilação. Programas que não compilarem corretamente terão nota zerada.
    - A cópia parcial ou completa do exercício terá como consequência a atribuição de nota 0 (zero) aos exercícios dos alunos envolvidos. Para análise de similaridade será utilizado o MOSS (https://theory.stanford.edu/~aiken/moss/).